Linguaggi di Programmazione

Alessio Marini, 2122855

Appunti presi durante il corso di **Linguaggi di Programmazione** nell'anno **2025/2026** del professsore Pietro Cenciarelli.
Gli appunti li scrivo principalmente per rendere il corso più comprensibile **a me** e anche per imparare il linguaggio Typst. Se li usate per studiare verificate sempre le informazioni 🙏🏼.

Contatti:

alem1105

marini.2122855@studenti.uniroma1.it

September 27, 2025

Indice

1. Algebre e Strutture Dati Induttive

Questa tipologia di Algebre ci servirà a dare un significato alla struttura dei programmi, ovvero la **semantica**, sono inoltre la base matematica di strutture dati come *alberi, liste ecc...*, ci serviranno anche per fare induzione su altre strutture e non solo su sistemi numerici, questa è chiamata **induzione strutturale**.

Ci serviranno delle strutture universali, proviamo ad esempio a descrivere i numeri naturali $\mathbb{N}$ attraverso delle regole, gli **Assiomi di Peano**.

Assiomi di Peano

- $0 \in \mathbb{N}$
- $n \in \mathbb{N} \Rightarrow \text{succ}(n) \in \mathbb{N}$
- $\nexists n \quad \text{t.c.} \quad 0 = \text{succ}(n)$
- $\forall n, m \quad \text{se} \quad \text{succ}(n) = \text{succ}(m) \Rightarrow n = m$
- $\forall S \subseteq \mathbb{N}(0 \in S \land n \in S \Rightarrow \text{succ}(n) \in S) \Rightarrow S = \mathbb{N}$

Grazie a queste regole possiamo «staccarci» dagli elementi dei numeri naturali, abbiamo descritto la loro **struttura**.

L'ultimo degli assiomi viene anche chiamato **assioma di Induzione**, infatti è molto simile al **principio di induzione**.

Principio di Induzione

Data una proprietà P che vale per un $n = 0$, la assumiamo vera per un $n \in \mathbb{N}$ e dimostriamo che è vera anche per $n + 1$, se riusciamo abbiamo dimostrato che P vale $\forall n \in \mathbb{N}$.

In simboli:

$$P(0) \land (P(n) \Rightarrow P(n + 1)) \Rightarrow \forall m \in \mathbb{N} P(m)$$

2. Algebre

Proprietà ed Insiemi

Dire che un elemento appartiene ad un insieme o che soddisfa una proprietà possiamo vederla come la stessa cosa.

Quando definiamo un'algebra dobbiamo definire l'insieme dei suoi elementi le operazioni che ne fanno parte, ad esempio: (A, Γ) e le sue operazioni possono essere:

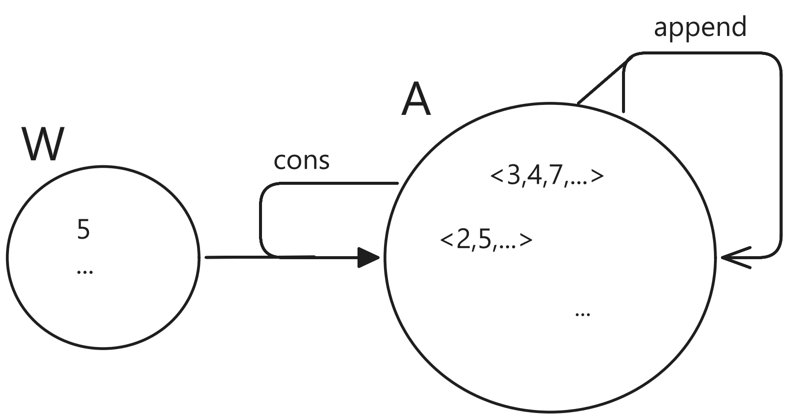
$$\Gamma = \{\Gamma_1, \Gamma_2, \Gamma_3, ...\}$$

Questo serve perché sullo stesso insieme possiamo definire più algebre.

Esempio

Prendiamo come insieme di elementi delle liste di numeri naturali e due operazioni:
- **append**: Prende in input due liste e restituisce la lista che concatena le due prese in input.
- **cons**: Prende in input un numero da $\mathbb{N}$ ed una lista e inserisce il numero all'inizio della lista.

Graficamente abbiamo che:



- $\text{append}(< 3, 4, 7 >, < 2, 5 >) =< 3, 4, 7, 2, 5 >$
- $\text{cons}(5, < 3, 4, 7 >) =< 5, 3, 4, 7 >$

Notiamo che come risultato **abbiamo sempre un elemento dell'algebra**.

Come input possiamo avere anche elementi estranei, se questo accade allora l'algebra prende il nome di **Algebra Eterogenea**.

2.1. Chiusura rispetto ad una funzione

Data un'algebra A prendiamo $S \subseteq A$ e una funziona $f : A \to S$

- S è **chiusa** rispetto a f quando

$$x \in S \Rightarrow f(x) \in S$$

Quindi se prendo come input un elemento da S devo tornare in S, questo deve funzionare anche se prendo come input più elementi.

- Se abbiamo ad esempio un insieme $B \not\subseteq A$ e $S \subseteq A$ allora:

$$\begin{gathered} \forall y \in B \\ x \in S \Rightarrow f(x, y) \in S \end{gathered}$$

- Ultimo caso da tenere in mente è quando come input non abbiamo elementi di S, in questo caso la funzione S è comunque chiusa rispetto ad f dato che stiamo negando la prima parte dell'implicazione.

Adesso, con questo concetto in mente possiamo parlare di Algebre Induttive.

2.2. Algebre Induttive

Definizione

Un Algebra (A, Γ) si dice induttiva quando:
- Tutte le Γ_i sono iniettive
- Tutte le Γ_i hanno immagini disgiunte
- $\forall S \subseteq A$ se S è chiuso rispetto a tutte le Γ_i allora $S = A$

Proviamo a costruire un'algebra induttiva con i numeri naturali usando queste 3 regole e gli assiomi di Peano.

I primi due assiomi di Peano:
- $0 \in \mathbb{N}$
- $n \in \mathbb{N} \Rightarrow \text{succ}(n) \in \mathbb{N}$

Ci danno la segnatura dell'algebra:

$$\left(\mathbb{N}, \underbrace{\{0, \text{succ}, \text{zero}\}}_{\Gamma}\right)$$

La funzione nullaria zero ci serve per rappresentare l'elemento 0.

Funzione Nullaria

Prendiamo come esempio la coppia $(7, 3)$ questa sarà elemento di $\mathbb{N}^2$ mentre $(7, 3, 5)$ sarà elemento di $\mathbb{N}^3$ ma allora $()$ sarà elemento di $\mathbb{N}^0$ e sarà anche l'**unico**. Indichiamo con $\mathbb{1}$ questo insieme.

$$\mathbb{N}^0 = \{()\} = \mathbb{1}$$

Quindi una funzione nullaria su un insieme A avrà una segnatura del tipo $\mathbb{1} \to A$.

Una funzione nullaria su un insieme A può essere vista come un elemento di A.

Vediamo se rispettiamo le proprietà delle algebre induttive:

- Entrambe le funzioni sono induttive, zero è nullaria mentre succ rispetta l'induzione:
  - Vale per 0
  - Se vale per n vale anche per $n + 1$
- Le due funzioni hanno immagini disgiunte, una ha solo 0 come immagine mentre l'altra ha $\mathbb{N} - \{0\}$.
- Prendiamo un $S \subseteq \mathbb{N}$ e supponiamo che sia chiuso su entrambe le funzione succ, zero questo implica che:
  - $0 \in S$ per zero
  - $n \in S \Rightarrow n + 1 \in S$ per succ

Quindi se S è chiuso su entrambe allora abbiamo preso $\mathbb{N}$ e l'algebra è induttiva perché rispettiamo le 3 proprietà.

5 Assiomi - Algebra Induttiva

I 5 Assiomi di Peano sono quindi un caso particolare di Algebra Induttiva con le operazioni zero e succ.

Quando un'algebra è induttiva le sue operazioni Γ_i si chiamano **costruttori dell'algebra**.

2.2.1. Liste finite come algebre induttive

Dato un insieme A, indichiamo con $A -$ list l'insieme delle liste finite di elementi di A. La tupla $(A\text{-list}, \text{empty}, \text{cons})$ é un algebra induttiva dove:
- empty: $\mathbb{1} \to A$-list é la funzione costante che restituisce la lista vuota <>.
- cons: $(A\text{-list} \times A) \to A$-list. Ad esempio: $\text{cons}(3, < 5, 7 >) =< 3, 5, 7 >$. É quindi la funzione che costruisce una lista aggiungendo un elemento in testa.

Questa è un'algebra induttiva, infatti:
- I costruttori hanno immagini disgiunte
- I costruttori sono chiusi per A-list
- C'è un unico modo per costruire ogni lista

Liste Infinite

Le liste infinite non possono essere un'algebra induttiva, infatti contengono una sottoalgebra induttiva, quella delle liste finite che abbiamo appena visto.

2.2.2. Booleani come Algebra non Induttiva

Consideriamo l'algebra (B, not) dove $B = \{0, 1\}$ e $\text{not} : B \to B : b \to \neg b$

- not rispetta le prime due caratteristiche delle algebre induttive
- L'algebra però non rispetta il terzo requisito, infatti se consideriamo $\emptyset \subseteq B$ notiamo che not è chiusa rispetto ad esso, questo perchè se consideriamo un $x \in \emptyset$ e l'implicazione $x \in \emptyset \Rightarrow \text{not}(x) \in \emptyset$ questa risulta vera dato che la premessa è falsa.

Abbiamo quindi trovato un S ovvero $\emptyset$ chiuso per le operazioni dell'algebra ma che è diverso da B. Quindi possiamo dire che (B, not) non è un'algebra induttiva.

2.2.3. Alberi Binari come Algebre Induttive

L'insieme degli alberi binari finiti (B-trees, leaf, branch) dove:

- B-trees: $\{t | t \text{ è una foglia, oppure } t =< t_1, t_2 > \text{ con } t_1, t_2 \in \text{B-trees}\}$
- leaf: $1 \to$ B-trees. un elemento foglia
- branch: B-trees $\times$ B-trees $\to$ B-trees : $(t_{sx}, t_{dx}) \to t$. Costruisce un ramo in modo che t_{sx}, t_{dx} siano i due sottoalberi di t.

È un algebra induttiva.

Teorema

Un albero binario con n foglie ha $2n - 1$ nodi.

Dimostrazione

Possiamo dimostrarlo per induzione strutturale sui costruttori degli alberi:

Caso Base: Consideriamo l'albero formato da una sola foglia, costruito quindi con leaf(). Questo avrà $n = 1$ foglie e $2n - 1 = 1$ nodi.

Ipotesi Induttiva: Ogni argomento dato in input ai costruttori rispetta la proprietà.

Dimostriamo quindi che branch, dati due argomenti che rispettano la proprietà, rispetti la proprietà.

Passo Induttivo: Abbiamo $t = \text{branch}(t_1, t_2)$. Siano:

- $n = n_1 + n_2$ il numero di foglie di t
- n_1 sono le foglie di t_1
- n_2 le foglie di t_2.

Per ipotesi induttiva t_1 ha $2n_1 - 1$ nodi e t_2 ne ha $2n_2 - 1$, dunque t ne avrà

$$(2n_1 - 1) + (2n_2 - 1) + 1$$
$$(+1 \text{ perché c'è se stesso})$$

Che corrisponde a

$$2(n_1 + n_2) - 1 = 2n - 1 \quad \blacksquare$$

3. Espressioni

Definiamo un linguaggio L come un insieme di stringhe. Per descrivere la sintassi di linguaggi formali (la grammatica), usiamo la **BNF (Backus-Naur Form)**, con questa sintassi:

<simbolo> ::= _espressione_

Esempio

Conssideriamo la grammatica:

$$M, N ::= 5|7|M + N|M * N$$

Le espressioni che rispettano questa grammatica sono del tipo:
- «5» o «7»
- Un'espressione del tipo $M + N, M * N$ che rispetta a sua volta la grammatica.

Introduciamo una funzione $\text{eval} : L \to \mathbb{N}$ che valuta le espressioni del linguaggio:
- $\text{eval}(5) = 5$
- $\text{eval}(7) = 7$
- $\text{eval}(M + N) = \text{eval}(M) + \text{eval}(N)$
- $\text{eval}(M * N) = \text{eval}(M) * \text{eval}(N)$

Notiamo che nell'esempio precedente l'algebra (L, eval) non è induttiva, infatti una stringa $5 + 7 * 5$ potrebbe essere stata generata in due modi diversi: $(5 + 7) * 5$ e $5 + (7 * 5)$.

Possiamo però considerare $5, 7, +, *$ come costruttori dell'algebra e in questo modo $(5 + 7) * 5 \neq 5 + (7 * 5)$, si potrebbe quindi dimostrare come $(L, 5, 7, +, *)$ è un'algebra induttiva.