

Completo Concetto di Ambiente

fino ad ora

$\{ \text{Exp + ID Costanti}$
 $\text{Dichiarazioni + ID Costanti} \}$ } \Rightarrow AMBIENTE

Associazione $ID \leftrightarrow$ Oggetto denotato

SEMANTICA STATICÀ

CHE VERIFICA i vincoli

Concetto di TIPO (dell'Oggetto Denotato)

CONTESTAZIONE
(vincoli di TIPO)

Per Arrivare ai COMANDI Bisogna Aggiungere la MEMORIA

MEMORIA:

Capiere qual'è il Ruolo dell' ID Nell' Assegnamento



Il problema a dx è che a destra può avere significati diversi:

MAIN

:

x

add(y,z)

add(..)

:

x

L'AMBIENTE Cambia

Cosa può contenere x?
Che valore contiene x?

Sono 2 CASE BEN DIVERSE

Aliasing \Rightarrow x e y possono riferirsi alla stessa cella di memoria

Mi serve allora un altro tipo di associazione, ad ogni ID un Oggetto Denotato e anche il concetto di locazione

LOCAZIONE:

Concetto astratto della cella di memoria (a livello logico o rappresenta)



Ho 2 parti: **STATICA** \rightarrow tipo (Nell' AMBIENTE)

DINAMICA \rightarrow Valore Contenuto (Nella MEM.)

MEMORIA → Insieme di Associazioni Tra Locazioni e Valori Memorizzabili (Non più Denotabili)

VALORI: **■ Esprimibile** se RAPPRESENTABILE Mediante EXP
■ Denotabile (con Ambienti) sono Riferibili con ID,
Posso dargli un Nome
■ Memorizzabile se Possono Essere Associate a locaz.

Le **MEM.** Serve per Catturare le azioni **IRREVERSIBILI** generate dal PROGRAMMA

L'Ambiente Cattura Solo Trasformazioni **REVERSIBILI**

$x := x + 1$ } Ho Neutralizzato la Prima Operazione,
 $x := x - 1$ } facendo 2 Operazioni DIVERSE e SUCCESSIVE

✗ SOSPESA

→ DELIMITATORE DELL' AMBIENTE
{ int **x** CREAZIONE DI **x**
:
}

Tutte le Modifiche ad **x** Sono Annullate e **x** Stessa è ELIMINATA (Quella Interna) e Viene Recuperato **x**

MEMORIA:

MEM, insieme di Memorie Sono ASSOCIAZIONI Tra Locazione e Valore MEMORIZZABILE

Loc, insieme di Locazioni

MVal, insieme di Valori Memorizzabili

$$\delta \in \text{MEM} \quad \delta : \text{Loc} \rightarrow \text{Val}$$

$$L \subseteq_f \text{Loc} \quad \forall \delta \in L. \quad \delta : l \mapsto v \in \text{MVal}$$

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

$\text{MEM} = \bigcup_{L \subseteq_f \text{Loc}} \text{MEM}_L$ Insieme di Associazioni da un Qualunque SOTTOINSIEME FINITO

Insieme finito di Locazioni ad un Insieme di Valori

DVal = Eval \cup Loc che Nel Nostro Lingueggio:

$$N \cup B \cup Loc$$

$$\text{e } \text{MVal} = N \cup B$$

AGGIORNAMENTO MEM

$\delta[\delta']$

$\delta, \delta' \in M_{\text{EM}}$

$\delta : L$

$\delta' : L'$

dominio di δ

dominio di δ'

$\delta'' = \delta[\delta'] \in M_{\text{EM}}$

$$\forall l \in L \cup L'. \delta''(l) = \begin{cases} \delta'(l) & \text{con } l \in L' \\ \delta(l) & \text{con } l \in L \setminus L' \end{cases}$$

Aggiungiamo Alle dichiarazioni, la DICHIARAZIONE DI ID VARIABILE

Tipi Prima = {int, Bool}

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

tutte Quelle già
Viste Prima

Tipi Ora = {int, Bool, int_{loc}, Bool_{loc}}

Sono MODIFICABILI

Devo CREARE un TIPO 'locazione' = τ_{loc} per dire che è
una LOCAZIONE di TIPO τ

int_{loc}, Bool_{loc} \Rightarrow Locazione per INTERI, BOOLEANI

Rimane Tipo τ Quando Mettiamo CONST

SEMANTICA STATICÀ: dichiarazioni

Associa un Ambiente Statico

$$\Delta I \left(\text{var } x : \tau = e \right) = \{x\} \quad FI \left(\text{var } x : \tau = e \right) = FI(e)$$

Aggiungo Queste 2 Definizioni a Quelle già date

$$\frac{\Delta \vdash e : \tau}{\Delta \vdash x : \tau = e : [x \mapsto \tau_{loc}]}$$

SEMANTICA STATICÀ: Espressioni

$$\Delta \vdash I : \tau \text{ se } \Delta(I) \in \{\tau \text{ oppure } \tau_{loc}\}$$

SEMANTICA DINAMICA: Cambia per Aggiunta delle mem.

$$T : E \times \text{MEM} \rightarrow \text{Val} \times \text{MEM}$$

\downarrow
NvB

Dove $T = V$ [Terminali Rimangono VALORI]

$$\beta \vdash \langle e, d \rangle \rightarrow \langle e', d' \rangle$$

La SEMANTICA DINAMICA delle EXPR. Accede ALLA MEMORIA

in Cui Valutare l'ESPRESSIONE (Ma NON la modifica)

Nelle Dichiarazioni:

$$\text{T}: D \times \text{Mem} \rightarrow \underbrace{\text{Env} \times \text{Mem}}_{\text{T}}$$

$\vdash < d, \delta > \rightarrow < d', \delta' >$ dove δ' può essere $\neq \delta$

REGOLE DA AGGIUNGERE ALLE Expr.:

$$\vdash < x, \delta > \longrightarrow < n, \delta > \quad \begin{array}{l} \text{se } \delta(x) = n \text{ oppure } \delta(x) = l \\ \text{e } \delta(l) = n \end{array}$$

x
COSTANTE → n
VARIABILE → l → n ∈ Numb

REGOLE DA AGGIUNGERE ALLE DICHIARAZIONI:

$$\frac{\vdash < e, \delta > \longrightarrow^* < K, \delta >}{\vdash < \text{Var } x : r = e, \delta > \longrightarrow < [x : r = \underline{K}], \delta[l \mapsto \underline{K}] \rangle}$$

LE LOC
(nuova Locazione)

Genero una MODIFICA IRREVERSIBILE con la MEMORIA e Questa è l'unica Regola che Può Modificare La MEMORIA (che è un SIDE EFFECT, visto che Op. su MEM dovrebbero Essere

Solo dei Comandi)

Con Queste Inizializzazione delle Variabili Modifichiamo la Memoria

VALUTAZIONE DI UN Expr. CON ID VARIABILI:

Eval: Expr \times Mem \rightarrow Val

$$\text{Eval}(e, \delta) = (K, \delta) \text{ sse } \vdash \langle e, \delta \rangle \xrightarrow{*} \langle K, \delta \rangle$$

EQUIVALENZA: $\equiv \subseteq \text{Expr} \times \text{Expr}$

$\forall e_0, e_1 \in \text{Expr. } e_0 \equiv e_1 \text{ se } \forall \delta. \text{Eval}(e_0, \delta) = \text{Eval}(e_1, \delta)$

ESEMPIO: $(2 + x - x)$ o Anche $\left(\frac{2x}{x}\right)$ Sono

INARIANTI ed INDIPENDENTI dal Valore Assegnato da x

ELABORAZIONE DI DICHIARAZIONI CON ID VAR:

Elab: Dec \times Mem \rightarrow Env \times Mem

$$\text{Elab}(d, \delta) = (\rho, \delta') \text{ sse } \vdash \langle d, \delta \rangle \xrightarrow{*} \langle \rho, \delta' \rangle$$

EQUIVALENZA: deve Tener Conto dei SIDE EFFECTS (anche Sulla Memoria GENERATA)

$\forall d_1, d_2 \in \text{Dec } d_1 \equiv d_2 \text{ sse } \forall \delta \in \text{Mem } \text{Elab}(d_1, \delta) = \text{Elab}(d_2, \delta)$

Comandi:

Strumenti per **MODIFICARE LA MEMORIA** (costrutti dedicati)

Nuova CATEGORIA SINTATTICA, mi fanno Cambiare lo STATO DELLA MACCHINA.

Mi permettono di **DENOTARE** richieste di **MODIFICA** alle **MEM.**

Comandi **Vengono Eseguiti** per OTTENERE la TRASFORMAZIONE della MEMORIA DERIVATA

fino ad ora Tutto è DECIDIBILE (Ma aggiungeremo il WHILE, con Problema della non TERMINAZIONE)

ASSEGNAZIONE → Base di Modifica della Memoria.

COMPOSIZIONE SEQ. → Come Componiamo Pezzi del Progr.

CONTROLLI FLUSSO ESECUZIONE → Condizionale / Iterativo

SINTASSI:

$C \rightarrow \text{Skip} \mid x := e \mid C; C \mid \text{WHILE } B \text{ do } C \mid \text{if } B \text{ Then } C \text{ Else } C$

Elemento Neutro

Assegnamento (x è ID e EXPR)

Composizione Sequenziale ⇒ Tutto il Primo Comando e Poi il Secondo

Ripete C fino a che B è Valutata Vero

Scelta Condizionale \Rightarrow Se B è Vero Esegua C Altrimenti si Esegue il Ramo ELSE.

Assegnamento Introduce il Concetto di Variabile che Consiste in un ELEMENTO COMPLESSO in un PL

Elementi Che Caratterizzano la Variabile:

■ Nome/ID

■ Tipo \rightarrow Valori/Operazioni che Caratterizzano la Var.

■ Valore \rightarrow Denotabile/Locazione è un Val Riferibile Mediante l'occorrenza del Nome

■ Indirizzo \rightarrow A Lvl. fisico è quello che Abbiamo assegnato con la Locazione

■ SCOPE \rightarrow Visibilità nell' Ambiente / Reggio d'Azione ovvero L'Ambiente di Riferimento

■ TEMPO DI VITA \rightarrow Misura in Termini Temporali dell' Esistenza della Variabile (DIPENDE ANCHE DAL LINGUAGGIO)

Mi Permettono

ANCHE DAL LINGUAGGIO)

DEALLOCAZIONE e Quindi diventa DINAMICO

È Quindi Un Insieme di Elementi con Significato Complesso

SINTASSI DEL COSTRUTTO \Rightarrow := che non si CONFERMA
con = Usato per l'UGUAGLIANZA