Equivalenza

Due programmi sono detti equivalenti se hanno la stessa semantica, quindi se calcolano la stessa funzione.
L'equivalenza è usata per:

- Analizzare la correttezza, ovvero per dimostrare che il programma scritto calcola esattamente quella funzione;
- Verificare l'efficienza dei due programmi, confrontandoli l'uni con l'altro.

Categorie sintattiche (sintassi pt. 2)

Le categorie sintattiche (ovvero i non terminali della grammatica) consistono nella classificazione dei costrutti in funzione del loro significato atteso, ovvero della classe di effetti che la loro esecuzione causa. In pratica le categorie sintattiche vengono categorizzate in base ai loro effetti sulla stato della macchina.

Stato Con il termine stato si intende l'insieme di ambiente e memoria, dove:

- L'**ambiente** specifica quali nome sono usati per indicare quali oggetti. Si tratta quindi dell'insieme dei legami tra identificazioni e locazioni di memoria. Poichè è possibile che lo stesso nome identifichi più oggetti differenti (ad esempio in scope differenti), non è detto che i legami durino per tutta la vita del programma. Le modifiche all'ambiente sono reversibili, in quanto avvengono solo all'interno dello scope attuale dell'identificatore -> quando l'ambiente termina tutte le modifiche vengono annullate automaticamente;
- La **memoria** rappresenta una mappa delle variazioni dei valori associati agli identificatori. Si tratta quindi dell'insieme dei legami tra locazioni di memoria e valori.

Le categorie sintattiche necessarie ad un linguaggio di programmazione sono distinte in base a cosa denotano/-producono/ottengono. In base a ciò, le categorie sintattiche dei linguaggi di programmazione sono:

- **Espressioni**: sono utilizzate per manipolare valori; per restitutire un valore devono essere valutate. Non effettuano modifiche sullo stato.

  È possibile avere espressioni sintatticamente diverse che ritornano lo stesso valore in tutti gli stati (es: $!(a \&\& b)$ e $(!a)||(|b)$). In questo caso si parla di spressioni equivalenti;
- **Dichiarazioni**: sono utilizzate per creare o modificare i legami tra identificatori e locazioni di memoria (operano sull'ambiente).

  Due dichiarazioni sono equivalenti se producono la stessa memoria in tutti gli stati;
- **Comandi**: sono utilizzati per esegire modifiche sullo stato. Realizzano operazioni irreversibili (definitive nell'esecuzione del programma). I comandi quindi rappresentano funzioni di trasformazione; devono essere eseguiti.

  Due comandi sono equivalenti se per ogni stato in input producono lo stesso stato in output.

Semantica operazionale

La descrizione della semantica è effettuata tramite sistemi di transizione, che sono matematicamente precisi, concisi e permettono l'astrazione. Infatti specificano cosa viene calcolato per induzione sulla struttura sintattica del linguaggio.
Questi sistemi consistono in un insieme di regole date in funzione della sintassi, che specificano cosa avviene induttivamente sulla sintassi.

Definizione: un sistema di transizione è una struttura $(\Gamma, \rightarrow)$ dove Γ è un insieme di elementi chiamati configurazioni e la relazione binaria $\rightarrow \subseteq \Gamma x \Gamma$ è chiamata relazione di transizione. Se $T \subseteq \Gamma$ è un insieme di configurazioni terminali, allora il sistema è detto terminale.

Poichè la relazione di transizione è una relazione, e non una funzione, è possibile avere comportamenti non deterministici. Quindi la semantica è non deterministica.