Linguaggi di programmazione: panoramica

- **Linguaggi di programmazione ad alto livello:**
  - **Programmazione procedurale: C, Pascal, Fortran, Algol**
  - **Programmazione object oriented: Java, C++**
  - **Programmazione di sistema: C**
  - **Programmazione di rete**
  - **Programmazione funzionale: Lisp, Caml, Haskel, Miranda**
  - **Programmazione logica: Prolog**

Linguaggi di programmazione: sintassi e semantica

- La sintassi definisce la forma: come e' fatto un programma del linguaggio (correttezza sintattica)
- Semantica definisce il significato associato a ciascuna forma: cosa calcola ciascun programma (correttezza semantica)

Sintassi: ruolo

- Definisce la forma delle frasi del linguaggio
  - Definisce la ofrma delle frasi del linguaggio
  - Linguaggio del la frasi definisce delle forma
  - It defines la forma delle frasi of the language
- Alfabeto
- Struttura

Sintassi: Alfabeto, parole

- Alfabeto: insieme finito di simboli Λ={a,b,c}
- Frasi: sequenze finite di simboli nell'alfabeto (stringhe di caratteri)
  - Sequenza vuota ε (talvolta indicata con λ)
  - Esempi: ab, abbbb mentre a8bc6,
  - Funzioni su stringhe:
    - lunghezza #abbbb = 5
    - concatenazione se s=ab e t=cc st=abcc

      sε =s

Operazioni

- La concatenazione di sequenze: ovvia “ab” concatenato con “bc” calcola “abbc”
- Il prodotto cartesiano concatenato (è una generalizzazione del prodotto cartesiano) definito su insiemi di sequenze di simboli A x B
- La potenza definito su insiemi di sequenze di simboli A^i
- L’operatore stella definito su insiemi di sequenze di simboli A*
- Operazioni su insiemi.

Prodotto cartesiano (x)

- Il prodotto cartesiano di due insiemi (di sequenze di simboli) A e B è l'insieme di tutte le coppie ordinate (a,b) dove $a \in A$ e $b \in B$.

  $A \times B = \{(a,b) \mid a \in A \text{ e } b \in B\}$

- Esempi dati i due insiemi A e B
  - A= {a,b} , B= {c,d}, A x B={(a,c) ,(a,d),(b,c),(b,d)}
  - A= {1,2} , B= {3,4}, A x B={(1,3),(1,4),(2,3), (2,4)}

Prodotto cartesiano (x)
(concatenato)

- Il prodotto cartesiano concatenato di due insiemi A e B di sequenze di simboli è l'insieme di tutte le coppie che ottengo concatenando ogni elemento del primo insieme con ogni elemento del secondo insieme
- Esempi dati i due insiemi A e B
  - A= {a,b} , B= {c,d}, A x B={ac,ad,bc,bd}
  - A= {cc,b} , B= {c,dc}, A x B={ccc,ccdc,bc,bdc}

Potenze e * su insiemi

$A^0 = \{\varepsilon\}$

$A^1 = A$

$A^2 = A \times A$

$A^* = \cup_i A^i = A^0 \cup A^1 \cup A^2 \cup A^3 \ldots$

Es. A={ab,a,cb}

$A^0 = ?$

$A^1 = ?$

$A^2 = ?$

$A^3 = ?$

$A^* = ?$

Cardinalità di un insieme

Def. Cardinalità (#) di un insieme è il numero dei suoi elementi.

Se A={ab,a,cb}

A=3

A^0 =

A^1

…

A^* =

Sintassi: alfabeto e linguaggio

- Linguaggio su un alfabeto Λ è un insieme (anche infinito) di frasi sull'alfabeto Λ quindi $L \subseteq \Lambda^*$
  - insieme vuoto ({} anche indicato $\varnothing$) è un linguaggio
  - anche Λ^* è un linguaggio
  - in genere ci interessano linguaggi che sono sottoinsiemi propri di Λ^* con cardinalità infinita e proprietà rilevanti (struttura)

Sintassi dei linguaggi di programmazione

- Per i linguaggi di programmazione Λ = ASCII dove ASCII è l'insieme dei caratteri alfanumerici cioè tutti i caratteri presenti sulla tastiera dei computer (standard internazionale).
- Quindi ogni linguaggio di programmazione L abbiamo che $L \subseteq ASCII^*$.
- Un programma è una frase di L quindi un programma è una sequenza di caratteri alfanumerici

Note

- Attenzione

  $\{\} \neq \{\varepsilon\}$

  A= {} , B= {c,dc}, A x B= ?

  A= {ε} , B= {c,dc}, A x B= ?

  B x A = A x B ?

- Ancora: A* x A= ?

  A x A*

Grammatiche e automi

- Grammatiche e automi sono strutture per definire la sintassi dei linguaggi di programmazione.
- Automi sono:
  - meno *potenti* delle grammatiche
  - didatticamente interessanti perché molto semplici
  - l'implementazione di un automa è il riconoscitore del linguaggio.
- Grammatiche sono:
  - Le strutture effettivamente usate per la definizione della sintassi dei linguaggi di programmazione.
  - sono la base per definire il riconoscitore del linguaggio (parser)