Corso: Fondamenti, Linguaggi e Traduttori

Paola Giannini

Analisi Semantica (typechecking di **ac**)

Analisi Semantica

- Tutti quegli aspetti dell'analisi di un programma che dipendono dalla specifica del linguaggio e che non possono essere risolti nella fase di analisi sintattica.
- Il controllo di tipo, type checking, è l'analisi principale della fase di analisi semantica di un linguaggio tipato staticamente, ma i linguaggi moderni (Java, C#, ecc..) specificano anche altre analisi, ad esempio,
  - l'analisi di raggiungibilità di istruzioni cioè individuare codice che NON verrà mai eseguito,
  - l'analisi di inizializzazione di variabili, ecc.
- Per linguaggi molto semplici, l'analisi di tipo si può effettuare durante il parsing, calcolando il tipo delle componenti e mantenendo la symbol table come struttura globale
- Per linguaggi più complessi questa analisi viene fatta visitando (anche più volte) l'AST del programma.
- Noi effettueremo il controllo di tipi con una visita del AST del programma.

UPO
UNIVERSITÀ DEL PIEMONTE ORIENTALE

Controllo di tipo per le componenti di un linguaggio

Concettualmente ci sono 3 componenti distinte in un linguaggio di programmazione:

- Le dichiarazioni che introducono un "binding" fra un nome e una sua definizione (e il suo tipo).
- Le istruzioni che hanno effetti (non c'e' un valore ritornato), ma hanno delle restrizioni di tipo, ad esempio l'assegnamento richiede che il tipo dell'espressione a destra sia compatibile con quello della variabile a sinistra.
- Le espressioni che computano valori e il cui tipo è il tipo del valore ritornato.

In molti linguaggi alcune istruzioni possono essere usate anche come espressioni, ad esempio l'assegnamento ritorna il valore dell'espressione a destra di "=" in questo caso hanno il tipo del valore ritornato.

UPO
UNIVERSITÀ DEL PIEMONTE ORIENTALE

Analisi delle dichiarazioni (di **ac**)

L' analisi delle dichiarazioni (abbiamo un solo tipo di dichiarazione!)

- inserisce l'identificatore nella Symbol Table con associata una entry che contiene i suoi attributi. Per il momento l'unico attributo è il tipo, ma durante la generazione del codice aggiungeremo il registro associato all'identificatore.
- deve segnalare un errore se un identificatore è già definito

UPO
UNIVERSITÀ DEL PIEMONTE ORIENTALE

Analisi delle istruzioni e compatibilità di tipo (di **ac**)

Ci sono 2 istruzioni: stampa e assegnamento

- La stampa è corretta se la variabile è definita (per questo si deve controllare che l'identificatore nel `NodeId` sia definito nella symbol table).
- L'assegnamento è corretto se
  1. la variabile a sinistra è definita (di nuovo si deve controllare che l'identificatore nel `NodeId` sia definito nella symbol table)
  2. l'espressione a destra ha un tipo che è compatibile con quello della variabile a sinistra altrimenti si deve segnalare un errore
  3. se i tipi sono compatibili ma diversi si deve inserire una conversione di tipo

Compatibilità fra `INT` e `FLOAT` (i nostri unici tipi)

- Come in Java `INT` è compatibile con`FLOAT` (cioè un `INT` si può usare dove è richiesto un `FLOAT`)
- Come in Java avremo una conversione automaticada `INT` a `FLOAT` se un `INT` è usato in una posizione in cui si richiede un `FLOAT`, ma non il viceversa.

UPO
UNIVERSITÀ DEL PIEMONTE ORIENTALE

Analisi delle espressioni (di **ac**)

- I nodi costante sono corretti e hanno il tipo della costante
- il tipo del nodo `NodeDeRef` è uguale a quello del `NodeId` contenuto, ma per questo si deve controllare che l'identificatore nel `NodeId` sia definito nella symbol table.
- Per i nodi `NodeBinOp`
  - entrambi i nodi devono avere lo stesso tipo (per questo come per l'assegnamento può essere necessario introdurre una conversione se un nodo ha tipo `INT` e l'altro `FLOAT`)
  - il tipo del nodo `NodeBinOp` è uguale a quello dei suoi operandi

UPO
UNIVERSITÀ DEL PIEMONTE ORIENTALE

Analisi del nodo `NodeId`(di **ac**)

Abbiamo visto che ci sono 3 istanze in cui si deve analizzare un `NodeId` che sono:

- quando si analizza il `NodePrint`
- quando si analizza il `NodeAssign`
- quando si analizza il `NodeDeref`

In ogni caso si deve

- controllare che l'identificatore nel `NodeId` sia definito, cioè abbia una entry nella symbol table,
- controllare se l'identificatore è definito e quale è il suo tipo

Quindi questo verrà fatto nell'analisi del `NodeId`.

UPO
UNIVERSITÀ DEL PIEMONTE ORIENTALE