UNIVERSITÀ DEGLI STUDI DI SALERNO

DIPARTIMENTO DI INFORMATICA

Corso di Laurea in Informatica

# Appunti Metodi Matematici per l'Informatica

Autore:

**Oleg BILOVUS**

**Mat. 0512105721**

ANNO ACCADEMICO 2020/2021

# ABSTRACT

Gli appunti sono molto brevi e sintetici in quanto non sono pensati allo studio della materia, bensì al ripasso. Gli appunti possono contenere errori, quindi non sono da considerare veritieri.

# INDICE

# Parte I

# Logica, insiemi e dimostrazioni

# CAPITOLO 1

# LOGICA PROPOSIZIONALE

## 1.1 Perché studiare la logica proposizionale

La materia insegna a distinguere quelle che sono delle proposizioni, ossia che hanno valore **True** *oppure* **False**, e quelle che invece non lo sono.

La frase *Che bella giornata* non è una proposizione, in quanto essa può essere sia True che False.

Allo stesso modo affermazioni matematiche del tipo $x + 2 = 5$ non è una proposizione in quanto il valore della $x$ può variare e con esso varia il valore di verità della proposizione in True o False.

Invece proposizioni del tipo *La penna è sul tavolo* oppure $5 + 4 = 0$ sono proposizioni in quanto hanno un valore di verità univoco al momento della loro affermazione.

## 1.2 Proposizioni

### 1.2.1 Semplici

Le proposizioni semplici sono dette tali quando hanno un valore **T** o **F** e non sono usati connettivi logici.

**Esempio 1.2.1.1**

*La televisione è accesa*

**Esempio 1.2.1.2**

$3 + 5 = 8$

### 1.2.2 Composte

Le proposizioni composte sono dette tali quando hanno un valore **T** o **F** e sono usati connettivi logici.

**Esempio 1.2.2.1**

*Laura fa i compiti* ***e*** *ascolta la musica*

**Esempio 1.2.2.2**

*Se domani piove* ***allora*** *prenderò l'ombrello*

## 1.3 Connettivi logici

### 1.3.1 NOT

Il connettivo logico $\neg$ ha valore **T** se e solo se la proposizione $p$ ha valore **F**, e ha valore **F** se e solo se la proposizione $p$ ha valore **T**.

Tabella 1.1: Tabella di verità del connettivo logico NOT.

| $p$ | $\neg p$ |
|---|---|
| T | F |
| F | T |

### 1.3.2 AND

Il connettivo logico $\wedge$ ha valore **T** se e solo se entrambe le proposizioni $p$ e $q$ hanno valore **T**, se una delle due è **F** allora il valore della proposizione è **F**.

Tabella 1.2: Tabella di verità del connettivo logico AND.

| $p$ | $q$ | $p \wedge q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

### 1.3.3 OR

Il connettivo logico $\vee$ ha valore **T** se e solo se una delle proposizioni $p$ o $q$ ha valore **T**, se sono entrambe **F** allora il valore della proposizione è **F**.

Tabella 1.3: Tabella di verità del connettivo logico OR.

| $p$ | $q$ | $p \vee q$ |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

### 1.3.4 XOR

Il connettivo logico $\oplus$ ha valore **T** se e solo se una delle le proposizioni $p$ o $q$ ha valore **T** ma non entrambe, se sono entrambe **F** o **T** allora il valore della proposizione è **F**.

Tabella 1.4: Tabella di verità del connettivo logico XOR.

| $p$ | $q$ | $p \oplus q$ |
|---|---|---|
| T | T | F |
| T | F | T |
| F | T | T |
| F | F | F |

### 1.3.5 Implicazione

Il connettivo logico $\implies$ ha valore **F** se e solo se la proposizione $p$ ha valore **T** e la proposizione $q$ ha valore **F**, altrimenti il valore della proposizione è **T**.

Tabella 1.5: Tabella di verità del connettivo logico Implicazione.

| $p$ | $q$ | $p \implies q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | T |
| F | F | T |

### 1.3.6 Equivalenza

Il connettivo logico $\iff$ ha valore **T** se e solo se la proposizione $p$ e la proposizione $q$ hanno valori uguali, altrimenti il valore della proposizione è **F**.

Tabella 1.6: Tabella di verità del connettivo logico Equivalenza.

| $p$ | $q$ | $p \iff q$ |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | T |

## 1.4 Equivalenza logica

Due proposizioni composte $p$ e $q$ si dicono equivalenti logicamente se e solo se hanno la stessa tabella di verità e si indica con $\mathbf{p} \equiv \mathbf{q}$.

**N.B:** $\equiv$ non è un connettivo logico.

**N.B:** $\equiv$ e $\iff$ sono diversi.

**N.B:** Non si usa il simbolo = per esprimere l'equivalenza logica, ma si usa il simbolo $\equiv$.

### 1.4.1 Tautologia

La tautologia è una proposizione composta sempre **True**, ossia tutte **T** nella colonna della proposizione composta, qualsiasi sia il valore delle proposizioni elementari che la compongono.

Tabella 1.7: Tabella di verità della tautologia $(p \land q) \implies (p \lor q)$.

| $p$ | $q$ | $p \land q$ | $p \lor q$ | $(p \land q) \implies (p \lor q)$ |
|---|---|---|---|---|
| T | T | T | T | T |
| T | F | F | T | T |
| F | T | F | T | T |
| F | F | F | F | T |

### 1.4.2 Contraddizione

La contraddizione è una proposizione composta sempre **False**, ossia tutte **F** nella colonna della proposizione composta, qualsiasi sia il valore delle proposizioni elementari che la compongono.

Tabella 1.8: Tabella di verità della contraddizione $p \land \neg p$.

| $p$ | $\neg p$ | $p \land \neg p$ |
|---|---|---|
| T | F | F |
| F | T | F |

## 1.5 Inverso, opposto e contronominale

### 1.5.1 Inverso

Inverso di $p \implies q$ è $q \implies p$

$p \implies q \not\equiv q \implies p$

Tabella 1.9: Tabella di verità di $p \implies q \not\equiv q \implies p$.

| $p$ | $q$ | $p \implies q$ | $q \implies p$ |
|---|---|---|---|
| T | T | T | T |
| T | F | F | T |
| F | T | T | F |
| F | F | T | T |

**Esempio 1.5.1.1**

*Se domani piove* ***allora*** *prenderò l'ombrello*

Inverso: *Prenderò l'ombrello* ***se*** *domani piove*

### 1.5.2 Opposto

Opposto di $p \implies q$ è $\neg p \implies \neg q$

$p \implies q \not\equiv \neg p \implies \neg q$

Tabella 1.10: Tabella di verità di $p \implies q \not\equiv \neg p \implies \neg q$.

| $p$ | $q$ | $\neg p$ | $\neg q$ | $p \implies q$ | $\neg p \implies \neg q$ |
|---|---|---|---|---|---|
| T | T | F | F | T | T |
| T | F | F | T | F | T |
| F | T | T | F | T | F |
| F | F | T | T | T | T |

**Esempio 1.5.2.1**

*Se domani piove* ***allora*** *prenderò l'ombrello*

Opposto: *Se domani non piove* ***allora*** *non prenderò l'ombrello*

### 1.5.3 Contronominale

Contronominale di $p \implies q$ è $\neg q \implies \neg p$

$p \implies q \equiv \neg p \implies \neg q$

Tabella 1.11: Tabella di verità di $p \implies q \equiv \neg p \implies \neg q$.

| $p$ | $q$ | $\neg p$ | $\neg q$ | $p \implies q$ | $\neg q \implies \neg p$ |
|---|---|---|---|---|---|
| T | T | F | F | T | T |
| T | F | F | T | F | F |
| F | T | T | F | T | T |
| F | F | T | T | T | T |

**Esempio 1.5.3.1**

*Se domani piove* ***allora*** *prenderò l'ombrello*

Contronominale: *Non prenderò l'ombrello* ***se*** *domani non piove*

### 1.5.4 Osservazione: l'inverso è equivalente logicamente all'opposto

Dalla Tabella 1.9 e dalla Tabella 1.10 si ha che $q \implies p \equiv \neg p \implies \neg q$

Tabella 1.12: Tabella di verità di $q \implies p \equiv \neg p \implies \neg q$.

| $p$ | $q$ | $\neg p$ | $\neg q$ | $q \implies p$ | $\neg p \implies \neg q$ |
|---|---|---|---|---|---|
| T | T | F | F | T | T |
| T | F | F | T | T | T |
| F | T | T | F | F | F |
| F | F | T | T | T | T |

## 1.6 Equivalenze logiche note

### 1.6.1 Leggi di De Morgan

- $\neg(p \land q) \equiv \neg p \lor \neg q$
- $\neg(p \lor q) \equiv \neg p \land \neg q$
- $p \implies q \equiv \neg p \lor q$
- $\neg(p \implies q) \equiv \neg(\neg p \lor q) \equiv p \land \neg q$

### 1.6.2 Equivalenze più usate

- $\neg(\neg p) \equiv p$
- $p \lor (q \land r) \equiv (p \lor q) \land (p \lor r)$
- $p \land (q \lor r) \equiv (p \land q) \lor (p \land r)$
- $(x < y < z) \equiv (x < y) \land (y < z)$
- $p \iff q \equiv (p \implies q) \land (q \implies p) \equiv (\neg p \lor q) \land (\neg q \lor p)$

### 1.6.3 Altre equivalenze

- $p \land T \equiv p$
- $p \lor F \equiv p$
- $p \lor T \equiv T$
- $p \land F \equiv F$
- $p \lor p \equiv p$
- $p \land p \equiv p$
- $p \lor \neg p \equiv T$
- $p \land \neg p \equiv F$

# CAPITOLO 2

# LOGICA PREDICATIVA

## 2.1 Predicato

Esprime una proprietà che un oggetto di un gruppo può avere o non avere o relazioni tra gli oggetti di un gruppo.

**Esempio 2.1.0.1**

$P(x, y)$ $x$ ama $y$

*Luca ama Laura*

Il predicato $P$ è una funzione con dominio l'universo del discorso e codominio il valore **T** o **F**.

## 2.2 Quantificatori

Essi vengono usati per esprime una proprietà su un gruppo di oggetti o l'esistenza di un oggetto con una proprietà in un gruppo.

### 2.2.1 Universale

Il quantificatore $\forall$ viene usato per esprimere una proprietà su un gruppo di oggetti. Per essere **T**, tutti gli oggetti nel dominio devono rispettare le proprietà.

**Esempio 2.2.1.1**

*Tutti i laureati in informatica hanno fatto l'esame di MMI*

$P(x)$, $x$ è laureato in informatica

$Q(x)$, $x$ ha fatto l'esame di MMI

Dominio di $x$ sono tutti gli umani

$\forall x(P(x) \implies Q(x))$

**N.B:** Se $\forall x P(x)$ è falso, ciò non significata che tutti gli oggetti del dominio di $x$ non rispettino la proprietà $P$, ma che esiste almeno un oggetto che non rispetta le proprietà.

### 2.2.2 Esistenziale

Il quantificatore $\exists$ viene usato per esprime l'esistenza di un oggetto con una determinata proprietà in un gruppo di oggetti. Affinché sia **T**, deve esistere almeno un oggetto che rispetti le proprietà.

**Esempio 2.2.2.1**

*Alcuni laureati in informatica hanno fatto l'esame di programmazione avanzata*

$P(x)$, $x$ è laureato in informatica

$Q(x)$, $x$ ha fatto l'esame di programmazione avanzata

Dominio di $x$ sono tutti gli umani

$\exists x(P(x) \land Q(x))$

### 2.2.3 Dominio vuoto

Se il dominio del predicato $P(x)$ è $\emptyset$, i quantificatori hanno i seguenti comportamenti:

- $\forall x P(x)$: è **T** in quanto non si può trovare un controesempio per dimostrare che è **F**.
- $\exists x P(x)$ è **F** in quanto non si può trovare un esempio per dimostrare che è **T**.

### 2.2.4 Cambio dominio

Se si cambia il dominio di un predicato, il valore di verità dell'intera proposizione può cambiare totalmente.

**Esempio 2.2.4.1**

*Alcuni laureati in informatica hanno fatto l'esame di programmazione avanzata*
$P(x)$, $x$ ha fatto l'esame di programmazione avanzata
Dominio di $x$ tutti i laureati in informatica. Allora avremmo che $\exists x P(x)$ è **T**.
Se cambiamo il dominio di $x$ in tutti i laureati di filosofia, avremmo che $\exists x P(x)$ è **F**.

### 2.2.5 Quantificatori innestati

I quantificatori $\forall$ e $\exists$ si possono combinare tra di loro per formare predicati più complessi.

**Esempio 2.2.5.1**

*Ogni figlio ha una madre*
$P(x)$, $x$ è un figlio
$Q(x)$, $x$ è una madre
$R(x, y)$, $x$ è figlio di $y$
Dominio di $x$ e $y$ sono tutti gli umani
$\forall x \exists y (R(x, y) \wedge P(x) \wedge Q(y))$

**Esempio 2.2.5.2**

*Ognuno apprezza qualcuno*
$P(x, y)$, $x$ apprezza y

Dominio di $x$ e $y$ sono tutti gli umani

$\forall x \exists y P(x, y)$

### 2.2.6 Negazione quantificatori

I quantificatori possono essere negati. La regola generale è che se il quantificatore è $\forall$, diventa $\exists$ e viceversa e poi si nega il predicato.

- $\neg(\forall x P(x)) \equiv \exists x \neg P(x)$
- $\neg(\exists x P(x)) \equiv \forall x \neg P(x)$

#### Negazione quantificatori innestati

La regola generale è che si cambiano i quantificatori con il loro opposto e si nega il predicato.

**Esempio 2.2.6.1**

$\neg(\forall x(P(x) \implies Q(x))) \equiv$

$\exists x \neg(P(x) \implies Q(x)) \equiv$

$\exists x \neg(\neg P(x) \vee Q(x)) \equiv$

$\exists x(P(x) \wedge \neg Q(x))$

**Esempio 2.2.6.2**

Negare il seguente predicato $\forall x(x > 0 \implies \exists y \mid (x \cdot y = 1))$

$\neg(\forall x(x > 0 \implies \exists y \mid (x \cdot y = 1))) \equiv$

Si può spostare $\exists y$ all'inizio, in quanto lo scope di y non cambierebbe e risulta più facile la negazione

$\neg(\forall x(x > 0 \implies \exists y \mid (x \cdot y = 1))) \equiv$

$\neg(\forall x \exists y(x > 0 \implies (x \cdot y = 1))) \equiv$

$\exists x \forall y \neg(x > 0 \implies (x \cdot y = 1))) \equiv$

$\exists x \forall y \neg(\neg(x > 0) \vee (x \cdot y = 1)) \equiv$

$\exists x \forall y((x > 0) \wedge \neg(x \cdot y = 1)) \equiv$

$\exists x \forall y((x > 0) \wedge (x \cdot y \neq 1))$

### 2.2.7 Dominio finito

Se il dominio del predicato è finito ed è presente un quantificatore, esso può essere anche rimosso a seconda del quantificatore e la proposizione continuerà ad avere lo stesso valore di verità.

**Esempio 2.2.7.1**

Supponiamo che il dominio del predicato $P(x)$ abbia $n$ oggetti. Avremmo che le seguenti equivalenze sono **T**:

- $\forall x P(x) \iff P(x_1) \land P(x_2) \land ... \land P(x_n)$
- $\exists x P(x) \iff P(x_1) \lor P(x_2) \lor ... \lor P(x_n)$
- Funzione proposizionale $Q(x, y)$ con dominio $\{1, 2\} \times \{1, 2\}$
  $\exists x \forall y Q(x, y) \iff (Q(1, 1) \land Q(1, 2)) \lor (Q(2, 1) \land Q(2, 2))$
- Funzione proposizionale $Q(x, y)$ con dominio $\{1, 2\} \times \{1, 2\}$
  $\forall x \exists y Q(x, y) \iff (Q(1, 1) \lor Q(1, 2)) \land (Q(2, 1) \lor Q(2, 2))$

## 2.3 Insieme di verità

L'insieme di verità di $P(x)$ è l'insieme degli elementi $x_i$ nel dominio di $P(x)$ tali che $P(x)$ è **T**.

In termini matematici l'insieme di verità rappresenta un sottoinsieme della controimmagine di $P(x)$ tale che $P(x) = \mathbf{T}$
Il codominio di $P(x)$ è $C = \{T, F\}$, sia $D$ il dominio di $P(x)$, $V$ l'insieme di verità di $P(x)$ e $A = \{T\} \subset C$.
Si ha che l'insieme di verità di $P(x)$ è $f^{-1}(A) = \{x \in D \mid P(x) = T\} = V$

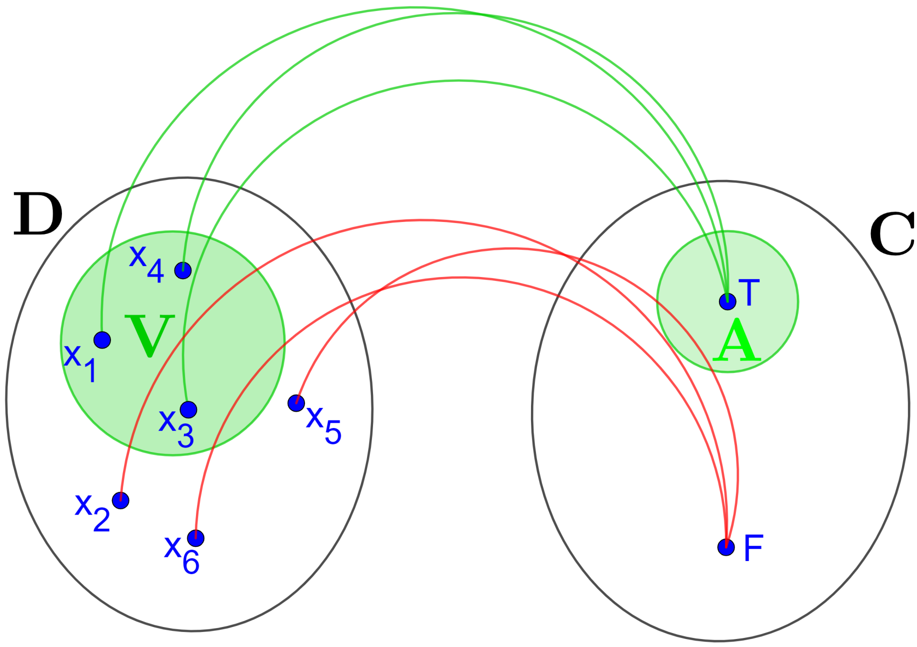


Figura 2.1: Rappresentazione grafica esempio di insieme di verità.

# CAPITOLO 3

## INSIEMI

- $A = B \iff A \subseteq B \wedge B \subseteq A$
- $A = B : \forall x(x \in A \iff x \in B) \equiv \forall x((x \in A \implies x \in B) \wedge (x \in B \implies x \in A))$
- $A \subseteq B : \forall x(x \in A \implies x \in B)$
- $A \nsubseteq B : \neg(\forall x(x \in A \implies x \in B) \equiv$
  $\exists x \neg(x \in A \implies x \in B) \equiv$
  $\exists x \neg(\neg(x \in A) \vee x \in B)) \equiv$
  $\exists x(x \in A \wedge x \notin B)$
- $\emptyset \in S : \forall x(x \in \emptyset \implies x \in S)$ è sempre **True** perché $x \in \emptyset$ è sempre falsa e dalla Tabella 1.5, se $p$ è **F**, la proposizione è sempre **T**.
- $S \subseteq S : \forall x(x \in S \implies x \in S)$ è sempre **True** perché $p \implies p$ è una tautologia.

Tabella 3.1: Tabella di verità di $p \implies p$.

| $p$ | $p \implies p$ |
|---|---|
| T | T |
| F | T |

- $A \subset B : \forall x (x \in A \implies x \in B) \land \exists y (y \in B \land y \notin A)$
- $A \times B = \{(a, b) \mid a \in A \land b \in B\}$

# CAPITOLO 4

# DIMOSTRAZIONI

## 4.1 Dimostrazione diretta

$$p \implies q$$

**Esempio 4.1.0.1**

*Se $n$ è dispari allora $n^2$ è dispari.*

**Dimostrazione**: Sia $n$ un intero dispari.

$n^2 = (2k+1)^2 = 4k^2 + 1 + 4k = 2(2k^2 + 2k) + 1$

Quindi $n^2$ è dispari.

## 4.2 Dimostrazione per contrapposizione

$$\neg q \implies \neg p$$

Dalla Tabella 1.11 si ha che $\neg q \implies \neg p \equiv p \implies q$

**Esempio 4.2.0.1**

Se $3n + 2$ è dispari allora $n$ è dispari.

**Contronominale**: Se $n$ è pari allora $3n + 2$ è pari.

**Dimostrazione**: Sia n pari.

$3n + 2 = 3(2k) + 2 = 2(3k) + 2 = 2(3k + 1)$

$2(3k + 1)$ è sempre un numero pari in quanto $2 \cdot r, r \in \mathbb{R}$ è sempre pari.

Avendo dimostrato che $\neg q \implies \neg p$ è **True** e poiché $\neg q \implies \neg p \equiv p \implies q$, si ha che $3n + 2$ *è dispari allora* $n$ *è dispari* è **True**.

## 4.3 Dimostrazione per assurdo o contraddizione

$p \implies q$ è **T** quando $p \wedge \neg q$ è **F**

Si ricorda che dalle Leggi di De Morgan si ha che $\neg(p \implies q) \equiv p \wedge \neg q$

Tabella 4.1: Tabella di verità di $p \implies q$ e $p \wedge \neg q$.

| $p$ | $q$ | $\neg q$ | $p \implies q$ | $p \wedge \neg q$ |
|---|---|---|---|---|
| T | T | F | T | F |
| T | F | T | F | T |
| F | T | F | T | F |
| F | F | T | T | F |

**Esempio 4.3.0.1**

*Se* $3n + 2$ *è dispari allora* $n$ *è dispari.*

$p \wedge \neg q$: $3n + 2$ è dispari e $n$ è pari.

**Dimostrazione**:

$3n + 2 = 3(2k) + 2 = 2(3k + 1)$

Nella Dimostrazione per contrapposizione abbiamo visto che $2(3k+1)$ è sempre pari. Poiché $3n + 2 = 2(3k + 1)$ allora $3n + 2$ è pari, ma avevamo detto che era dispari. Quindi è un assurdo e $p \wedge \neg q$ è **F**, quindi $p \implies q$ è **T**, cioè *Se* $3n + 2$ *è dispari allora* $n$ *è dispari* è **T**.

## 4.4 Dimostrazione banale o vuota

$p \implies q$, $p$=**F** allora dalla Tabella 1.5 $p \implies q$ è sempre **True**.

## 4.5 Dimostrazione di equivalenza

$p \iff q$

Si dimostra $(p \implies q) \land (q \implies p)$

## 4.6 Controesempio

$\forall x P(x)$

$\neg(\forall x P(x)) \equiv \exists x \neg P(x)$

## 4.7 Prova di esistenza

$\exists x P(x)$

- Può essere costruttiva esibendo un elemento $x$ nel dominio per cui $P(x)$ è **True**.
- Può essere non costruttiva.

## 4.8 Dimostrazione per casi

$(p_1 \lor p_2 \lor ... \lor p_n) \implies q \equiv (p_1 \implies q) \lor (p_2 \implies q) \lor ... \lor (p_n \implies q)$

## 4.9 Dimostrazione esaustiva

«TODO»

# Parte II

# Ricorsione, principio induzione e relazioni di ricorrenze

# CAPITOLO 5

# DEFINIZIONE FUNZIONI RICORSIVAMENTE

## 5.1 Funzione matematica in $\mathbb{N}$

**Passo base**: Specificare il valore della funzione in 0.

**Passo ricorsivo**: Definire la regola per ottenere il valore della funzione su un intero attraverso i valori della funzione su interi più piccoli.

**Esempio 5.1.0.1**

$f : \mathbb{N} \longrightarrow \mathbb{N}$

**Passo Base**: $f(0) = 3$

**Passo ricorsivo**: $f(n + 1) = 2f(n) + 3$

La funzione target, ad esempio $f(3)$, si può calcolare buttom up o top down.

### 5.1.1 Calcolo buttom up

Si parte dal **Passo base**, in quanto si conosce il valore della funzione e si arriva a ogni iterazione al valore della funzione target.

**Esempio 5.1.1.1**

$f(0) = 3$

$f(1) = 2 \cdot 3 + 3 = 9$

$f(2) = 2 \cdot 9 + 3 = 21$

$f(3) = 2 \cdot 21 + 3 = 45$

### 5.1.2 Calcolo top down

Si parte dal target, ad esempio $f(3)$, e si scrive la sua funzione ricorsiva. Si scende fino al **Passo base** in quanto non si conosce il valore delle precedenti funzioni ricorsive. Arrivati al **Passo base**, si inizia a scrivere il valore delle funzioni che si conoscono a ritroso, in questo modo si arriverà al valore della funzione target.

**Esempio 5.1.2.1**

$f(3) = 2f(2) + 3 = 2 \cdot 21 + 3 = 45$

$f(2) = 2f(1) + 3 = 2 \cdot 9 + 3 = 21$

$f(1) = 2f(0) + 3 = 2 \cdot 3 + 3 = 9$

$f(0) = 3$

## 5.2 Dalla definizione ricorsiva alla funzione ricorsiva

Per passare dalla definizione ricorsiva alla funzione ricorsiva bisogna scoprire il meccanismo ricorsivo. Lo si può fare calcolando il **Passo base**, la funzione in $n$, cioè $f(n)$, e la funzione in $f(n + 1)$.

**N.B**: la funzione ricorsiva è $f(n + 1)$.

**N.B**: nella funzione ricorsiva $f(n + 1)$ ci deve essere sempre una chiamata a valori di funzioni precedenti a $f(n+1)$, altrimenti non è una funzione ricorsiva.

**Esempio 5.2.0.1**

Fibonacci

**Definizione ricorsiva**

**Passo base**: $F_0 = F_1 = 1$

**Passo ricorsivo**: $F_n = F_{n-1} + F_{n-2}, n \geq 2$

**Funzione ricorsiva**

$f(0) = 1$

$f(1) = 1$

$f(n) = f(n-1) + f(n-2)$

$f(n+1) = f((n+1)-1) + f((n+1)-2) = f(n) + f(n-1)$

**Esempio 5.2.0.2**

$f : \mathbb{N} \longrightarrow \mathbb{N}$

**Definizione ricorsiva**

$f(n) = a^n$

**Funzione ricorsiva**

$f(0) = a^0 = 1$

$f(n) = a^n$

$f(n+1) = a^{n+1} = a \cdot a^n = a \cdot f(n)$

**N.B.**: Se avessimo scritto solamente $f(n+1) = a^{n+1} = a \cdot a^n$, non è corretto, in quanto non è una definizione ricorsiva poiché non vi sono chiamate ai valori delle funzioni precedenti a $f(n+1)$.

# CAPITOLO 6

# DEFINIZIONE INSIEMI RICORSIVAMENTE

## 6.1 Insiemi numerici

Applicare il passo ricorsivo un paio di volte, capire cosa contiene l'insieme e darne la costruzione ricorsiva. La correttezza della costruzione ricorsiva sarà dimostrata successivamente con il Principio di induzione.

**Esempio 6.1.0.1**

**Definizione ricorsiva**

**Passo base**: $1 \in A$

**Passo ricorsivo**: Se $x \in A$, allora $x + 2 \in A$

**Costruzione ricorsiva**

Applico il passo ricorsivo

$x = 1 \in A$, allora $x + 2 = 1 + 2 = 3 \in A$

Applico il passo ricorsivo

$x = 3 \in A$, allora $x + 2 = 3 + 2 = 5 \in A$

Si può notare che $A$ contiene i numeri dispari, quindi la sua costruzione ricorsiva è: $A = \{2n + 1 \mid n \geq 0, n \in \mathbb{N}\}$

**Esempio 6.1.0.2**

**Definizione ricorsiva**

**Passo base**: $3 \in S$

**Passo ricorsivo**: Se $x, y \in S$, allora $x + y \in S$

**Costruzione ricorsiva**

Applico il passo ricorsivo

$x = 3 \in S, y = 3 \in S$, allora $x + y = 3 + 3 = 6 \in S$

Applico il passo ricorsivo

$x = 6 \in S, y = 3 \in S$, allora $x + y = 6 + 3 = 9 \in S$

Applico il passo ricorsivo

$x = 9 \in S, y = 3 \in S$, allora $x + y = 9 + 3 = 12 \in S$

Applico il passo ricorsivo

$x = 9 \in S, y = 6 \in S$, allora $x + y = 9 + 6 = 15 \in S$

Si può notare che $S$ contiene i numeri multipli di 3, quindi la sua costruzione ricorsiva è: $S = \{3n \mid n \geq 1, \in \mathbb{N}\}$

**Esempio 6.1.0.3**

**Definizione ricorsiva**

**Passo base**: $1 \in T$

**Passo ricorsivo**: Se $x \in T$, allora $3 \cdot x \in T$

**Costruzione ricorsiva**

Applico il passo ricorsivo

$x = 1 \in T$, allora $3 \cdot x = 3 \cdot 1 = 3 \in T$

Applico il passo ricorsivo

$x = 3 \in T$, allora $3 \cdot x = 3 \cdot 3 = 9 \in T$

Applico il passo ricorsivo

$x = 9 \in T$, allora $3 \cdot x = 3 \cdot 9 = 27 \in T$

Si può notare che $T$ contiene i numeri $3^n$, quindi la sua costruzione ricorsiva è: $T = \{3^n \mid n \in \mathbb{N}\}$

# CAPITOLO 7

# DEFINIZIONE STRINGHE RICORSIVAMENTE

## 7.1 Definizioni

- L'alfabeto $\Sigma$ è un insieme di simboli per creare stringhe.
- La stringa è una **sequenza** di simboli presi da un alfabeto $\Sigma$.
- $\lambda$ è la **stringa vuota** che non contiene simboli. $\lambda$ è una **stringa** e non un simbolo dell'alfabeto $\Sigma$, quindi $\lambda \notin \Sigma$.
- $\Sigma^*$ è l'insieme di tutte le possibili stringhe sull'alfabeto $\Sigma$.
- L'insieme $\Sigma^*$ è infinito e $\lambda \in \Sigma^*$.

## 7.2 Definizione ricorsiva di $\Sigma^*$

**Passo base**: $\lambda \in \Sigma^*$

**Passo ricorsivo**: Se $w \in \Sigma^*$ e $x \in \Sigma$, allora $wx \in \Sigma^*$

**Esempio 7.2.0.1**

$\Sigma = \{0, 1\}$

Dal **Passo base** si ha che $\lambda \in \Sigma^*$. Quindi al **Passo base** $\Sigma^* = \{\lambda\}$

Applico il passo ricorsivo

$w = \lambda \in \Sigma^*, x = 0 \in \Sigma$, allora $wx = \lambda 0 = 0 \in \Sigma^*$. Quindi $\Sigma^* = \{\lambda, 0\}$

Applico il passo ricorsivo

$w = \lambda \in \Sigma^*, x = 1 \in \Sigma$, allora $wx = \lambda 1 = 1 \in \Sigma^*$. Quindi $\Sigma^* = \{\lambda, 0, 1\}$

Applico il passo ricorsivo

$w = 0 \in \Sigma^*, x = 0 \in \Sigma$, allora $wx = 00 \in \Sigma^*$. Quindi $\Sigma^* = \{\lambda, 0, 1, 00\}$

Applico il passo ricorsivo

$w = 0 \in \Sigma^*, x = 1 \in \Sigma$, allora $wx = 01 \in \Sigma^*$. Quindi $\Sigma^* = \{\lambda, 0, 1, 00, 01\}$

...

## 7.3 Lunghezza di una stringa

**Passo base**: $|\lambda| = 0$

**Passo ricorsivo**: Se $w \in \Sigma^*$ e $x \in \Sigma$, allora $|wx| = |w| + 1$

**Esempio 7.3.0.1**

$|abb| = |ab| + |b| = |a| + |b| + |b| = 1 + 1 + 1 = 3$

## 7.4 Concatenazione di una stringa

$u$ e $v$ due stringhe, la concatenazione di $u$ e $v$ è la stringa $u \cdot v$. Si indica anche semplicemente con $uv$ senza usare il $\cdot$

**N.B**: $uv \neq vu$

**Passo base**: Se $w \in \Sigma^*$, allora $w\lambda = w$

**Passo ricorsivo**: Se $w_1, w_2 \in \Sigma^*$ e $x \in \Sigma$, allora $w_1 \cdot (w_2 x) = (w_1 \cdot w_2)x \in \Sigma^*$

**Esempio 7.4.0.1**

$abb \cdot ab = abb \cdot (ab) = abba \cdot b = abbab$

## 7.5 Potenza di una stringa

**Passo base**: $w^0 = \lambda$

**Passo ricorsivo**: $w^{n+1} = w^n \cdot w, \forall n \geq 0$

**Esempio 7.5.0.1**

$\{(aa)^i \mid 0 \leq i \leq 3\} = \{\lambda, aa, aaaa, aaaaaa\}$

**Esempio 7.5.0.2**

$\{aa^i \mid 0 \leq i \leq 3\} = \{\lambda, aa, aaa, aaaa\}$

**N.B**: $(aa)^i \neq aa^i$, $(aa)^2 = aaaa, aa^2 = aaa$

## 7.6 Stringa palindroma

**Passo base**: $\forall x \in \Sigma$ e $\lambda$ sono stringhe palindrome

**Passo ricorsivo**: Se $w$ è una stringa palindroma e $x \in \Sigma$, allora $xwx$ è una stringa palindroma.

**Esempio 7.6.0.1**

*abba*

Per il **Passo base** $a, b, \lambda$ sono stringhe palindrome

Applico il passo ricorsivo

$w = \lambda$ palindroma per il **Passo base** e $x = b \in \Sigma$, allora $xwx = b\lambda b = bb$ è una stringa palindroma.

Applico il passo ricorsivo

$w = bb$ palindroma per il passo ricorsivo precedente e $x = a \in \Sigma$, allora $xwx = abba$ è una stringa palindroma.

## 7.7 Inversione di una stringa

**Passo base**: $\lambda^R = \lambda$

**Passo ricorsivo**: Se $w \in \Sigma^*$ e $x \in \Sigma$, allora $(wx)^R = xw^R$

**Esempio 7.7.0.1**

$(abb)^R = b(ab)^R = bb(a)^R = bba$

**Esempio 7.7.0.2**

$a^R = (\lambda a)^R = a\lambda^R = a\lambda = a$

## 7.8 Ulteriori definizioni di specifiche stringhe

### 7.8.1 Stringhe su $\{a, b\}$ di lunghezza pari

**Passo base**: $\lambda \in S$

**Passo ricorsivo**: Se $w \in S$, allora $waa, wab, wba, wbb \in S$

### 7.8.2 Stringhe pari su $\{a, b\}$ che iniziano con a

**Passo base**: $aa, ab \in S$

**Passo ricorsivo**: Se $w \in S$, allora $waa, wab, wba, wbb \in S$

### 7.8.3 Morfismo

Prende in input $w \in \{a, b\}^* \setminus \lambda$ e sostituisce $a$ con 0 e $b$ con 1

**Passo base**: $change(a) = 0$, $change(b) = 1$

**Passo ricorsivo**: $change(wx) = \begin{cases} change(w)0, \text{ se } x = a \\ change(w)1, \text{ se } x = b \end{cases}$

# CAPITOLO 8

# DEFINIZIONE ALBERI RADICATI RICORSIVAMENTE

## 8.1 Definizioni

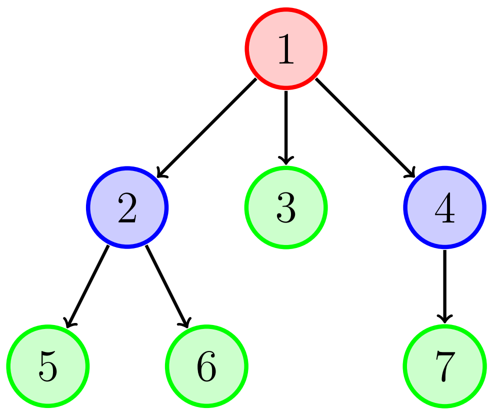


Figura 8.1: Esempio albero radicato.

- Un albero radicato per definizione ha **una sola radice**, nella Figura 8.1 in questo caso è il nodo 1.
- Un albero radicato ha **una o più foglie**, nella Figura 8.1 sono i nodi 3, 5, 6 e 7.

- Un albero radicato può avere **uno o più genitori/nodi interni**, essi per definizione sono nodi che hanno almeno un figlio, cioè che non sono foglie, nella Figura 8.1 sono i nodi 2, 4 e anche la radice nodo 1 in questo caso è un genitore/nodo interno.

- Un albero radicato può essere rappresentato da una coppia di insiemi $T = (V, E)$. Prendiamo come esempio la Figura 8.1.
  - $V$ rappresenta i nodi, detti anche vertici o dall'inglese ***V****ertexes*.
    $V = \{1, 2, 3, 4, 5, 6, 7\}$.
  - $E$ rappresenta gli archi, dall'inglese ***E****dges*. L'insieme è composto a sua volta da coppie di nodi *(genitore, figlio)* dall'insieme $V$.
    $E \subset V \times V$
    $E = \{(1, 2), (1, 3), (1, 4), (2, 5), (2, 6), (4, 7)\}$.
    **Osservazione**: la radice compare solo a sinistra delle coppie in quanto è l'unico nodo che è solo genitore e non ha genitori. Le foglie invece compaiono solo a destra delle coppie poiché per definizione non hanno figli.
  - Infine la coppia viene chiamata $T$, dall'inglese ***T****ree*, cioè albero.
    $T = (\{1, 2, 3, 4, 5, 6, 7\}, \{(1, 2), (1, 3), (1, 4), (2, 5), (2, 6), (4, 7)\})$.

## 8.2 Definizione ricorsiva

**Passo base**: $T = (\{r\}, \{\emptyset\})$ è un albero radicato

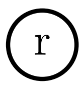


Figura 8.2: Passo base definizione ricorsiva albero radicato.

**Passo ricorsivo**: Supponiamo che $T_1 = (V_1, E_1), ..., T_n = (V_n, E_n)$ siano alberi radicati disgiunti, cioè $\bigcap_{i=1}^{n} V_i = \emptyset$ (gli alberi non hanno nodi in comune). Le respettive radici sono $r_1 \in V_1, ..., r_n \in V_n$. Allora $T = (V, E)$ si ottiene ponendo come radice un nodo $r \notin V_1 \cup ... \cup V_n$ e da $r$ si aggiunge un arco a ogni $r_1 \in V_1, ..., r_n \in V_n$.

$V = \{r\} \cup V_1 \cup ... \cup V_n$

$E = \{(r, r_1), ..., (r, r_n)\} \cup E_1 \cup ... \cup E_n$

$T = (V, E)$ è un albero radicato.

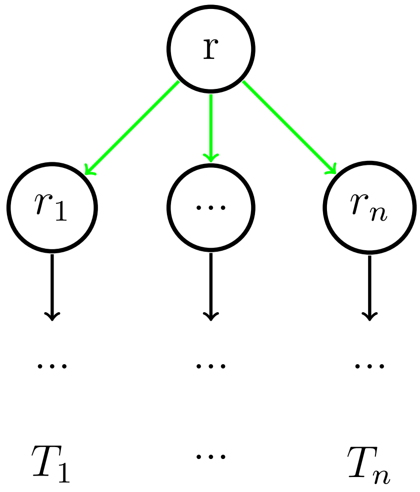


Figura 8.3: Passo ricorsivo definizione ricorsiva albero radicato.

## 8.3 Numero di vertici

**Passo base**: Se $T = (\{r\}, \emptyset)$, allora $|V| = 1$

**Passo ricorsivo**: Se $T = (V, E)$ è un albero radicato costruito a partire dagli alberi $T_1 = (V_1, E_1), ..., T_n = (V_n, E_n)$, allora $|V| = 1 + |V_1| + ...|V_n|$.

**N.B.**: L'1 rappresenta la radice $r \notin V_1 \cup ... \cup V_n$, così come definito nel Passo ricorsivo della costruzione dell'albero radicato. Nella Figura 8.3 è il vertice $r$.

## 8.4 Numero di edges(archi)

**Passo base**: Se $T = (\{r\}, \{\emptyset\})$, allora $|E| = 0$

**Passo ricorsivo**: Se $T = (V, E)$ è un albero radicato costruito a partire da $T_1 = (V_1, E_1), ..., T_n = (V_n, E_n)$, allora $|E| = n + |E_1| + ... + |E_n|$.

**N.B.**: $n$ sono gli archi dalla radice ai vertici $r_1 \in V_1, ..., r_n \in V_n$, nella Figura 8.3 sono gli archi in verde.

## 8.5 Numero di foglie

Sia $f(T)$ la funzione che prende in input un albero e restituisce il numero di foglie di esso.

**Passo base**: Se $T = (\{r\}, \emptyset)$, allora $f(T) = 1$

**Passo ricorsivo**: Se $T = (V, E)$ è un albero radicato costruito a partire da $T_1 = (V_1, E_1), ..., T_n = (V_n, E_n)$, allora $f(T) = f(T_1) + ... + f(T_n)$.

**N.B.**: La radice non viene aggiunta al conteggio del numero di foglie, perché $T_1, ..., T_n$ essendo alberi radicati, dal Passo base della definizione di albero radicato sappiamo che hanno almeno un vertice, ossia la radice. Poiché $T$ è costruito a partire da $T_1 = (V_1, E_1), ..., T_n = (V_n, E_n)$, la radice di $T$ avrà $n$ figli e per definizione di foglia, non può essere una foglia se ha figli.

## 8.6 Numero di nodi interni

Sia $i(T)$ la funzione che prende in input un albero e restituisce il numero di nodi interno di esso.

**Passo base**: Se $T = (\{r\}, \emptyset)$, allora $i(T) = 0$

**Passo ricorsivo**: Se $T(V, E)$ è un albero radicato costruito a partire da $T_1 = (V_1, E_1), ..., T_n = (V_n, E_n)$, allora $i(T) = 1 + i(T_1) + ... + i(T_n)$.

**N.B.**: L'1 è la radice, in quanto nel Passo ricorsivo non è più una foglia. Il perché è spiegato nel "N.B." di Numero di foglie.

## 8.7 Altezza di un vertice

**N.B.**: L'altezza di un vertice si conta dal basso verso l'alto.

- Se $v$ è una foglia, allora l'altezza di $v$ è 0.
- Altrimenti l'altezza di $v$ è la massima altezza tra i figli di $v$ più 1.

**Osservazione**: L'altezza di un albero è l'altezza della sua radice.

Prendendo come esempio Figura 8.1, i nodi di colore rosso hanno altezza 2, quelli di colore blu hanno altezza 1 e quelli di colore verde hanno altezza 0.

## 8.8 Profondità di un vertice

**N.B.**: La profondità di un vertice si conta dall'alto verso il basso.

- Se $v$ è la radice, allora la profondità di v è 0.
- Altrimenti la profondità di $v$ è la profondità del padre di $v$ più 1.

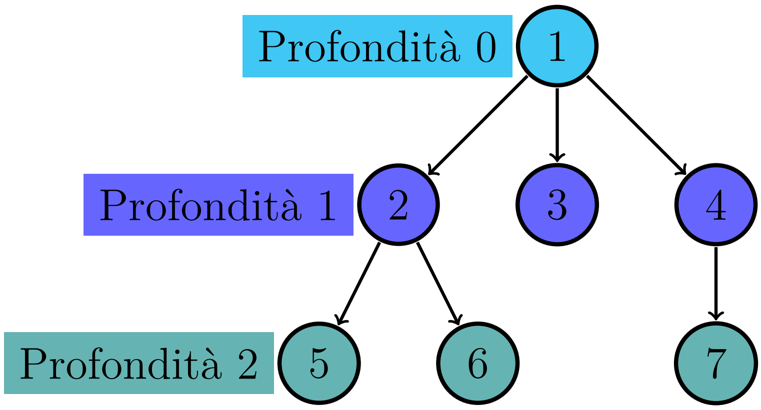


Figura 8.4: Esempio profondità di un vertice.

# CAPITOLO 9

# DEFINIZIONE ALBERI BINARI RICORSIVAMENTE

## 9.1 Definizioni

- Un albero binario ha la caratteristica che ogni vertice può avere al massimo 2 figli.

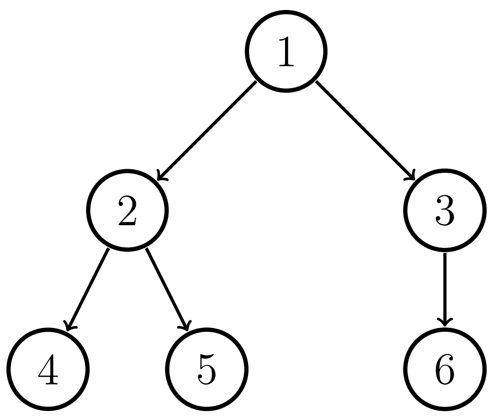


Figura 9.1: Esempio di albero binario.

- Un albero binario pieno ha 0 o 2 figli a ogni vertice.

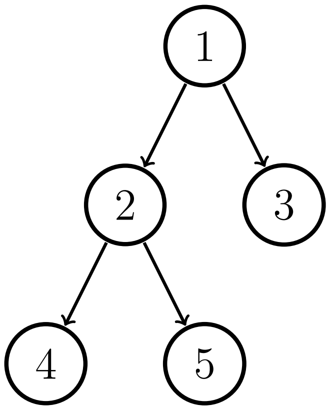


Figura 9.2: Esempio di albero binario pieno.

- Un albero binario pieno completo ha 2 figli a ogni vertice.

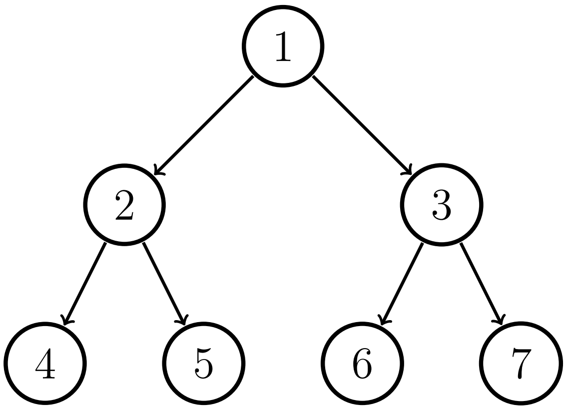


Figura 9.3: Esempio di albero binario pieno completo.

- Poiché vi è un limite sui figli che ogni vertice può avere, ad ogni profondità/livello $d$ dell'albero ci possono essere al massimo $2^d$ vertici.

## 9.2 Definizione ricorsiva albero binario pieno

**Passo base**: $T = (\{r\}, \emptyset)$ è un albero binario pieno.

**Passo ricorsivo**: Supponiamo che $T_1(V_1, E_1)$ e $T_2(V_2, E_2)$ siano alberi binari pieni disgiunti, cioè $V_1 \cap V_2 = \emptyset$ con radici $r_1 \in V_1$ e $r_2 \in V_2$. Allora $T = (V, E)$ si ottiene ponendo come radice un nodo $r \notin V_1 \cup V_2$ e da $r$ si aggiunge un arco a $r_1 \in V_1$ e $r_2 \in V_2$.

$V = \{r\} \cup V_1 \cup V_2$

$E = \{(r, r_1), (r, r_2)\} \cup E_1 \cup E_2$

$T = (V, E)$ è un albero binario pieno.

## 9.3 Definizione ricorsiva di albero binario

**Passo base**: $T = (\emptyset, \emptyset)$ è un albero binario.

**Passo ricorsivo** Supponiamo che $T_1(V_1, E_1)$ e $T_2(V_2, E_2)$ siano alberi binari disgiunti, cioè $V_1 \cap V_2 = \emptyset$ con radici $r_1 \in V_1$ e $r_2 \in V_2$. Allora $T = (V, E)$ si ottiene ponendo come radice un nodo $r \notin V_1 \cup V_2$ e da $r$ si aggiunge un arco a $r_1 \in V_1$ e $r_2 \in V_2$.

$V = \{r\} \cup V_1 \cup V_2$

$E = \{(r, r_1), (r, r_2)\} \cup E_1 \cup E_2$

$T = (V, E)$ è un albero binario.

# CAPITOLO 10

# PRINCIPIO DI INDUZIONE

## 10.1 Principio di induzione matematico

**Passo base**: Provare che il Passo base è vero, ossia $P(m)$ è vera, dove è $m$ è l'intero più piccolo del dominio.

**Passo induttivo**: Supporre per **Ipotesi induttiva** che $P(k)$ è vera, provare che $P(k+1)$ è vera. Per farlo, è essenziale usare l'ipotesi induttiva.
$P(k) \implies P(k+1)$

**N.B.**: Giustificare ogni uguaglianza non banale.

## 10.2 Principio di induzione forte

**Passo base**: Provare che il Passo base è vero, ossia $P(m)$ è vera, dove è $m$ è l'intero più piccolo del dominio. Il Passo base può variare, possono essere anche più proposizioni vere nel passo base.

**Passo induttivo**: Supporre per **Ipotesi induttiva** che $P(m), ..., P(k)$ è vera, provare che $P(k+1)$ è vera. Per farlo, è essenziale usare l'ipotesi induttiva.
$(P(m) \wedge P(m+1) \wedge ... \wedge P(k)) \implies P(k+1)$

**N.B.**: Giustificare ogni uguaglianza non banale.

## 10.3 Principio di induzione strutturale

**Passo base**: Provare che l'enunciato $P$ è vero per ogni elemento dell'insieme specificato nel **Passo base** della definizione ricorsiva dell'insieme.

**Passo induttivo**: Supporre per **Ipotesi induttiva** che l'enunciato $P$ è vero per gli elementi nell'insieme, provare che l'enunciato è vero quando si costruiscono nuovi elementi dell'insieme usando il Passo ricorsivo dell'insieme e l'Ipotesi induttiva.

**N.B.**: Giustificare ogni uguaglianza non banale.

## 10.4 Quale induzione usare

Per dimostrare che due definizioni sono uguali, si deve dimostrare che $def_1 \subseteq def_2$ e $def_2 \subseteq def_1$.

- $def_{nonRicorsiva} \subseteq def_{ricorsiva}$: si usa il principio di induzione matematico in cui si fa induzione su un $k$.
- $def_{ricorsiva} \subseteq def_{nonRicorsiva}$: si usa il principio di induzione strutturale in cui si fa induzione sul Passo ricorsivo della definizione ricorsiva dell'insieme.
- $\forall w \in def_{ricorsiva} P(w)$: Si usa il principio di induzione strutturale anche quando si deve dimostrare che la $def_{ricorsiva}$ ha una proprietà.

## 10.5 Principio induzione sulle stringhe

**Passo base**: a seconda di quale induzione si usa, provare che gli elementi dell'insieme nel Passo base della definizione ricorsiva dell'insieme sono sottoinsiemi della definizione non ricorsiva o hanno una certa proprietà nel caso del principio di induzione strutturale, il viceversa nel caso di induzione matematica.

**Passo induttivo**: a seconda di quale induzione si usa, si suppone per **Ipotesi induttiva** che l'insieme è sottoinsieme della definizione non ricorsiva oppure ha una certa proprietà e si dimostra che costruendo gli altri elementi dell'insieme usando il Passo ricorsivo, i nuovi elementi sono sempre sottoinsiemi della definizione non ricorsiva o hanno una certa proprietà nel caso di induzione strutturale, il viceversa nel caso di induzione matematica. Usare una $w$ appartenente alla definizione ricorsiva che sia diversa dagli elementi nel Passo base.

## 10.6 Esempi

**Esempio 10.6.0.1**

**Traccia**

**Passo base**: $a \in B, b \in B$

**Passo ricorsivo**: Se $w \in B$, allora $wbb \in B$ e $wba \in B$

Utilizzando il Principio di induzione, dimostrare che ogni elemento di $B$ ha lunghezza dispari.

**Svolgimento**

Bisogna dimostrare che $\forall w \in B, |w| = 2k + 1, k \geq 0$

Dimostrazione per il Principio di induzione strutturale, poiché si dimostra che una definizione ricorsiva ha una proprietà.

**Passo base**: $|a|$ = (definizione lunghezza stringa) $= 1 = 2 \cdot 0 + 1$, $|b|$ =(definizione lunghezza stringa)$= 1 = 2 \cdot 0 + 1$

**Passo induttivo**:

**Ipotesi induttiva**: $|w| = 2k + 1$, $w \in B$.

Sia $w \in B, w \neq a, w \neq b$. Allora $w = ubb$ oppure $w = uba$, $u \in B$. Per ipotesi induttiva esiste un $k \geq 0$ tale che $|w| = 2k + 1$.

Per la definizione ricorsiva di lunghezza di stringa, risulta che:

- $|w| = |ubb| = |ub| + 1 = |u| + 2 = 2k + 1 + 2 = 2(k + 1) + 1$

- $|w| = |uba| = |ub| + 1 = |u| + 2 = 2k + 1 + 2 = 2(k + 1) + 1$

Il Passo induttivo è stato dimostrato e pertanto l'enunciato è vero.

**Esempio 10.6.0.2**

**Traccia**

$A \subset \{a, b\}^*$

**Passo base**: $b \in A$

**Passo ricorsivo**: Se $w \in A$, allora $wab \in A$

Dimostrare che $\forall n \in \mathbb{N}, b(ab)^n \in A$

**Svolgimento**

Dimostrazione per Principio di induzione matematico, in quanto si dimostra $def_{nonRicorsiva} \subseteq def_{ricorsiva}$.

**Passo base**: $b(ab)^0$ =(definizione potenze stringhe)= $b\lambda$ =(definizione concatenazione stringhe)= $b \in A$

**Passo induttivo**:

**Ipotesi induttiva**: $b(ab)^k = w \in A$.

Sia $w \in A, w \neq b$. Allora $w = uab, u \in A$.

$b(ab)^{k+1} = b(ab)^k \cdot ab = wab \in A$

Il passo induttivo è stato dimostrato e pertanto l'enunciato è vero.

## 10.7 Principio di induzione strutturale sugli alberi radicati

**Passo base**: Provare che $P(T)$ è vera se $T = (\{r\}, \emptyset)$

**Passo induttivo**: Sia $T = (V, E)$ un albero costruito a partire dagli alberi radicati $T_1 = (V_1, E_1), ..., T_n = (V_k, E_k)$. Per **Ipotesi induttiva** $P(T_1), ..., P(T_k)$ sono vere, provare usando l'ipotesi induttiva che $P(T)$ è vera quando si costruisce $T$.

**Esempio 10.7.0.1**

**Traccia**

Per ogni albero radicato $T = (V, E)$ risulta $|V| = |E| + 1$.

**Svolgimento**

**Passo base**: $T = (\{r\}, \emptyset)$, $|V| = 1 = |\emptyset| + 1 = 0 + 1 = 1$

**Passo induttivo**:

Sia $T = (V, E)$ costruito a partire dagli alberi $T_1 = (V_1, E_1), ..., T_k = (V_k, E_k)$.

**Ipotesi induttiva**: $|V_i| = |E_i| + 1, 1 \leq i \leq k$

$V =$(definizione albero radicale)$= r \cup V_1 \cup ... \cup V_k$

$E =$(definizione albero radicale)$= \{(r, r_1), ..., (r, r_k)\} \cup E_1 \cup ... \cup E_k$

$|V| = r+|V_1|+...+|V_k| =$(ipotesi induttiva)$= 1+|E_1|+1+...+|E_k|+1 = |E|+1$

Il passo ricorsivo è stato dimostrato e pertanto l'enunciato è vero.

## 10.8 Principio induzione strutturale sugli alberi binari pieni e non

**Passo base**: Provare che $P(T)$ è vera se $T = (\{r\}, \emptyset)$. Nel caso di albero binario non pieno, si prova che $P(T)$ è vera se $T = (\emptyset, \emptyset)$. Il Passo induttivo è uguale.

**Passo induttivo**: Sia $T = (V, E)$ un albero binario pieno o non costruito a partire dagli alberi binari pieno o non $T_1 = (V_1, E_1)$ e $T_2 = (V_2, E_2)$. Per **Ipotesi induttiva** $P(T_1)$ e $P(T_2)$ sono vere, provare usando l'ipotesi induttiva che $P(T)$ è vera quando si costruisce $T$.

**Esempio 10.8.0.1**

**Traccia**

Sia $f(T)$ la funzione che prende in input un albero e restituisce il numero di foglie di esso.

Sia $h(T)$ la funzione che prende in input un albero e restituisce la sua altezza.

In un albero binario $T$ il numero di foglie è minore o uguale di $2^h$, dove $h$ è l'altezza di $T$, cioè $f(T) \leq 2^{h(T)}$.

**Svolgimento**

**Passo base**:

$T = (\emptyset, \emptyset)$ oppure $T = (\{r\}, \emptyset)$

$h(T) = 0$

$f(T) = 0 \leq 2^0 = 1$ oppure $f(T) = 1 \leq 2^0 = 1$

**Passo induttivo**:

Sia $T = (V, E)$ un albero binario pieno o non costruito a partire dagli alberi $T_1 = (V_1, E_1)$ e $T_2 = (V_2, E_2)$.

**Ipotesi induttiva**: $f(T_i) \leq 2^{h(T_i)}, 1 \leq i \leq 2$

Abbiamo due casi a seconda che la radice abbia uno o due figli:

- Nel primo caso abbiamo che $h(T_1) = h(T) - 1$ e che $f(T) = f(T_1)$ e quindi si ha che $f(T) = f(T_1) \leq 2^{h(T_1)} = 2^{h(T)-1} < 2^{h(T)}$.
- Nel secondo caso abbiamo che $h(T_i) = h(T) - 1, 1 \leq i \leq 2$ e che $f(T) = f(T_1) + f(T_2)$, quindi si ha che $f(T) = f(T_1) + f(T_2) \leq 2^{h(T_1)} + 2^{h(T_2)} = 2^{h(T)-1} + 2^{h(T)-1} = 2 \cdot 2^{h(T)-1} = 2^{h(T)}$

Il passo induttivo è stato dimostrato e pertanto l'enunciato è vero.

# CAPITOLO 11

# RELAZIONI DI RICORRENZA

## 11.1 Metodo di iterazione

Una relazione di ricorrenza può essere risolta con il metodo di iterazione.

- Per prima cosa calcolare il valore per alcune chiamate ricorsive a $T$ in $T(n)$, ossia calcolare $T(--)$ in $T(n) = ...T(--)...$ per trovare un pattern nelle soluzioni.

- Una volta individuato il pattern, sostituirlo con opportune variabili, ad esempio $i$ e porre le variabili in un intervallo in modo tale che la condizione iniziale di $n > k$ sia vera. Se non è presente chiaramente questa condizione, deve essere ricavata. Si ricava semplicemente ponendo $n > m$, dove $m$ è il "Passo base" della relazione di ricorrenza, ad esempio se è $T(1) = ...$, allora $n > 1$.

- Porre $i$ al suo valore massimo nell'intervallo, in quanto di sta calcolando $T(n)$.

- Fare i calcoli e ci si ritroverà con la risoluzione della relazione di ricorrenza.

**N.B.**: In generale durante i calcoli di $T(n) = ...T(--)...$ deve scomparire qualsiasi richiamo a $T(--)$, l'unico modo per farlo scomparire è che $--$ sia uguale $m$, dove $m$ è il "Passo base" della relazione di ricorrenza, poiché ne conosciamo il valore di $T(m)$ si sostituisce il richiamo a $T(m)$ con il suo valore. Ad esempio $T(1) = 4$, allora $m = 1$ e $-- = m = 1$ e si sostituisce la chiamata a $T(m)$ con 4. Questo può aiutare con i calcoli e a vedere si sta facendo bene.

**Esempio 11.1.0.1**

**Traccia**

$T(1) = 2$

$T(n) = T(n-1) + 3, n > 1$

**Svolgimento**

$T(n) = T(n-1) + 3 =$

[1] $= [T(n-2) + 3] + 3 = T(n-2) + 3 \cdot 2 =$

[2] $= [T(n-3) + 3] + 2 \cdot 3 = T(n-3) + 3 \cdot 3$

...

$T(n) = T(n-i) + 3 \cdot i,\ 1 \leq i \leq n-1$

$i \leq n-1$ perché abbiamo bisogno che $n-i$ sia uguale a 1 perché $T$ inizia da 1.

$T(n) = T(n-(n-1) + 3(n-1) =$

$T(n-n+1) + 3n - 3 =$

$T(1) + 3n - 3 = 2 + 3n - 3 = 3n - 1$

$T(n) = a_n = 3n - 1$

La relazione di ricorrenza è stata risolta, si verifica con il Principio di induzione matematico se è vera.

[1] $T(n-1) = T((n-1)-1) + 3 = T(n-2) + 3$
[2] $T(n-2) = T((n-2)-1) + 3 = T(n-3) + 3$

## 11.2 Principio di induzione matematico sulle relazioni di ricorrenze

**Passo base**: Provare che $a_m = T(m)$, dove $m$ rappresenta il più piccolo elemento del dominio.

**Passo induttivo**: Provare che $T(n) = a_n$. Per **Ipotesi induttiva** in $T(n) = ...T(--)...$ si ha che $a_{--} = T(--)$. Sostituire $T(--)$ con $a_{--}$ nella definizione di $T(n)$, fare i calcoli e risulterà che $T(n) = a_n$.

**Esempio 11.2.0.1**

**Traccia**

Dall'esempio precedente dimostrare che $T(n) = T(n-1)+3 = a_n = 3n-1$

**Svolgimento**

**Passo base**: $a_1 = 3 \cdot 1 - 1 = 3 - 1 = 2 = T(1)$

**Passo induttivo**:

**Ipotesi induttiva**: $a_{n-1} = T(n-1) = 3(n-1) - 1 = 3n - 3 - 1 = 3n - 4$

$T(n) = T(n-1) + 3 = a_{n-1} + 3 =$

$= [3n - 4] + 3 = 3n - 1$

Il passo induttivo è stato dimostrato e pertanto l'enunciato è vero.

## 11.3 Esempio più complesso

**Esempio 11.3.0.1**

**Traccia**

$T(1) = 1$

$T(n) = 2T(\frac{n}{2}) + n$, $n > 1$ e $n$ potenza di 2.

**Svolgimento**

$T(n) = 2T(\frac{n}{2}) + n =$

[3] $= 2[2T(\frac{n}{2^2}) + \frac{n}{2}] + n = 2^2T(\frac{n}{2^2}) + n + n = 2^2T(\frac{n}{2^2}) + 2n =$

[4] $= 2^2[2T(\frac{n}{2^3}) + \frac{n}{2^2}] + 2n = 2^3T(\frac{n}{2^3}) + n + 2n = 2^3T(\frac{n}{2^3}) + 3n$

[3] $T(\frac{n}{2}) = 2T((\frac{n}{2})/2) + \frac{n}{2} = 2T(\frac{n}{2^2}) + \frac{n}{2}$

[4] $T(\frac{n}{2^2}) = 2T((\frac{n}{2^2})/2) + \frac{n}{2^2} = 2T(\frac{n}{2^3}) + \frac{n}{2^2}$

...

$T(n) = 2^i T(\frac{n}{2^i}) + i \cdot n$, $1 \leq i \leq \log_2 n$

**N.B.**: $i \leq \log_2 n$ perché abbiamo bisogno che $\frac{n}{2^i}$ sia uguale a 1 perché $T$ inizia da 1 e quindi che $2^i$ sia uguale a $n$. Dalle proprietà matematiche si ha che $x^{\log_x n} = n$.

$T(n) = 2^{\log_2 n} T(\frac{n}{2^{\log_2 n}}) + \log_2 n \cdot n =$

$= nT(n/n) + n \log_2 n =$

$= n \cdot 1 + n \log_2 n = n + n \log_2 n$

**Dimostrazione**

**Passo base**: $a_1 = 1 + 1 \log_2 1 = 1 + 0 = 1 = T(1)$

**Passo induttivo**:

**Ipotesi induttiva**: $n$ potenza di 2 e $a_{\frac{n}{2}} = T(\frac{n}{2}) = \frac{n}{2} + \frac{n}{2} \log_2 \frac{n}{2}$

$T(n) = 2T(\frac{n}{2}) + n = 2a_{\frac{n}{2}} + n =$

$= 2[\frac{n}{2} + \frac{n}{2} \log_2 \frac{n}{2}] + n$

$= n + n \log_2 \frac{n}{2} + n =$

$= n + n(\log_2 n - \log_2 2) + n =$

$= n + n \log_2 n - n + n =$

$= n + n \log_2 n = a_n$

Il passo induttivo è stato dimostrato e pertanto l'enunciato è vero.

# ELENCO DELLE FIGURE

# ELENCO DELLE TABELLE