Sistema di Tipi Y di Small21

- **Sintassi Concreta**
  Type ::= Simple | void | Simple [Num]
  Simple ::= int | bool

- **Sintassi Astratta ed Espressioni di Tipo**
  Type ::= [int] | [bool] | [lab] | [void] | [arr] Type Num
  | [mut] Type | [abs] Type TypeSeq | [terr]
  | [unit] | [type]

- **Regole: Una Regola di $\mathtt{Y}_{\mathtt{DCL}^{\mathtt{Small21}}}$**

$$\frac{\langle \mathtt{e}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{t}', \mathtt{Y}_\rho) \qquad \mathtt{t}' \neq [\mathtt{terr}] \qquad \mathtt{t}' = \mathtt{t}}{\langle [\mathtt{const}]\ \mathtt{t}\ \mathtt{I}\ \mathtt{e}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{Void}, [\mathtt{I}/\mathtt{t}] \circ \mathtt{Y}_\rho)}$$

Sono inferenze di un **Sistema Deduttivo** che associa ad ogni struttura del programma, s, con tipi degli identificatori definiti da $\mathtt{Y}_\rho$, una coppia $(\mathtt{t}', \mathtt{Y}'_\rho)$. Sopra, nella premessa della regola, nella formula:

$$\langle \mathtt{e}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{t}', \mathtt{Y}_\rho)$$

vediamo ciò. In particolare, la regola è applicabile quando il sistema ha dedotto che l'espressione e, dove i suoi identificatori liberi hanno tipi tutti legati in $\mathtt{Y}_\rho$, abbia tipo $\mathtt{t}'$ (i.e. $\mathtt{t}' = \mathtt{t}$). Ovvero:

$$\mathtt{Y}_\rho \vdash \mathtt{e} : \mathtt{t}.$$

Nella conclusione della regola sopra, la struttura di programma s è:

$$[\mathtt{const}]\ \mathtt{t}\ \mathtt{I}\ \mathtt{e}.$$

Quando la regola è applicabile allora il sistema è in grado di inferire

$$\langle \mathtt{s}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{Void}, [\mathtt{I}/\mathtt{t}] \circ \mathtt{Y}_\rho)$$

ovvero il sistema deduce che la struttura s è/introduce una corretta dichiarazione che estende in modo consistente $\mathtt{Y}_\rho$ con il binding $[\mathtt{I}/\mathtt{t}]$. Ovvero:

$$\mathtt{Y}_\rho, \mathtt{I} : \mathtt{t} \vdash \mathtt{s} : \mathtt{Void}.$$

Sistema di Tipi: Regole, Strutture e Proprietà di Y

$$\frac{\langle \mathtt{e}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{t}', \mathtt{Y}_\rho) \qquad \mathtt{t}' \neq [\mathtt{terr}] \qquad \mathtt{t}' = \mathtt{t}}{\langle [\mathtt{const}]\ \mathtt{t}\ \mathtt{I}\ \mathtt{e}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{Void}, [\mathtt{I}/\mathtt{t}] \circ \mathtt{Y}_\rho)}$$

- **Regole.** Sono inferenze di un sistema deduttivo che associano ad ogni struttura del programma, s, con tipi degli identificatori definiti da $\mathtt{Y}_\rho$, una coppia $(\mathtt{t}', \mathtt{Y}'_\rho)$.

  Sopra, in premessa $\langle \mathtt{e}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{t}', \mathtt{Y}_\rho)$ vediamo ciò. In particolare, il sistema ha dedotto che l'espressione e, dove i suoi identificatori liberi siano tutti legati in $\mathtt{Y}_\rho$, abbia tipo $\mathtt{t}' = \mathtt{t}$. Ovvero: $\mathtt{Y}_\rho \vdash \mathtt{e} : \mathtt{t}$.
  Sopra, nella conclusione, la struttura di programma s è: $[\mathtt{const}]\ \mathtt{t}\ \mathtt{I}\ \mathtt{e}$. Quando il sistema è in grado di inferire $\langle \mathtt{s}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{Void}, [\mathtt{I}/\mathtt{t}] \circ \mathtt{Y}_\rho)$ allora il sistema ha dedotto che la struttura s introduce una corretta dichiarazione che estende in modo consistente $\mathtt{Y}_\rho$ con il binding $[\mathtt{I}/\mathtt{t}]$. Ovvero: $\mathtt{Y}_\rho, \mathtt{I} : \mathtt{t} \vdash \mathtt{s} : \mathtt{Void}$.

- **Identificatori.** Tutti gli identificatori introdotti nel programma hanno un tipo t (definito da un'espressione di tipo del linguaggio). Questo tipo è associato all'identificatore nell'ambiente $\mathtt{Y}_\rho$ come il binding dell'identificatore in $\mathtt{Y}_\rho$.

- **Struttura $\mathtt{Y}_\rho$.** È una struttura di ambiente con le stesse operazioni viste per gli ambienti di denotazioni. In un Linguaggio a blocchi, l'applicazione di Y in differenti blocchi di un programma richiede che $\mathtt{Y}_\rho$ sia uno stack $> \rho :: \mathtt{Y}'_\rho]$ di ambienti senza ripetizioni con ρ ambiente del corrente blocco e $\mathtt{Y}'_\rho$ lo stack di ambienti (uno per ogni blocco attraversato fino al corrente) creati staticamente dal sistema Y.

- **Completezza** di Y richiede che il sistema sia dotato di un insieme di regole tali che: L'applicazione di Y ad ogni programma (sintatticamente corretto e legale) P, nell'ambiente $\mathtt{Y}_\mathtt{L}$, contenente i binding per tutti e soli gli identificatori di primitive del Linguaggio, sia in grado di fornire una e una sola derivazione di tipi ovvero un albero dove ogni nodo è la conclusione di una regola e i figli sono le premessa di tale regola sotto una stessa istanza di simboli della regola con strutture di P. L'albero così ottenuto ha:

  - radice: $\langle \mathtt{P}, \mathtt{Y}_\mathtt{L} \rangle \rightarrow_{\mathtt{Y}} (\mathtt{Void}, \mathtt{Y}_\mathtt{L})$. Ovvero: $\mathtt{Y}_\mathtt{L} \vdash \mathtt{P} : \mathtt{Void}$
  - discendenti: Per ogni termine s contenuto in P, $\langle \mathtt{s}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} (\mathtt{t}', \mathtt{Y}'_\rho)$. Ovvero: $\mathtt{Y}_\mathtt{L}, \mathtt{Y}_\rho \vdash \mathtt{s} : \mathtt{t}$

- **Ben tipato** in Y, è ogni termine (costrutto) s di ogni programma P (sintatticamente corretto e legale) che abbia una derivazione di tipi con radice come sopra e con discendenti come sopra e tali che $\mathtt{t} \neq [\mathtt{terr}]$.

Sistema di Tipi: Regole, Strutture e Proprietà di Y - 2

$$\frac{\Gamma_1 \quad \ldots \quad \Gamma_n}{\langle s, Y_\rho \rangle \rightarrow_Y (t, Y'_\rho)}$$

- **Stato di Stuck** in un Linguaggio con un sistema di tipi Y, è ogni stato inconsistente rispetto a Y. Definiamo inconsistente ogni stato che contiene:
  - **binding** tra un identificatore di tipo `t1` e un valore (denotabile o memorizzabile) di tipo `t2`, con `t1` $\neq$ `t2`;
  - **valore atteso**, in qualche componente di un qualche AR (inclusi buffer di I/O), di tipo `t1` e un valore calcolato di tipo `t2`, con `t1` $\neq$ `t2`

  Ogni Linguaggio tipato ha una propria struttura di stato che include sempre l'ambiente dei bindings degli identificatori (introdotti nel programma), ρ, lo stack di AR e, secondo i casi, una (o più per heap, per variabili statiche) memoria μ per valori modificabili e ogni altra struttura (buffer di I/O, tabella delle classi, tabella dei moduli,..) che debba far parte dello stato. Il sistema di tipi provvede esplicitamente a definire la consistenza per la struttura di stato dello specifico Linguaggio tipato.
- **Stato NonStuck** in un Linguaggio con un sistema di tipi Y, è ogni stato che soddisfa la consistenza richiesta da Y.
- **Correttezza (Safety)** di Y richiede che esso sia definito in accordo alla semantica del Linguaggio garantendo le due proprietà sotto:
  - **Progress**: Ogni termine ben tipato `s` (avente $Y_\rho \vdash s : t$) o è esso stesso un valore di tipo `t`[1] che occorre in uno stato NonStuck, oppure in accordo alla semantica la computazione contiene un successivo stato;
  - **Preservation**: Se `s` nel successivo stato è stato rimpiazzato con s' allora anche s' è ben tipato ed anche per esso vale $Y_\rho \vdash s' : t$
- **Theorem Completezza+Correttezza (locale)** Ogni Linguaggio tipato con sistema di tipi Completo e Safe è tale che tutti i suoi programmi ben tipati hanno computazioni prive di stati di Stuck.

[1] Ad ogni tipo è associato un insieme di valori non vuoto. Il tipo Void contiene il solo valore: `void`

Sintassi Astratta di Small21

```
Type ::= [int] | [bool] | [lab] | [void]
       | [mut] Type | [arr] Type Num | [terr]
       | [abs] TypeSeq | [unit] | [type]

Dcl ::= [var] Type Ide Exp | [const] Type Ide Exp
      | [array] Type Ide | Dcl [seqD] Dcl
      | [pcd] Type Ide FPars BlockC | [emptyD]

Exp ::= DExp
      | [num] Num | Exp [+] Exp
      | Exp [==] Exp
      | [true] | [false] | Exp [or] Exp
      | Exp [>] Exp | Exp [<] Exp
      | [apply] Ide APars | [emptyE]
DExp ::= [val] Ide | DExp [↑] Exp | Ide [↑1] Exp

Cmd ::= Lab [:] Stm | Stm

Stm ::= DExp [=] Exp | [ifE] Exp Stm Stm | [goto] Lab
      | Stm [seqS] Stm | BlockS | [emptyS]
      | [call] Ide APars | [return] Exp

Prog ::= [prog] Ide Block
```

Sintassi Astratta di Small21 - 2

```
TypeSeq ::= Type [::] TypeSeq | Type

Num                              -- token del lessico degli interi
Ide                              -- token del lessico degli identificatori
Lab                              -- token del lessico delle labels di cmd

FPars ::= [fp] PPF Type Ide | [emptyFP]

PPF ::= [value] | [ref]          -- Parameter passing Form

BlockE ::= [blockE] Dcl Exp
BlockC ::= [blockC] Dcl Stm      -- in-line block
Block ::= [block] Dcl CmdSeq
CmdSeq ::= [seqC] Cmd CmdSeq | EC

APars ::= [ap] Exp | [emptyAP]
```

Osservazioni

- [mut]: Costruttore per Tipi di valori mutable. Ad es.: [mut][int], [mut]([arr]([int],30)), [arr]([mut][bool],15).
- ↑: Costruttore di termine di accesso a componente array. Ad es.: `x` ↑ 3 corrisponde, in C, all'espressione `x[3]` quando `x` sia una variabile di tipo [mut]([arr]([mut][int],30)).
- SeqTypes: Il tipo più a sinistra indica il tipo immagine i successivi, quando presenti, i tipi degli argomenti della astrazione funzionale, procedurale o di operatore primitivo.
- FPars e APars: Parametri per astrazioni al momento con al più un solo parametro e solo 2 forme di trasmissione.

Sistema Y: Regole per DCL

$$[\text{Y1}]\ \frac{\langle \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} (\texttt{t}', \texttt{Y}_\rho) \qquad \texttt{t} \in \texttt{Simple} \qquad \texttt{t}' = \texttt{t} \qquad \texttt{Y}_\rho = {>}\rho :: \texttt{Y}'_\rho] \qquad \rho(\texttt{I}) = \bot \qquad {>}[\texttt{I}/\texttt{t}] \circ \rho :: \texttt{Y}'_\rho] = \texttt{Y}''_\rho}{\langle [\texttt{const}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{void}], \texttt{Y}''_\rho)}$$

$$[\text{Y2}]\ \frac{\langle \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} (\texttt{t}', \texttt{Y}_\rho) \qquad \texttt{t} \in \texttt{Simple} \qquad (\texttt{t}' = \texttt{t}\ \texttt{or}\ \texttt{t}' = [\texttt{unit}]) \qquad \texttt{Y}_\rho|_0(\texttt{I}) = \bot}{\langle [\texttt{var}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{void}], [\texttt{I}/[\texttt{mut}]\ \texttt{t}] \otimes \texttt{Y}_\rho)}$$

Gestione Errori di Tipo:

$$[\text{E1}]\ \frac{\texttt{t} \notin \texttt{Simple}}{\langle [\texttt{const}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{terr}], \texttt{Y}_\rho)} \qquad [\text{E2}]\ \frac{\langle \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} (\texttt{t}', \texttt{Y}_\rho) \qquad \texttt{t}' \neq \texttt{t}}{\langle [\texttt{const}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{terr}], \texttt{Y}_\rho)}$$

$$[\text{E3}]\ \frac{\texttt{Y}_\rho|_0(\texttt{I}) \neq \bot}{\langle [\texttt{const}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{terr}], \texttt{Y}_\rho)}$$

$$[\text{E4}]\ \frac{\texttt{t} \notin \texttt{Simple}}{\langle [\texttt{var}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{terr}], \texttt{Y}_\rho)} \qquad [\text{E5}]\ \frac{\langle \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} (\texttt{t}', \texttt{Y}_\rho) \qquad \texttt{t}' \neq \texttt{t}}{\langle [\texttt{var}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{terr}], \texttt{Y}_\rho)}$$

$$[\text{E6}]\ \frac{\texttt{Y}_\rho|_0(\texttt{I}) \neq \bot}{\langle [\texttt{var}]\ \texttt{t}\ \texttt{I}\ \texttt{e}, \texttt{Y}_\rho\rangle \rightarrow_{\texttt{Y}} ([\texttt{terr}], \texttt{Y}_\rho)}$$

Notazione: Simboli e Strutture

- ${>}\rho :: \texttt{Y}_\rho]$, stack non vuoto di ambienti di legami `[I/t]`, avente ρ come top e $\texttt{Y}_\rho$ come stack residuo.
- $\texttt{Y}_\rho|_{\texttt{n}}$, n-esimo (`n>=0`) ambiente precedente il top dello stack $\texttt{Y}_\rho$.
- $[\texttt{I}/\texttt{t}] \otimes \texttt{Y}_\rho$, aggiunta del binding `[I/t]` nel top dello stack $\texttt{Y}_\rho$.
- `N`, Intero; • `I` (anche con apici), Identificatore; • `t` (anche con apici), Tipo; • `e`, espressione.
- $\texttt{Simple} = \{[\texttt{int}], [\texttt{bool}]\}$

Sistema Y: Regole per DCL - 2

$$[\text{Y3}]\ \frac{\texttt{t} \in \texttt{Simple} \qquad \texttt{N} > 0 \qquad \texttt{Y}_\rho|_0(\texttt{I}) = \bot}{\langle [\texttt{array}]\,\texttt{t}\,\texttt{I}\,\texttt{N}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{void}], [\texttt{I}/[\texttt{arr}]([\texttt{mut}]\ \texttt{t})\ \texttt{N}] \otimes \texttt{Y}_\rho)}\ **$$

$$[\text{Y4}]\ \frac{\langle \texttt{d1}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{void}], \texttt{Y1}_\rho) \quad \langle \texttt{d2}, \texttt{Y1}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{void}], \texttt{Y2}_\rho)}{\langle \texttt{d1}\,[\texttt{seqD}]\,\texttt{d2}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{void}], \texttt{Y2}_\rho)}\ **$$

Gestione Errori di Tipo:

$$[\text{E7}]\ \frac{\texttt{t} \notin \texttt{Simple}}{\langle [\texttt{array}]\,\texttt{t}\,\texttt{I}\,\texttt{N}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y}_\rho)} \qquad [\text{E8}]\ \frac{\texttt{N} <= 0}{\langle [\texttt{array}]\,\texttt{t}\,\texttt{I}\,\texttt{N}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y}_\rho)}$$

$$[\text{E9}]\ \frac{\texttt{Y}_\rho|_0(\texttt{I}) \neq \bot}{\langle [\texttt{array}]\,\texttt{t}\,\texttt{I}\,\texttt{N}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y}_\rho)}$$

$$[\text{E10}]\ \frac{\langle \texttt{d1}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y1}_\rho)}{\langle \texttt{d1}\,[\texttt{seqD}]\,\texttt{d2}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y}_\rho)} \qquad [\text{E11}]\ \frac{\langle \texttt{d1}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{void}], \texttt{Y1}_\rho) \quad \langle \texttt{d2}, \texttt{Y1}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y2}_\rho)}{\langle \texttt{d1}\,[\texttt{seqD}]\,\texttt{d2}, \texttt{Y}_\rho\rangle \rightarrow_\texttt{Y} ([\texttt{terr}], \texttt{Y1}_\rho)}$$

Notazione: Simboli e Strutture

- [], ambiente vuoto
- $>\rho :: \texttt{Y}_\rho]$, stack non vuoto di ambienti di legami `[I/t]`, avente ρ come top e $\texttt{Y}_\rho$ come stack residuo.
- $\texttt{Y}_\rho|_\texttt{n}$, n-esimo (`n>=0`) ambiente precedente il top dello stack $\texttt{Y}_\rho$.
- $[\texttt{I}/\texttt{t}] \otimes \texttt{Y}_\rho$, aggiunta del binding `[I/t]` nel top dello stack $\texttt{Y}_\rho$.

Sistema Y: Regole per DCL - 3

$$[\text{Y5})\ \frac{\mathtt{t} \in \mathtt{Simple} \cup \{[\mathtt{void}]\} \qquad \mathtt{F} = [\mathtt{fp}]\,\mathtt{p}\,\mathtt{t}'\,\mathtt{I}' \qquad \mathtt{t}' \in \mathtt{Simple} \qquad \mathtt{Y}_\rho|_0(\mathtt{I}) = \bot \qquad > [\mathtt{I}'/\mathtt{t}'] \circ [] :: \mathtt{Y}_\rho] = \mathtt{Y}'_\rho \qquad \langle \mathtt{Bc}, \mathtt{Y}'_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{void}], \mathtt{Y}''_\rho)}{\langle [\mathtt{pcd}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{F}\,\mathtt{Bc}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{void}], [\mathtt{I}/[\mathtt{abs}]\,\mathtt{t}\,[::]\,\mathtt{t}'] \otimes \mathtt{Y}_\rho)}\ **$$

$$[\text{Y6})\ \frac{\mathtt{t} \in \mathtt{Simple} \cup \{[\mathtt{void}]\} \qquad \mathtt{F} = [\mathtt{emptyFP}] \qquad \mathtt{Y}_\rho|_0(\mathtt{I}) = \bot \qquad > [] :: \mathtt{Y}_\rho] = \mathtt{Y}'_\rho \qquad \langle \mathtt{Bc}, \mathtt{Y}'_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{void}], \mathtt{Y}''_\rho)}{\langle [\mathtt{pcd}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{F}\,\mathtt{Bc}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{void}], [\mathtt{I}/[\mathtt{abs}]\,\mathtt{t}] \otimes \mathtt{Y}_\rho)}\ **$$

Gestione Errori di Tipo:

$$[\text{E12}]\ \frac{\mathtt{t} \notin \mathtt{Simple} \cup \{[\mathtt{void}]\}}{\langle [\mathtt{pcd}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{F}\,\mathtt{Bc}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{terr}], \mathtt{Y}_\rho)}$$

$$[\text{E13}]\ \frac{\mathtt{F} = [\mathtt{fp}]\,\mathtt{p}\,\mathtt{t}'\,\mathtt{I}' \qquad \mathtt{t}' \notin \mathtt{Simple}}{\langle [\mathtt{pcd}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{F}\,\mathtt{Bc}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{terr}], \mathtt{Y}_\rho)}$$

$$[\text{E14}]\ \frac{\mathtt{Y}_\rho|_0(\mathtt{I}) \neq \bot}{\langle [\mathtt{pcd}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{F}\,\mathtt{Bc}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{terr}], \mathtt{Y}_\rho)}$$

$$[\text{E15}]\ \frac{\ldots\ldots \qquad \langle \mathtt{Bc}, > \rho :: \mathtt{Y}_\rho] \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{terr}], \mathtt{Y}'_\rho)}{\langle [\mathtt{pcd}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{F}\,\mathtt{Bc}, \mathtt{Y}_\rho \rangle \rightarrow_{\mathtt{Y}} ([\mathtt{terr}], \mathtt{Y}_\rho)}$$

Notazione: Simboli e Strutture

- [], ambiente vuoto
- $> \rho :: \mathtt{Y}_\rho]$, stack non vuoto di ambienti di legami [I/t], avente ρ come top e $\mathtt{Y}_\rho$ come stack residuo.
- $\mathtt{Y}_\rho|_{\mathtt{n}}$, n-esimo (n>=0) ambiente precedente il top dello stack $\mathtt{Y}_\rho$.
- $[\mathtt{I}/\mathtt{t}] \otimes \mathtt{Y}_\rho$, aggiunta del binding [I/t] nel top dello stack $\mathtt{Y}_\rho$.
- $[\mathtt{abs}]\,\mathtt{t}\,[::]\,\mathtt{t}'$, $[\mathtt{abs}]\,\mathtt{t}$, Tipi per procedura e funzione;
- F, parametro formale;
- Bc blocco di statements;

AM21: Il Modello Astratto - Le Operazioni

- **Relazione tra Stato, Semantica ed Operazioni**.
  - Lo Stato esprime tutte e sole le configurazioni della AM21
  - Computazione di un programma Small21 P a partire da uno stato σ è la Sequenza di Stati che la Semantica associa a P valutato nello stato iniziale σ.
  - La Semantica è espressa nelle sole operazioni di AM21.

- **Funzioni Semantiche e Sistema di Inferenza associato**.
  - **SistemaY** : $\texttt{costrutto} : \texttt{AST} \rightarrow \texttt{contesto} : \texttt{Y}_\rho \rightarrow \texttt{tipo} : \texttt{Type} * \texttt{contestoEsteso} : \texttt{Y}_\rho$
    $(\forall \texttt{d}, \sigma) \quad \textbf{dclSem}(\texttt{d}, \sigma) = (\texttt{t}, \sigma') \quad \textbf{iff} \quad \langle \texttt{d}, \sigma \rangle_{\texttt{Y}} \rightarrow (\texttt{t}, \sigma') \in \textbf{Sem}_{\textbf{DCL}}$
  - **dclSem** : $\texttt{Dcl} \rightarrow \texttt{State} \rightarrow \texttt{Type} * \texttt{State}$
    $(\forall \texttt{d}, \sigma) \quad \textbf{dclSem}(\texttt{d}, \sigma) = (\texttt{t}, \sigma') \quad \textbf{iff} \quad \langle \texttt{d}, \sigma \rangle_{\texttt{DCL}} \rightarrow (\texttt{t}, \sigma') \in \textbf{Sem}_{\textbf{DCL}}$

- **Stato: Struttura**
  - (Δ, μ) : Stato = Stack di AR, Δ + Store, μ.

- **Stack di AR: Struttura**
  - $>\texttt{ar}_{\texttt{top}} \ldots \texttt{ar}_1]$, struttura LIFO, con $\texttt{ar}_{\texttt{top}}$ ultimo inserito e $\texttt{top} \geq 0$

- **Activation Record, AR: Struttura**.
  - $\{\texttt{head}, \texttt{chain}, \texttt{frame}, \texttt{cont}, \texttt{val}\}$: AR a 5 componenti

- **Frame: Struttura**.
  - $[\texttt{Ide}_1/\texttt{Den}_1, \ldots, \texttt{Ide}_k/\texttt{Den}_k]$ frame di un AR mantiene i binding del blocco attraversato

- **Store: Struttura**
  - $[\texttt{loc}_1 \leftarrow \texttt{Mv}_1, \ldots, \texttt{loc}_k \leftarrow \texttt{Mv}_k]$ store di uno stato mantiene il valore dei modificabili della computazione

Dichiarazioni Small21: Le Transizioni $\mathrm{SEM}_{\mathrm{DCL}}$

Il Sistema di regole $\mathtt{Sem}_{\mathrm{DCL}}$ definisce il comportamento delle dichiarazioni durante la computazione dei Programmi Small21 sulla Macchina Astratta AM21.
Controllo dei Tipi Dinamico: Il Sistema $\mathtt{Sem}_{\mathrm{DCL}}$ è Integrato con il Sistema Y (slides sopra, per le dichiarazioni).

$\mathtt{Dcl} ::= \mathtt{[const]\ Type\ Ide\ Exp} \mid \mathtt{[var]\ Type\ Ide\ Exp}$
$\quad \mid \mathtt{[varN]\ Type\ Ide} \mid \mathtt{[array]\ Type\ Ide\ Num}$

$$[\text{D1}]\ \frac{\begin{array}{c}\langle \mathtt{e}, (\rho, \mu)\rangle \to \langle \mathtt{t_e}, \mathtt{v}, (\rho, \mu_{\mathtt{e}})\rangle \\ \mathtt{t} \in \mathtt{Simple} \qquad \mathtt{t} = \mathtt{t_e} \\ \Delta|_0(\mathtt{I}) = \bot \qquad [\mathtt{I}/(\mathtt{t}, \mathtt{v})] \otimes \Delta = \Delta_{\mathtt{I}}\end{array}}{\langle [\mathtt{const}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{e}, (\Delta, \mu)\rangle \to ([\mathtt{void}], (\Delta_{\mathtt{I}}, \mu_{\mathtt{e}}))}$$

$$[\text{D2}]\ \frac{}{\langle [\mathtt{var}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{e}, (\Delta, \mu)\rangle \to (\qquad , \qquad)}$$

$$[\text{D3}]\ \frac{}{\langle [\mathtt{array}]\,\mathtt{t}\,\mathtt{I}\,\mathtt{N}, (\rho, \mu)\rangle \to (\qquad , \qquad)}$$

Notazione, Osservazioni

- [xxx] è un costruttore di AST (i.e, Albero di Sintassi Astratta). I rimanenti identificatori (inessenziale se minuscoli o Maiuscoli o se con indici) che occorrono in una regola sono nomi di variabile quantificate universalmente se introdotte, i.e. bound, nel termine sinistro della conclusione, esistenzialmente se introdotte nel termine destro di una premessa o di una uguaglianza.
- $\mathtt{Simple} = \{[\mathtt{Int}], [\mathtt{Bool}]\}$

Dichiarazioni Small21: Le Transizioni $\mathtt{SEM}_{\mathtt{DCL}}$ — 2

$$[\text{D4}]\ \frac{}{\langle \mathtt{d1[seqD]d2}, (\Delta, \mu)\rangle \rightarrow (\qquad\quad , \qquad\quad)}$$

$$[\text{D5}]\ \frac{}{\langle [\mathtt{pcd}]\qquad\quad , (\Delta, \mu)\rangle \rightarrow (\qquad , \qquad)}$$

$$[\text{D6}]\ \frac{}{\langle [\mathtt{pcd}]\qquad\quad , (\Delta, \mu)\rangle \rightarrow (\qquad , \qquad)}$$

Notazione, Osservazioni

- [xxx] è un costruttore di AST (i.e, Albero di Sintassi Astratta). I rimanenti identificatori (inessenziale se minuscoli o Maiuscoli o se con indici) che occorrono in una regola sono nomi di variabile quantificate universalmente se introdotte, i.e. bound, nel termine sinistro della conclusione, esistenzialmente se introdotte nel termine destro di una premessa o di una uguaglianza.
- $\mathtt{Simple} = \{[\mathtt{Int}], [\mathtt{Bool}]\}$
- Errori Dinamici ed Errori Statici.