Corso di Laurea Triennale in Informatica

Università di Pisa

FONDAMENTI DELL'INFORMATICA

Dispense per il corso di Laurea in Informatica

Filippo Bonchi, Alessio Conte, Andrea Corradini, Roberto Grossi

[Anno accademico 2020-21] 1 dicembre 2020

Indice

CAPITOLO 1

Insiemi

In questo capitolo esamineremo la nozione di *insieme*, richiamando brevemente le principali operazioni di confronto e di composizione tra insiemi e le leggi che le regolano. Infine vedremo come dimostrare l'uguaglianza di insiemi usando tre tecniche diverse: le dimostrazioni basate sui diagrammi di Eulero-Venn, intuitive e che non richiedono particolare creatività, ma di applicabilità limitata; le dimostrazioni per sostituzione, che sfruttano le leggi sugli insiemi e alcune semplici tecniche di manipolazione algebrica; e le dimostrazioni discorsive, basate direttamente sulla definizione di uguaglianza tra insiemi. Nel presentare le tecniche di dimostrazione daremo enfasi anche alla costruzione di controesempi per confutare enunciati erronei. Il capitolo si concluderà con due costruzioni di importanza fondamentale per il resto del corso: l'insieme delle parti e il prodotto cartesiano tra due insiemi.

1.1 Rappresentare gli insiemi

Definizione 1.1.1 (Insieme)**.** *Un* INSIEME *è una collezione di oggetti, detti* ELEMENTI*.*
Dati un oggetto a e un insieme A, scriviamo $a \in A$ per dire che a è un elemento di A (oppure a appartiene ad *A), mentre scriviamo $a \notin A$ per dire che a non è un elemento di A (oppure a* non appartiene ad *A). Il simbolo "$\in$" è chiamato* simbolo di appartenenza*.*

Come vedremo più avanti, l'ordine in cui sono presentati gli elementi di un insieme non è importante, come non lo è il numero di ripetizioni di un elemento.

Gli insiemi sono quindi usati per raggruppare oggetti. Tipicamente gli oggetti vengono raggruppati perché posseggono qualche caratteristica comune, ma non necessariamente.

Esempio 1.1.2 (Alcuni insiemi)**.**

- *I numeri interi / naturali / reali.*
- *I numeri pari / dispari / primi.*
- *Le lettere dell'alfabeto italiano / inglese.*
- *Le lettere dell'alfabeto che compaiono anche nei cognomi degli studenti.*
- *Le lettere dell'alfabeto che non compaiono come iniziali degli studenti.*
- *I giocatori di una squadra di calcio.*
- *Gli studenti dell'Università di Pisa.*
- *Gli studenti che seguono il corso di Fondamenti dell'Informatica.*
- *Gli utenti di Facebook / Twitter / Instagram.*
- *Le pagine web.*

Dagli esempi si capisce che gli elementi di un insieme possono essere di qualsiasi natura, non solo astrazioni matematiche. Normalmente useremo lettere maiuscole come $A, B, C, \ldots$ per denotare gli insiemi, e lettere minuscole come $a, b, c, \ldots$ per denotarne gli elementi.

Insiemi definiti per enumerazione

Si può definire un insieme *per enumerazione* (oppure *in modo estensionale*) elencandone tutti gli elementi tra parentesi graffe e separati da virgole. Vediamo alcuni esempi:

- I giorni della settimana: $\{$ lunedì, martedì, mercoledì, giovedì, venerdì, sabato, domenica $\}$.
- Le ore in un giorno: $\{$ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23 $\}$.
- Le vocali $V = \{ \mathsf{a}, \mathsf{e}, \mathsf{i}, \mathsf{o}, \mathsf{u} \}$.
- I valori di verità $\{\mathtt{t}, \mathtt{f}\}$. Qui il simbolo `t` sta per "true" ("vero" in inglese) e `f` sta per "false" ("falso" in inglese). Questo insieme è particolarmente importante in Informatica e viene chiamato anche l'insieme dei valori booleani:[1]

$$Bool = \{\mathtt{t}, \mathtt{f}\}$$

E se gli insiemi sono molto grandi, o infiniti? Si usano i puntini (...) per sottintendere una regola di enumerazione:

- I minuti in un'ora $\{ 1, 2, ..., 60 \}$
- $K = \{ 1, 2, ..., 1000 \}$ (i numeri da 1 a mille)
- $\mathbb{N} = \{ 0, 1, 2, ... \}$ (i **naturali**)
- $\mathbb{N}^+ = \{ 1, 2, ... \}$ (i **naturali positivi**)
- $\mathbb{Z} = \{ ..., -2, -1, 0, 1, 2... \}$ (gli **interi**)

Un insieme può anche contenere elementi di natura diversa:

- $AN = \{ a, b, ..., z, A, B, ..., Z, 0, 1, ..., 9 \}$ (i caratteri alfanumerici)
- $\{42, \text{mercoledì}, a, 10:10, \text{Roma}\}$

L'insieme che non contiene nessun elemento è chiamato l'INSIEME VUOTO e viene rappresentato con il simbolo $\varnothing$:

- $\varnothing = \{\}$

Insiemi definiti per proprietà

In alternativa si può definire un insieme *per proprietà* (oppure *in modo intensionale*) descrivendo una proprietà soddisfatta da tutti e soli i suoi elementi. Di seguito indichiamo con P una generica proprietà e con la notazione $P(a)$ il fatto che la proprietà P vale per l'elemento a (diciamo anche che *a soddisfa P*). Assumiamo inoltre che presi un elemento a e una proprietà P, o a soddisfa P, oppure no. Per esempio, se a è un numero, $P(a)$ indica se a è un numero pari.

Dato insieme A, l'insieme di tutti gli elementi di A che godono della proprietà P è denotato da

$$X = \{ x \mid x \in A \text{ e } P(x) \}$$

O più semplicemente:

$$X = \{ x \in A \mid P(x) \}$$

O ancora più semplicemente (quando A è ovvio dal contesto):

$$X = \{ x \mid P(x) \}$$

[1] Dal nome di George Boole (1815–1864), importante logico e matematico inglese.

Il simbolo | si legge *"tale che"* e serve a specificare una condizione, riportata subito dopo; quindi scrivere $K = \{ x \in A \mid P(x) \}$ significa definire K come l'insieme di tutti gli elementi x appartenti ad A tali che x soddisfa P.

Vediamo alcuni esempi, usando il linguaggio naturale per specificare le proprietà (più avanti nel corso vedremo come essere più rigorosi):

- $\mathbb{N}^p = \{ n \mid n \in \mathbb{N}^+ \text{ e } n \text{ è divisibile per } 2 \} = \{ n \in \mathbb{N}^+ \mid n \text{ è divisibile per } 2 \} = \{2, 4, 6, \ldots\}$ (i numeri **pari positivi**)
- $\mathbb{N}^d = \{ n - 1 \mid n \in \mathbb{N}^p \} = \{1, 3, 5, \ldots\}$ (i numeri **dispari positivi**)
- $\mathbb{Q} = \{ n/m \mid n \in \mathbb{Z}, m \in \mathbb{N}^+\}$ (i **razionali**)
- $\mathbb{R} = \{ x \mid x \text{ è un numero reale} \}$
- $\mathbb{P} = \{ p \in \mathbb{N}^+ \mid p \text{ ha esattamente due divisori distinti}\}$ (i **numeri primi**)[2]
- $AB = \{ a \mid a \text{ è un anno bisestile} \}$[3]

Esempio 1.1.3. *Un altro esempio interessante è fornito dall'insieme delle pagine web che d'ora in avanti denoteremo con W. L'insieme*

$$PippoW = \{x \in W \mid \textit{la parola "Pippo" occorre in } x\}$$

rappresenta l'insieme di tutte le pagine web in cui appare la parola "Pippo". Cercando su un motore di ricerca (ad esempio Google) la parola "Pippo", si ottiene come risultato esattamente l'insieme PippoW. L'aspetto più interessante dei motori di ricerca è l'ordine in cui vengono visualizzati i risultati (nel nostro caso, gli elementi di PippoW), ma questo affascinante argomento esula dagli obbiettivi di queste note.

I paradossi

La definizione di insieme che abbiamo dato è piuttosto intuitiva, ma poco precisa (per esempio siamo stati vaghi sul concetto di elemento... e lo resteremo). In particolare, un insieme potrebbe a sua volta essere visto come elemento di un altro insieme (un anno è un insieme di mesi, ogni mese un insieme di giorni, ogni giorno un insieme di ore,...).

Vedremo gli insiemi di insiemi più in dettaglio nella Sezione 1.5 e, per il momento, ci limitiamo ad osservare che la nostra definizione di insieme conduce a *paradossi* (inconsistenze logiche), come evidenziato da Bertrand Russell nel 1902.

Esempio 1.1.4 (il paradosso di Russel)**.** *Esistono alcuni insiemi che contengono se stessi: ad esempio, l'insieme degli insiemi non vuoti*

$$NV = \{X \mid X \neq \varnothing\}$$

non è vuoto e quindi

$$NV \in NV.$$

È dunque naturale chiedersi se un certo insieme X appartiene a se stesso ($X \in X$) oppure no ($X \notin X$). Si può quindi definire l'insieme di tutti gli insiemi che contengono se stessi:

$$CS = \{X \mid X \in X\}$$

In modo del tutto analogo si può definire l'insieme di tutti gli insiemi che non *contengono se stessi:*

$$NCS = \{X \mid X \notin X\}$$

Il paradosso arriva nel momento in cui ci chiediamo: NCS appartiene a se stesso?

$$NCS \in NCS \ ?$$

[2] Si noti che $1 \notin \mathbb{P}$: 1 non è un numero primo perchè ha un solo divisore, se stesso. Il "primo" numero primo è 2.

[3] Una definizione precisa di *anno bisestile* è la seguente: a è un anno bisestile se $a > 1584$ e a è divisibile per 4 ma non per 100, oppure è divisibile per 400. Quindi in particolare $2000 \in AB$, ma $1400, 1900 \notin AB$.

- *se $NCS \notin NCS$ allora l'insieme NCS gode della proprietà che caratterizza tutti gli elementi di NCS, e quindi $NCS \in NCS$, in contrasto con l'ipotesi!*
- *se $NCS \in NCS$ allora per definizione si ha $NCS \notin NCS$ (perché tutti e soli gli elementi che soddisfano quella proprietà appartengono a NCS), di nuovo in contrasto con l'ipotesi!*

Una trattazione approfondita che eviti tali paradossi esula dagli obiettivi di queste note: accenniamo solo che esistono teorie assiomatiche degli insiemi che scongiurano la presenza di paradossi. Per semplicità in seguito faremo sempre riferimento alla definizione *intuitiva* di insieme.

1.2 Confrontare gli insiemi

Spesso incontreremo durate il corso la necessità di dover dimostrare che due insiemi $S_1 = \{ x \mid \mathcal{P}_1(x) \}$ e $S_2 = \{ x \mid \mathcal{P}_2(x) \}$ definiti in maniera diversa in realtà coincidono.

Definizione 1.2.1. *Due insiemi A e B sono* UGUALI *(scritto $A = B$) se hanno gli stessi elementi, cioè ogni elemento di A appartiene anche a B e ogni elemento di B appartiene anche ad A. Due insiemi A e B sono* DIVERSI *(scritto $A \neq B$) se non sono uguali, cioè se uno dei due insiemi contiene almeno un elemento che non appartiene all'altro.*

Una conseguenza importante di questa definizione è che due insiemi che differiscono solo per l'ordine in cui vengono elencati i loro elementi sono uguali, cioè sono lo stesso insieme. Lo stesso vale se i due insiemi differiscono solo per il numero di volte in cui un elemento compare nella loro descrizione. Per questo motivo, descrivendo gli elementi di un insieme spesso si dice che "l'ordine e la molteplicità non contano".

Esempio 1.2.2. *Si considerino gli insiemi $A = \{ \mathsf{a}, \mathsf{e}, \mathsf{i}, \mathsf{o}, \mathsf{u} \}$, $B = \{ \mathsf{o}, \mathsf{i}, \mathsf{a}, \mathsf{o}, \mathsf{i}, \mathsf{e}, \mathsf{u} \}$, e $C = \{ x \mid x$ è una vocale della lingua italiana.$\}$. Allora abbiamo $A = B$, $B = C$ e $A = C$, cioè sono tre descrizioni dello stesso insieme.*

Esempio 1.2.3.

- *L'insieme $\{ n \in \mathbb{N} \mid n > n^2 \}$ è uguale all'insieme vuoto $\varnothing$.*
- $\{ n \in \mathbb{N}^+ \mid n \geq n^2 \} = \{ 1 \}$.
- $\{ n + m \mid n \in \mathbb{N}^d, m \in \mathbb{N}^d \} = \mathbb{N}^p$
- $\{ n + m \mid n \in \mathbb{N}^p, m \in \mathbb{N}^d \} = \{ n \in \mathbb{N}^d \mid n > 1 \}$
- $\{ 1, 2, 1, 2, 1, ..., 2 \} = \{ 1, 2 \}$.

Esempio 1.2.4.

- $\mathbb{R} \neq \mathbb{Q}$ *perché, per esempio,* $\sqrt{2} \in \mathbb{R}$ *ma* $\sqrt{2} \notin \mathbb{Q}$.
- $\mathbb{Q} \neq \mathbb{Z}$ *perché, per esempio,* $\frac{1}{3} \in \mathbb{Q}$ *ma* $\frac{1}{3} \notin \mathbb{Z}$.
- $\mathbb{Z} \neq \mathbb{N}$ *perché, per esempio,* $-35 \in \mathbb{Z}$ *ma* $-35 \notin \mathbb{N}$.
- $\mathbb{N} \neq \mathbb{N}^+$ *perché* $0 \in \mathbb{N}$ *ma* $0 \notin \mathbb{N}^+$.
- $\mathbb{N}^+ \neq \varnothing$ *perché, per esempio,* $1 \in \mathbb{N}^+$ *ma* $1 \notin \varnothing$.
- $\mathbb{N}^p \neq \mathbb{N}^d$ *(perché?).*
- $\mathbb{N} \neq \{ n \mid n \in \mathbb{N}^p$ *oppure* $n \in \mathbb{N}^d \}$ *(perché?).*

Definizione 1.2.5. *A è* SOTTOINSIEME *di B, scritto $A \subseteq B$, se ogni elemento di A è anche elemento di B. A è* SOTTOINSIEME PROPRIO *di B, scritto $A \subset B$, se $A \subseteq B$ e $A \neq B$. Due insiemi A e B sono* DISGIUNTI *se non hanno elementi in comune.*

Dalle definizioni appena introdotte deriva che possiamo sfruttare la relazione di appartenenza ($\in$) nel modo seguente per confrontare due insiemi:

- per dimostrare che $A \subseteq B$ basta mostrare che ogni elemento a che appartiene ad A appartiene anche a B (si dimostra che se $a \in A$ allora $a \in B$)
- per dimostrare che $A = B$ si può mostrare che ogni elemento di A appartiene a B e viceversa, ogni elemento di B appartiene ad A (si dimostrano separatamente le due inclusioni $A \subseteq B$ e $B \subseteq A$)
- per dimostrare che $A \subset B$ si può mostrare che ogni elemento di A appartiene a B, ma che esiste un elemento di B che non appartiene ad A (si dimostra che $A \subseteq B$ e si fornisce almeno un elemento $b \in B$ tale che $b \notin A$)
- per dimostrare che A e B sono disgiunti si deve mostrare che ogni elemento di A non appartiene a B, e che ogni elemento di B non appartiene ad A

Esempio 1.2.6 (uguaglianze e inclusioni tra insiemi, motivate)**.**

- $\mathbb{N}^+ \subseteq \mathbb{N}$.
  Ogni naturale positivo $n \in \mathbb{N}^+$ è chiaramente un naturale, e quindi $n \in \mathbb{N}$.
- $\{\, n+m \mid n \in \mathbb{N}^d, m \in \mathbb{N}^d \,\} = \mathbb{N}^p$.
  L'insieme a sinistra nell'uguaglianza è incluso in quello a destra: se prendiamo due numeri dispari positivi n e m, la loro somma è un numero pari positivo, e quindi $n+m \in \mathbb{N}^p$. L'insieme a destra nell'uguaglianza è incluso in quello a sinistra: ogni numero pari positivo q può essere scritto come $q = (q-1)+1$, dove $q-1$ e 1 sono chiaramente numeri naturali dispari positivi, e quindi $q \in \{\, n+m \mid n \in \mathbb{N}^d, m \in \mathbb{N}^d \,\}$, ponendo $n = q-1$ e $m = 1$. (Si noti che l'uguaglianza sarebbe falsa se $\mathbb{N}^p$ contenesse lo zero.)
- *Dato un qualsiasi insieme A, vale $\varnothing \subseteq A$.*
  Infatti dovremmo mostrare che per ogni elemento a tale che $a \in \varnothing$, vale $a \in A$. Ma poiché non ci sono elementi a in $\varnothing$, non bisogna dimostrare nulla e l'inclusione vale in modo banale.

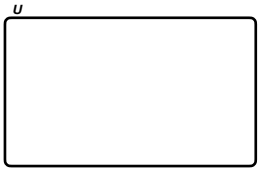
  Questa argomentazione potrebbe sembrare poco convincente a una prima lettura. Facciamo vedere, in modo equivalente, che non è vero che $\varnothing \not\subseteq A$. Infatti, per mostrare che $\varnothing \not\subseteq A$ dovremmo trovare un elemento di $\varnothing$ che non è in A. Ma dato che non ci sono elementi in $\varnothing$, questo è necessariamente falso, e quindi $\varnothing \subseteq A$.
- *Dati tre insiemi A, B e C, se $A \subset B$ e $B \subseteq C$, allora $A \subset C$.*
  Dobbiamo mostrare che (1) per ogni $a \in A$ vale $a \in C$, e che (2) esiste $c \in C$ tale che $c \notin A$. Infatti, per (1), se $a \in A$ allora $a \in B$ (visto che vale $A \subset B$), e quindi $a \in C$ (visto che vale $B \subseteq C$). D'altra parte, per (2), visto che vale $A \subset B$, sappiamo che esiste un elemento $b \in B$ tale che $b \notin A$; ma poiché vale $B \subseteq C$ sappiamo che $b \in C$ e quindi scegliamo $c = b$.

Diagrammi di Eulero-Venn

I diagrammi di Eulero-Venn sono un utile strumento per facilitare il ragionamento sulle relazioni tra insiemi e sulle operazioni tra insiemi, con una notazione grafica e intuitiva. In generale, se ne consiglia l'uso per verificare rapidamente se una certa relazione tra insiemi sia da ritenersi valida oppure no prima di procedere con una dimostrazione formale. L'uso di questi diagrammi è anche utile per identificare opportuni controesempi quando una presunta relazione tra insiemi risulti falsa.

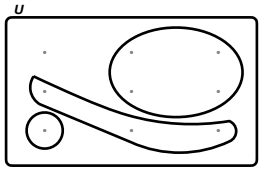
La notazione grafica di Eulero-Venn si basa sui seguenti principi:

- L'insieme universo $\mathcal{U}$ che contiene tutti gli elementi che intendiamo considerare viene rappresentato da un rettangolo.

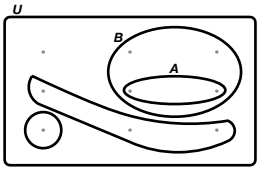


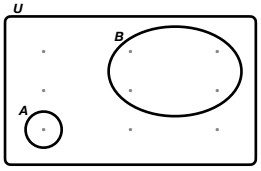
- Gli elementi vengono rappresentati come punti.

- Dentro il rettangolo usiamo circonferenze, ovali e altre forme per rappresentare gli insiemi.



- Se A è incluso in B allora la forma che rappresenta l'insieme A viene disegnata all'interno della forma che rappresenta B, come nel prossimo diagramma a sinistra. Se A e B sono disgiunti, le due forme corrispondenti non si intersecano, come del diagramma a destra.

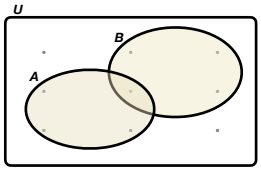




1.3 Operazioni su insiemi

A partire da alcuni insiemi possiamo definirne altri selezionando elementi comuni oppure che compaiono in un insieme ma non nell'altro, oppure in uno qualsiasi degli insiemi.

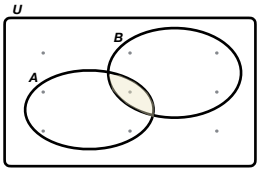
Definizione 1.3.1. *L'UNIONE di due insiemi A e B, denotata $A \cup B$, è l'insieme che contiene (tutti e soli) gli elementi che sono elementi di A oppure di B (o anche di entrambi). In formule, $A \cup B = \{\, x \mid x \in A$ oppure $x \in B \,\}$.*



Esempio 1.3.2.

- $\{\, 1, 2, 3 \,\} \cup \{\, 1, 3, 5 \,\} = \{\, 1, 2, 3, 5 \,\}$
- $\mathbb{N}^p \cup \mathbb{N}^d = \mathbb{N}^+$

Definizione 1.3.3. *L'INTERSEZIONE di due insiemi A e B, denotata $A \cap B$, è l'insieme che contiene (tutti e soli) gli elementi che appartengono sia ad A che a B. In formule, $A \cap B = \{\, x \mid x \in A$ e $x \in B \,\}$.*



Esempio 1.3.4.

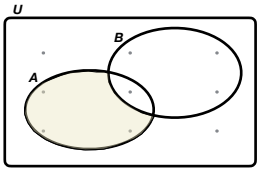
- $\{1,2,3\} \cap \{1,3,5\} = \{1,3\}$
- $\mathbb{N}^p \cap \mathbb{N}^d = \varnothing$

Esercizio 1.3.5. *Se sappiamo che $A \subseteq B$, cosa possiamo dire circa $A \cup B$ e $A \cap B$?*

Esempio 1.3.6.

- $\mathbb{N} \cup \mathbb{Z} = \mathbb{Z}$
- $\mathbb{N} \cap \mathbb{Z} = \mathbb{N}$

Definizione 1.3.7. *La **differenza** di un insieme A con un insieme B, scritta $A \setminus B$, è l'insieme che contiene (tutti e soli) gli elementi di A che non stanno in B. In formule, $A \setminus B = \{ x \mid x \in A \text{ e } x \notin B \}$.*



Esempio 1.3.8.

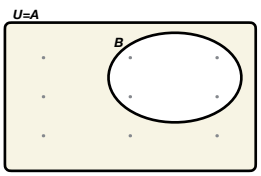
- $\mathbb{Z} \setminus \mathbb{N} = \{ -n \mid n \in \mathbb{N}^+ \}$
- $\mathbb{N} \setminus \mathbb{Z} = \varnothing$ *(si noti che in generale $A \setminus B \neq B \setminus A$)*
- $\mathbb{N} \setminus \mathbb{N}^+ = \{0\}$
- $\mathbb{N}^p \setminus \mathbb{N}^d = \mathbb{N}^p$

Definizione 1.3.9. *Nel caso particolare in cui $B \subseteq A$ si dice che $A \setminus B$ è il* COMPLEMENTO *di B rispetto a A.*

Se l'insieme di riferimento A è chiaro dal contesto (es. $A = \mathcal{U}$) allora si scrive semplicemente

$$\overline{B} = \{ x \mid x \notin B \}$$

invece di $A \setminus B$.



Esempio 1.3.10. *Assumendo come universo di riferimento l'insieme $\mathbb{N}$:*

- $\overline{\mathbb{N}^+} = \{\, 0 \,\}$
- $\overline{\mathbb{N}^p} = \{\, 0 \,\} \cup \mathbb{N}^d$
- $\overline{\mathbb{N}^d} = \{\, 0 \,\} \cup \mathbb{N}^p$
- $\overline{\mathbb{P}} = \{\, 0, 1 \,\} \cup \{\, n \mid n \text{ è un numero COMPOSTO} \,\}$
  Quindi i numeri composti *(o* compositi*) sono tutti i numeri non primi maggiori di 1.*

Operatori booleani

Nella definizione di unione, abbiamo utilizzato il termine italiano *oppure*:

$$A \cup B = \{\, x \mid x \in A \text{ oppure } x \in B \,\}$$

Nella lingua italiana, a questo termine viene spesso associato un significato esclusivo: piove *oppure* c'è il sole, rosso *oppure* verde, questo *oppure* quello... Nella teoria degli insiemi, e più in generale in Matematica e Informatica, *oppure* ha un significato non esclusivo: quando si scrive P *oppure* Q (dove P e Q sono due proprietà) significa che può valere P, oppure Q, oppure entrambe. Per denotare questo significato, in matematica si usa spesso il simbolo $\vee$ che talvolta si legge *or* (la traduzione dell'inglese di *oppure*). Se due insiemi A e B sono definiti come $A = \{x \mid P(x)\}$ e $B = \{x \mid Q(x)\}$ la loro unione può essere definita come

$$A \cup B = \{\, x \mid P(x) \ \vee \ Q(x) \,\}.$$

In modo analogo, la definizione dell'intersezione fa uso della parola italiana *e*. Anche questa parola ha un significato ben preciso nella teoria degli insiemi: P *e* Q significa che le proprietà P e Q devono valere entrambe. Per denotare questo significato, in matematica si usa spesso il simbolo $\wedge$ che talvolta si legge *and* (la traduzione dell'inglese di *e*). Utilizzando questo simbolo, l'intersezione di $A = \{x \mid P(x)\}$ e $B = \{x \mid Q(x)\}$ può essere definita come

$$A \cap B = \{\, x \mid P(x) \ \wedge \ Q(x) \,\}.$$

Infine la parola italiana *non* viene denotata dal simbolo $\neg$ che talvolta si legge *not* (traduzione dell'inglese di *no*). Anche il significato di questa parola si discosta un po' dal suo uso comune in italiano. In particolare scrivere $\neg\neg P$, per una qualche proprietà P, equivale a scrivere P, mentre in italiano la doppia negazione non sempre viene interpretata come una affermazione. Il simbolo $\neg$ può essere utilizzato per definire il complemento di un insieme $B = \{x \mid Q(x)\}$:

$$\overline{B} = \{\, x \mid \neg Q(x) \,\}.$$

Talvolta per indicare $\neg P$, si utilizza il simbolo $\not P$: abbiamo già visto un esempio: $x \notin X$ significa esattamente $\neg x \in X$.

Gli operatori $\vee$, $\wedge$ and $\neg$ formano la base del Calcolo Proposizionale, un argomento che tratteremo verso la fine del corso (Capitolo 8). Fino ad allora, utilizzeremo spesso questi simboli con il significato che abbiamo appena descritto.

Gli operatori booleani sono alla base di tutte le architetture hardware e sono esprimibili in tutti i linguaggi di programmazione. Chiaramente sono utilizzati anche nella ricerca di pagine web.

Esempio 1.3.11. *Anche i motori di ricerca permettono di esprimere gli operatori booleani. Ad esempio in Google,* $\vee$ *è esprimibile con il termine* OR*,* $\wedge$ *con il termine* AND *e* $\neg$ *con il termine* $-$*.*

Nell'Esempio 1.1.3, abbiamo visto l'insieme W *delle pagine web e l'insieme* $PippoW$ *delle pagine web in cui occorre la parola "Pippo". Definiamo* $PlutoW$ *come l'insieme* $\{x \in W \mid$ *la parola "Pluto" occorre in* $x\}$*. Adesso l'unione*

$$PippoW \cup PlutoW = \{x \in W \mid \text{le parole "Pippo" o "Pluto" occorrono in } x\}$$

è il risultato che si ottiene cercando su Google

"Pippo OR Pluto".

Similmente l'intersezione

$$PippoW \cap PlutoW = \{x \in W \mid \textit{le parole ``Pippo'' e ``Pluto'' occorrono in } x\}$$

è il risultato che si ottiene cercando su Google

``Pippo AND Pluto''.

Infine il complemento

$$\overline{PippoW} = \{x \in W \mid \textit{la parola ``Pippo'' non occorre in } x\}$$

è il risultato che si ottiene cercando su Google

``- Pippo''.

Esempio 1.3.12. *Alcuni esempi che usano gli operatori booleani per definire gli insiemi:*

- $\mathbb{N}^0 = \{\, n \mid n \in \mathbb{N}^+ \vee n = 0\}$ *(interi non-negativi),*
- $\mathbb{N}^p = \{\, n \mid n \in \mathbb{N}^+ \wedge\ n$ *è divisibile per 2* $\}$ *(pari positivi),*
- $\mathbb{N}^d = \{\, n \in \mathbb{N}^+ \mid \neg\, n \in \mathbb{N}^p\}$ *(dispari positivi).*

1.4 Leggi

Componendo insiemi tramite le operazioni viste è possibile ottenere lo stesso risultato in modi diversi. Si considerino per esempio gli insiemi

$$(A \cup B) \cup C \quad \text{e} \quad (A \cup C) \cup B.$$

L'insieme di sinistra è ottenuto unendo prima A con B e poi l'insieme risultante da questa unione con C; l'insieme di destra invece, unendo prima A con C e poi con B. È facile convincersi che i due insiemi così ottenuti sono uguali, e cioè che la seguente uguaglianza è vera.

$$(A \cup B) \cup C = (A \cup C) \cup B \tag{1.1}$$

Quando ci interroghiamo sulla verità di uguaglianze come quella di sopra è solitamente opportuno specificare cosa sono gli insiemi A, B e C. Consideriamo per esempio la domanda: è vero che

$$A \cup B = A \cap B? \tag{1.2}$$

Formulata in questo modo la domanda non ha molto senso: quali insiemi A e B stiamo considerando? Due insiemi fissati oppure ci stiamo chiedendo se l'uguaglianza vale indipendentemente da quali siano i particolari insiemi A e B (cioè deve valere *per tutti* i possibili A e B)? L'uguaglianza (1.2) risulta vera se prendiamo per esempio $A = B = \{\, 1\,\}$, ma risulta falsa se invece prendiamo $A = \{\, 1, 2\,\}$ e $B = \{\, 1\,\}$.

Ci sono uguaglianze che valgono solo per particolari insiemi, come l'uguaglianza (1.2), altre che invece valgono per qualunque insieme si consideri, come l'uguaglianza (1.1). Queste ultime sono ovviamente di maggiore interesse e sono dette *leggi*.

Alcune leggi sono ovvie, altre "credibili", ma non sempre è così... Consideriamo per esempio la seguente domanda: è vero che le seguenti uguaglianze valgono per ogni scelta di insiemi A, B, C, D?

$$A \cap (B \setminus C) = (A \cap B) \setminus C \tag{1.3}$$

$$(A \cup B) \setminus (B \cap A) = (A \setminus B) \cup (B \setminus A) \tag{1.4}$$

$$(A \cap B) \cup (C \cap D) = (A \cup B) \cap (C \cup D) \tag{1.5}$$

Naturalmente non possiamo verificare le uguaglianze per *tutte* le scelte degli insiemi, perché sono infinite. Quindi per rispondere positivamente a questa domanda bisogna fornirne una *prova* o *dimostrazione*, come vedremo tra poco.

Invece, per rispondere negativamente è sufficiente fornire un *controesempio*: essendo una legge valida per tutte le possibili scelte, basta trovare una scelta che la confuti. Nel nostro caso dobbiamo trovare degli insiemi per cui l'uguaglianza non vale.

Per esempio, l'uguaglianza (1.5) non vale scegliendo gli insiemi A, B, C, D come segue.

$$A = \{\,1,2\,\},\ B = \{\,2,3\,\},\ C = \{\,3,4\,\},\ D = \{\,4,5\,\}$$

Infatti $A \cap B = \{\,2\,\}$ e $C \cap D = \{\,4\,\}$, mentre $A \cup B = \{\,1,2,3\,\}$ e $C \cup D = \{\,3,4,5\,\}$, da cui

$$\{\,2\,\} \cup \{\,4\,\} = \{\,2,4\,\} \neq \{\,1,2,3\,\} \cap \{\,3,4,5\,\} = \{\,3\,\}$$

È buona prassi fornire un controesempio che sia il più "semplice" possibile. Per esempio, la seguente scelta di insiemi fornisce un controesempio più piccolo all'uguaglianza (1.5).

$$A = \{\,1\,\},\ B = \varnothing,\ C = \{\,1\,\},\ D = \varnothing$$

Possiamo adesso elencare alcune leggi fondamentali per gli insiemi.

Teorema 1.4.1. *Per tutti gli insiemi A, B, C (nell'universo $\mathcal{U}$) valgono le uguaglianze nelle Tabelle 1.1, 1.2 e 1.3.*

associatività	$A \cup (B \cup C) = (A \cup B) \cup C$	$A \cap (B \cap C) = (A \cap B) \cap C$
unità	$A \cup \varnothing = A$	$A \cap \mathcal{U} = A$
commutatività	$A \cup B = B \cup A$	$A \cap B = B \cap A$
idempotenza	$A \cup A = A$	$A \cap A = A$
assorbimento	$A \cup \mathcal{U} = \mathcal{U}$	$A \cap \varnothing = \varnothing$

Tabella 1.1: Leggi per $\cup$ e $\cap$

distributività di $\cup$ su $\cap$	$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$
distributività di $\cap$ su $\cup$	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$
assorbimento di $\cup$ su $\cap$	$A \cup (A \cap B) = A$
assorbimento di $\cap$ su $\cup$	$A \cap (A \cup B) = A$
complemento per $\cup$	$A \cup \overline{A} = \mathcal{U}$
complemento per $\cap$	$A \cap \overline{A} = \varnothing$

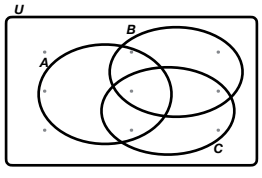
Tabella 1.2: Leggi che collegano $\cup$, $\cap$ e $\overline{(\ )}$

differenza	$A \setminus B = A \cap \overline{B}$

Tabella 1.3: Legge per $\setminus$

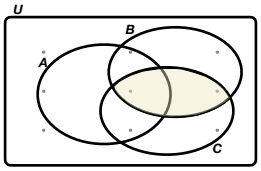
Per ognuna di queste leggi è possibile fornire una dimostrazione basata sulla definizione delle operazioni su insiemi presentate nella Sezione 1.3, o anche usando i diagrammi di Eulero-Venn. In queste note dimostriamo solamente la distributività di $\cup$ su $\cap$ (Tabella 1.2) e lasciamo la dimostrazione delle altre leggi come esercizio.

Dimostrazione di $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$. Vediamo una dimostrazione grafica utilizzando i diagrammi di Eulero-Venn. Abbiamo tre insiemi qualsiasi A, B e C.

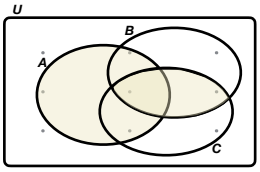


Consideriamo il membro sinistro dell'ugualianza: $A \cup (B \cap C)$. Questo insieme è ottenuto facendo prima l'intersezione di B con C e poi unendo l'insieme risultante con A. Procediamo con cautela, per piccoli passi.

Prima coloriamo l'insieme $B \cap C$:



Poi uniamo A all'insieme ottenuto:

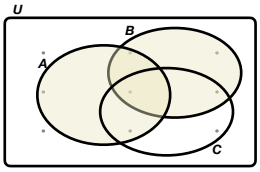


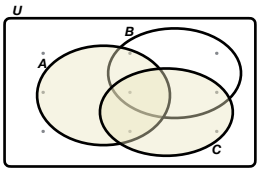
(1.6)

La colorazione del diagramma di sopra rappresenta esattamente l'insieme $A \cup (B \cap C)$.

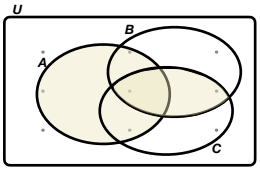
Consideriamo adesso il membro destro dell'uguaglianza: $(A \cup B) \cap (A \cup C)$. Questo insieme è ottenuto facendo prima l'unione di A con B e di A con C e poi prendendo l'intersezione degli insiemi risultanti.

Prima coloriamo gli insiemi $A \cup B$ (sinistra) e $A \cup C$ (destra):





Poi prendiamo la loro intersezione:



(1.7)

La colorazione del diagramma di sopra rappresenta esattamente l'insieme $(A \cup B) \cap (A \cup C)$.

Osservando che la colorazione dei diagrammi (1.6) e (1.7) è la stessa, possiamo concludere che $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$. ■

Esercizio 1.4.2. *Dimostrare le uguaglianza nelle Tabelle 1.1, 1.2 e 1.3 utilizzando i diagrammi di Eulero-Venn.*

Osservazione 1.4.3 (Parentesi)**.** *Le leggi nelle Tabelle 1.1 e 1.2, così come molte altre espressioni che vedremo durante il corso, fanno uso delle parentesi tonde (e). Spieghiamo adesso, una volta per tutte, il loro significato esatto, che è stato già anticipato in modo approssimativo nella dimostrazione di $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$ e all'inizio di questa sezione, prima dell'equazione* (1.1)*.*

Le parentesi tonde specificano l'ordine delle operazioni all'interno di una espressione: le operazioni racchiuse dentro le parentesi tonde vengono eseguite prima delle operazioni all'esterno. Per esempio l'espressione $A \cup (B \cup C)$ indica l'insieme ottenuto facendo prima l'unione di B con C e poi unendo il risultato con A; invece l'espressione $(A \cup B) \cup C$ indica l'insieme ottenuto facendo prima l'unione di A con B e poi con C.

Si noti che questo uso delle parentesi è lo stesso di quello che si fa con le comuni espressioni aritmetiche. Per esempio, $x \times (y + z)$ indica che prima si deve effettuare la somma di y con z e poi moltiplicare il risultato per x.

Osservazione 1.4.4 (Efficienza)**.** *Quando gli insiemi sono molto grandi, come ad esempio PippoW e PlutoW, fare le operazioni di unione e intersezione può essere molto costoso in termini di tempo e risorse di calcolo (ad esempio memoria ed energia). È quindi conveniente fare il minor numero di operazioni possibile.*

Alcune delle leggi che abbiamo introdotto in questa lezione sono molto utili a questo scopo. Ad esempio, la distributività ci permette di ridurre un calcolo che coinvolge tre operazioni (nel membro destro dell'uguaglianza) ad un calcolo che coinvolge solo due operazioni (membro sinistro).

Esercizio 1.4.5. *Alcune delle leggi nelle Tabelle 1.1 e 1.2 valgono anche per gli operatori aritmetici. Se sostituiamo $+$ al posto di $\cup$, $\times$ al posto di $\cap$, 0 al posto di $\varnothing$ e 1 al posto di $\mathcal{U}$, quali uguaglianze della Tabella 1.1 valgono per tutti i numeri naturali A, B e C?*

Esercizio 1.4.6. *È vero che per tutti gli insiemi A, B, C vale che $(A \cup B) \cap C = A \cup (B \cap C)$? In caso affermativo fornire una dimostrazione; in caso negativo fornire un controesempio.*

Dimostrazioni per sostituzione

Per dimostrare il Teorema 1.4.1, abbiamo presentato una dimostrazione grafica utilizzando i diagrammi di Eulero-Venn. In molti casi questi diagrammi permettono di presentare prove *visive*, facili da seguire e convincenti anche per chi non abbia basi matematiche. Però quando le leggi coinvolgono più di tre insiemi, le dimostrazioni con i diagrammi diventano più complicate e meno leggibili.

Un'alternativa per dimostrare la validità di una legge, consiste nell'effettuare una *dimostrazione per sostituzione* utilizzando altre leggi dimostrate in precedenza. Vediamo subito come dimostrare per sosituzione l'uguaglianza in (1.1).

Esempio 1.4.7. *Dimostriamo per sostituzione che per ogni A, B, C, vale che $(A \cup B) \cup C = A \cup (C \cup B)$. A tal fine è sufficiente osservare la seguente derivazione (una derivazione è una sequenza di uguaglianze giustificate da una legge dimostrata in precedenza).*

$$\begin{aligned} (A \cup B) \cup C &= A \cup (B \cup C) && \textit{(associatività)} \\ &= A \cup (C \cup B) && \textit{(commutatività)} \end{aligned}$$

Utilizzando le leggi nelle Tabelle 1.1, 1.2 e 1.3, è possibile dimostrare per sostituzione alcune leggi molto utili.

Teorema 1.4.8. *Per tutti gli insiemi A, B, C (nell'universo $\mathcal{U}$) valgono le uguaglianze nelle Tabella 1.4.*

complemento-1	$A \cup (\overline{A} \cap B) = A \cup B$	$A \cap (\overline{A} \cup B) = A \cap B$
complemento-2	$\overline{A} \cup (A \cap B) = \overline{A} \cup B$	$\overline{A} \cap (A \cup B) = \overline{A} \cap B$
convoluzione	$\overline{(\overline{A})} = A$	
De Morgan	$\overline{A \cup B} = \overline{A} \cap \overline{B}$	$\overline{A \cap B} = \overline{A} \cup \overline{B}$
$\mathcal{U} : \varnothing$	$\overline{\varnothing} = \mathcal{U}$	$\overline{\mathcal{U}} = \varnothing$

Tabella 1.4: Altre leggi importanti

Come per il Teorema 1.4.1 mostriamo la prova solamente di alcune leggi e lasciamo le rimanenti per esercizio

Dimostrazione di $A \cup (\overline{A} \cap B) = A \cup B$.

$$\begin{aligned} A \cup (\overline{A} \cap B) &= (A \cup \overline{A}) \cap (A \cup B) && \text{(distributività)} \\ &= \mathcal{U} \cap (A \cup B) && \text{(complemento per } \cup\text{)} \end{aligned}$$

$$= A \cup B \qquad \text{(unità)}$$

■

Esercizio 1.4.9. *Dimostrare per sostituzione che per tutti gli insiemi A, B, C vale che $\overline{A} \cup (A \cap B) = \overline{A} \cup B$ (complemento-2, nella Tabella 1.4).*

Dimostrazione di $\overline{(\overline{A})} = A$.

$$\begin{aligned}
A &= A \cup \varnothing && \text{(unità)}\\
&= A \cup (\overline{A} \cap \overline{(\overline{A})}) && \text{(complemento per } \cap)\\
&= A \cup \overline{(\overline{A})} && \text{(complemento-1)}\\
&= \overline{(\overline{A})} \cup A && \text{(commutatività)}\\
&= \overline{(\overline{A})} \cup (A \cap \overline{A}) && \text{(complemento-2)}\\
&= \overline{(\overline{A})} \cup \varnothing && \text{(distributività)}\\
&= \overline{(\overline{A})} && \text{(distributività)}
\end{aligned}$$

■

Esercizio 1.4.10. *Dimostrare per sostituzione le leggi di De Morgan e $\mathcal{U} : \varnothing$ nella Tabella 1.4.*

Proprietà di uguaglianza e inclusione

Nelle dimostrazioni per sostituzione abbiamo utilizzato tacitamente alcune banali proprietà dell'uguaglianza: per esempio, nella derivazione dell'Esempio 1.4.7, abbiamo scritto l'uguaglianza

$$(A \cup B) \cup C = A \cup (B \cup C)$$

giustificata dalla legge di associatività, che è

$$A \cup (B \cup C) = (A \cup B) \cup C.$$

Si noti che le espressioni a destra ed a sinistra dell'uguaglianza sono invertite. Questo non comporta alcun problema perchè ogni volta che vale $S = D$ allora vale anche $D = S$. Questa proprietà è detta *simmetria* di $=$. È importante osservare che $\subseteq$ non gode di questa proprietà: se vale che $S \subseteq D$ non necessariamente vale $D \subseteq S$.

Esercizio 1.4.11. *Per quali insiemi A e B valgono contemporaneamente $A \subseteq B$ e $B \subseteq A$?*

Un'altra proprietà dell'uguaglianza che utilizziamo sempre, ma implicitamente, nelle dimostrazioni per sostituzione è la *transitività*: sapendo che $A = B$ e che $B = C$ si può concludere che $A = C$. Si consideri per esempio una generica derivazione che utilizza tre leggi:

$$\begin{aligned}
E_0 &= E_1 && \text{(legge-1)}\\
&= E_2 && \text{(legge-2)}\\
&= E_3 && \text{(legge-3)}
\end{aligned}$$

Questa derivazione mostra tre uguaglianze: $E_0 = E_1$, $E_1 = E_2$ ed $E_2 = E_3$. Per concludere che $E_0 = E_3$ si deve ricorrere alla transitività due volte: (1) sapendo che $E_0 = E_1$ e $E_1 = E_2$ si conclude che $E_0 = E_2$; (2) sapendo che $E_0 = E_2$ e $E_2 = E_3$ si conclude che $E_0 = E_3$.

Il seguente risultato elenca le ovvie proprietà dell'uguaglianza.

Proposizione 1.4.12. *Per tutti gli insiemi A, B, C vale che:*

- $A = A$ *(riflessività);*
- *Se* $A = B$ *e* $B = C$*, allora* $A = C$ *(transitività);*
- *Se* $A = B$*, allora* $B = A$ *(simmetria).*

Queste proprietà sono ovvie, nel senso che seguono immediatamente dalla definizione di uguaglianza e quindi non necessitano di una dimostrazione.

Il seguente risultato elenca le ovvie proprietà dell'inclusione.

Proposizione 1.4.13. *Per tutti gli insiemi* A, B, C *vale che:*

- $A \subseteq A$ *(riflessività);*
- *Se* $A \subseteq B$ *e* $B \subseteq C$*, allora* $A \subseteq C$ *(transitività);*
- *Se* $A \subseteq B$ *e* $B \subseteq A$*, allora* $A = B$ *(anti-simmetria);*

Anche in questo caso, le proprietà seguono immediatamente dalla definizione di inclusione.

È importante notare che mentre per l'uguaglianza vale la simmetria, per l'inclusione vale l'anti-simmetria. Utilizzeremo frequentemente questa proprietà nelle dimostrazioni discorsive.

Dimostrazioni discorsive

Le dimostrazioni per sostituzione sono estremamente formali e convincenti, ma hanno lo svantaggio di essere talvolta molto lunghe e difficili da scoprire. Per esempio la dimostrazione per sostituzione di $\overline{(\overline{A})} = A$ che abbiamo visto prima è molto sofisticata, ma dimostrare questa legge con i digarammi di Eulero-Venn è quasi immediato.

Esercizio 1.4.14. *Dimostrare con i diagrammi di Eulero-Venn che per tutti gli insiemi* A *vale che* $\overline{(\overline{A})} = A$*.*

Le dimostrazioni che probabilmente siete abituati a conoscere alternano la notazione matematica (non ambigua, universale) con frasi in linguaggio naturale (ambiguo, deve essere tradotto a seconda della nazionalità del lettore).

Sono quelle più comuni nei libri e cercano di rendere leggibili e meno tediose le prove puramente formali, guidando il lettore ma saltando alcuni passaggi logici. Sono comunque fondate su prove puramente formali, non ambigue, universalmente leggibili e verificabili.

Uno degli scopi del corso sarà quello di conciliare queste due visioni per poter costruire argomentazioni solide, per presentarle ad altri e per verificare quelle di altri.

Adesso mostriamo una prova discorsiva della legge di distributività di $\cup$ su $\cap$ (che abbiamo già provato con i diagrammi di Eulero-Venn).

Esempio 1.4.15. *Vogliamo dimostrare che per tutti gli insiemi* A, B, C *vale che:*

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

Anziché dimostrare l'uguaglianza direttamente possiamo dimostrare separatamente le inclusioni

$$A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C) \quad e \quad A \cup (B \cap C) \supseteq (A \cup B) \cap (A \cup C)$$

e poi concludere l'uguaglianza utilizzando l'antisimmetria di $\subseteq$ *(Proposizione 1.4.13).*

- ***Prima inclusione*** *(*$A \cup (B \cap C) \subseteq (A \cup B) \cap (A \cup C)$*)*

  Prendiamo un qualsiasi elemento $x \in A \cup (B \cap C)$*.*

  Vogliamo fare vedere che $x \in (A \cup B) \cap (A \cup C)$*.*

  Dalla definizione di unione possiamo distinguere due possibilità, che consideriamo separatamente:

  $$\text{(i) } x \in A \qquad \text{(ii) } x \in B \cap C$$

  (Si noti che questi due casi non sono necessariamente disgiunti tra loro.)

  In entrambi i casi bisogna arrivare a dimostrare che $x \in (A \cup B) \cap (A \cup C)$*.*

– *caso (i)*
Siccome $x \in A$, per definizione di unione si ha che $x \in A \cup B$ e che $x \in A \cup C$.
Ma allora, visto che x appartiene a entrambi questi insiemi deve appartenere anche alla loro intersezione, ovvero $x \in (A \cup B) \cap (A \cup C)$ (per definizione di intersezione).

– *caso (ii)*
Siccome $x \in B \cap C$, per definizione di intersezione si ha che $x \in B$ e che $x \in C$.
Dato che $x \in B$, per definizione di unione si ha che $x \in A \cup B$.
Analogamente, dato che $x \in C$, per definizione di unione si ha che $x \in A \cup C$.
Ma allora, visto che x appartiene a entrambi questi insiemi deve appartenere anche alla loro intersezione, ovvero $x \in (A \cup B) \cap (A \cup C)$ (per definizione di intersezione).

Non ci restano altri casi da considerare e quindi la prima inclusione è dimostrata.

- ***Seconda inclusione*** *($(A \cup B) \cap (A \cup C) \subseteq A \cup (B \cap C)$)*
Prendiamo un qualsiasi elemento $y \in (A \cup B) \cap (A \cup C)$.
Vogliamo fare vedere che $y \in A \cup (B \cap C)$.
Dato che $y \in (A \cup B) \cap (A \cup C)$, dalla definizione di intersezione sappiamo che $y \in A \cup B$ e $y \in A \cup C$.
A questo punto possiamo distinguere due possibilità, che consideriamo separatamente:[4]

$$(a)\ y \in A \qquad\qquad (b)\ y \notin A$$

(Si noti che questi due casi sono disgiunti tra loro.)
In entrambi i casi bisogna arrivare a dimostrare che $y \in A \cup (B \cap C)$.

– *caso (a)*
Se $y \in A$ allora per definizione di unione $y \in A \cup (B \cap C)$ e abbiamo finito.

– *caso (b)*
Dato che $y \in A \cup B$, se $y \notin A$, allora necessariamente $y \in B$.
Analogamente, dato che $y \in A \cup C$, se $y \notin A$, allora $y \in C$.
Ma allora, visto che y appartiene a entrambi questi insiemi deve appartenere anche alla loro intersezione, ovvero $y \in B \cap C$ (per definizione di intersezione).
Per definizione di unione, $y \in A \cup (B \cap C)$ e abbiamo finito.

Non ci restano altri casi da considerare e quindi anche la seconda inclusione è dimostrata.

Esercizio 1.4.16. *Dimostrare in modo discorsivo che per tutti gli insiemi A, B, vale che $A \cup (A \cap B) = A$ (legge di assorbimento).*

Esempio 1.4.17. *Dimostriamo in maniera discorsiva che per tutti gli insiemi $\overline{(A \cup B)} = \overline{A} \cap \overline{B}$ (legge di De Morgan nella Tabella 1.4).*
Come prima dimostriamo separatamente le inclusioni

$$\overline{(A \cup B)} \subseteq \overline{A} \cap \overline{B} \quad e \quad \overline{A} \cap \overline{B} \subseteq \overline{(A \cup B)}$$

e poi concludiamo l'uguaglianza utilizzando l'antisimmetria di $\subseteq$ (Proposizione 1.4.13).

- *Per prima cosa facciamo vedere che $\overline{(A \cup B)} \subseteq \overline{A} \cap \overline{B}$.*
Preso un qualsiasi $x \in \overline{(A \cup B)}$, dobbiamo mostrare che $x \in \overline{A} \cap \overline{B}$.
Per definizione di complemento, se $x \in \overline{(A \cup B)}$ vuol dire che $x \notin A \cup B$. Quindi, per definizione di unione, si ha che $x \notin A$ e $x \notin B$.

[4] Sapendo che $y \in A \cup B$ e $y \in A \cup C$ ci sono quattro combinazioni possibili: (1) $y \in A$ e $y \in A$, (2) $y \in A$ e $y \in B$, (3) $y \in C$ e $y \in A$, (4) $y \in B$ e $y \in C$. Le prime tre sono sussunte da $y \in A$, la quarta da $y \notin A$.

Dato che $x \notin A$, per definizione di complemento si ha $x \in \overline{A}$.

Analogamente, dato che $x \notin B$, per definizione di complemento si ha $x \in \overline{B}$.

Ma allora $x \in \overline{A} \cap \overline{B}$ e abbiamo finito.

- *Resta da far vedere che $\overline{A} \cap \overline{B} \subseteq \overline{(A \cup B)}$.*

  Prendiamo un qualsiasi elemento $y \in \overline{A} \cap \overline{B}$. Dobbiamo far vedere che $y \in \overline{(A \cup B)}$.

  Dato che $y \in \overline{A} \cap \overline{B}$, per definizione di intersezione si ha che $y \in \overline{A}$ e $y \in \overline{B}$.

  Dunque, per definizione di complemento, si ha $y \notin A$ e $y \notin B$.

  Ma se y non appartiene a A e non appartiene a B allora non può certo appartenere alla loro unione, quindi $y \notin A \cup B$.

  Per definizione di complemento, questo vuol dire che $y \in \overline{(A \cup B)}$ e abbiamo finito.

Si noti che le dimostrazioni dei due casi sono composte dagli stessi passaggi ma in ordine rovesciato: in questi casi non occorre dimostrare separatamente le due inclusioni, ma basta una sequenza di "se-e-solo-se".

1.5 Insiemi di insiemi

Alcuni problemi richiedono di investigare tutte le possibili combinazioni degli elementi di un insieme. In questi casi è utile trattare insiemi i cui elementi siano a loro volta altri insiemi, come per esempio:

$$\{\,\{2,4,6,8\}\,,\,\{1,3,5,7,9\}\,\} \qquad \text{e} \qquad \{\,\{a\}\,,\,\{a,b\}\,,\,\{a,b,c\}\,\}$$

È importante notare che

$$\{a\} \in \{\,\{a\}\,,\,\{a,b\}\,,\,\{a,b,c\}\,\}$$

mentre invece

$$a \notin \{\,\{a\}\,,\,\{a,b\}\,,\,\{a,b,c\}\,\}\,.$$

Infatti a è un elemento, mentre $\{a\}$ è l'insieme che contiene solamente l'elemento a. Similmente $\{a\} \neq \{\{a\}\}$: il primo è l'insieme che contiene l'elemento a, mentre il secondo è l'insieme che contiene come elemento l'insieme $\{a\}$.

Osservazione 1.5.1. *Si deve fare estrema attenzione a usare la notazione in modo corretto, specialmente per quanto riguarda l'uso delle parentesi graffe.*

Possiamo adesso introdurre una costruzione che giocherà un ruolo fondamentale durante tutto il corso.

Definizione 1.5.2. *Dato un insieme A, il suo* INSIEME DELLE PARTI *$\mathcal{P}(A)$ è quell'insieme che ha come elementi tutti e soli i sottoinsiemi di A. In formule, $\mathcal{P}(A) = \{\, X \mid X \subseteq A \,\}$.*

Esempio 1.5.3. *Dato $A = \{\, 0, 1, 2 \,\}$ definire $\mathcal{P}(A)$ per enumerazione.*

$$\mathcal{P}(A) = \{\, \varnothing,\, \{0\},\, \{1\},\, \{2\},\, \{0,1\},\, \{0,2\},\, \{1,2\},\, A \,\}$$

Vediamo un semplice fatto che riguarda l'insieme delle parti.

Proposizione 1.5.4. *Per tutti gli insiemi A vale che:*

$$\varnothing \in \mathcal{P}(A) \qquad e \qquad A \in \mathcal{P}(A)$$

Dimostrazione. Dimostrare che $\varnothing \in \mathcal{P}(A)$ è, per definitizione di $\mathcal{P}(A)$, equivalente a dimostrare che $\varnothing \subseteq A$. Per definizione di $\subseteq$, si deve dimostrare che tutti gli elementi di $\varnothing$ sono anche elementi di A. Ma questo è banalmente vero dal momento che $\varnothing$ non contiene alcun elemento.

Dimostrare che $A \in \mathcal{P}(A)$ è, per definitizione di $\mathcal{P}(A)$, equivalente a dimostrare che $A \subseteq A$, e questo è vero per la Proposizione 1.4.13. ■

Esercizio 1.5.5. *È vero che per tutti gli insiemi A vale che $\varnothing \subset A$? In caso affermativo dare una dimostrazione. In caso negativo fornire un controesempio.*

Esempio 1.5.6.

- *Definire $\mathcal{P}(\varnothing)$ per enumerazione:*
$$\mathcal{P}(\varnothing) = \{ \varnothing \}$$
(l'unico sottoinsieme dell'insieme vuoto è... l'insieme vuoto!)
- *Definire $\mathcal{P}(\mathcal{P}(\varnothing))$ per enumerazione:*
$$\mathcal{P}(\mathcal{P}(\varnothing)) = \mathcal{P}(\{ \varnothing \}) = \{ \varnothing, \{\varnothing\} \}$$
($\mathcal{P}(\varnothing)$ contiene un solo elemento, quindi ha due possibili sottoinsiemi.)

Famiglie di insiemi

Durante il corso troveremo spesso utile considerare *famiglie* di insiemi.

Definizione 1.5.7. *Sia I un insieme tale che per ogni $i \in I$ sia definito un certo insieme A_i. L'insieme $\mathcal{F}$ che ha come elementi tutti gli insiemi A_i viene detta* FAMIGLIA INDICIZZATA DA *I. In formule, $\mathcal{F} = \{ A_i \mid i \in I \} = \{ A_i \}_{i \in I}$.*

Le operazioni di unione e intersezione sono generalizzate a famiglie di insiemi come segue:

- $\bigcup \mathcal{F} = \bigcup_{i \in I} A_i$ è l'insieme degli elementi che appartengono a A_i *per un qualche indice* $i \in I$.
- $\bigcap \mathcal{F} = \bigcap_{i \in I} A_i$ è l'insieme degli elementi che appartengono a A_i *per tutti gli indici* $i \in I$.

Esercizio 1.5.8. *Sia $I = \{ 1, 2, ..., n \}$ e*

- $A_i = \{ i \}$
- $B_i = \{ 1, 2, ..., i \}$
- $C_i = \{ i, i+1, i+2, ... \}$

Determinare:

- $\bigcup_{i \in I} A_i$ *e* $\bigcap_{i \in I} A_i$
- $\bigcup_{i \in I} B_i$ *e* $\bigcap_{i \in I} B_i$
- $\bigcup_{i \in I} C_i$ *e* $\bigcap_{i \in I} C_i$

E se invece I fosse uguale a $\mathbb{N}^+$?

Si noti che quando $I = \{ 1, 2, ..., n \}$ possiamo usare alternativamente la notazione $\bigcup_{i=1}^{n} A_i$ invece di $\bigcup_{i \in I} A_i$, e analogamente per l'intersezione.

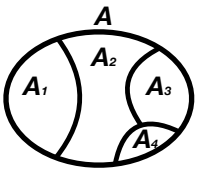
Partizioni

Quando parleremo di relazioni di equivalenza, ci interesseremo di particolari famiglie che *partizionano* gli elementi di un certo insieme A in sottoinsiemi separati.

Definizione 1.5.9. *Dato un insieme A, una* PARTIZIONE *di A è una famiglia di insiemi $\mathcal{F} = \{ A_i \}_{i \in I}$ tali che:*

- *Ogni insieme A_i è diverso da $\varnothing$ (insiemi non vuoti).*
- *$\bigcup_{i \in I} A_i = A$ (copertura di A);*
- *Presi due indici qualsiasi i e j con $i \neq j$ si ha $A_i \cap A_j = \varnothing$ (insiemi disgiunti);*

Nel seguente diagramma $\{A_1, A_2, A_3, A_4\}$ è una partizione dell'insieme A.



Esempio 1.5.10. *Un insieme di persone può essere partizionato in accordo all'anno di nascita di ciascuna persona.*

Esercizio 1.5.11. *È vero che gli insiemi $\mathbb{N}^p$ (pari positivi), e $\mathbb{N}^d$ (dispari positivi) formano una partizione di $\mathbb{N}$?*

Esercizio 1.5.12. *Sia $I = \mathbb{N}^+$ e*

- $A_i = \{\, i \,\}$
- $B_i = \{\, 1, 2, ..., i \,\}$
- $C_i = \{\, i, i+1, i+2, ... \,\}$

Dire se è vero che le seguenti famiglie di insiemi sono partizioni di $\mathbb{N}^+$:

- $\{\, B_5, A_6, C_7 \,\}$
- $\{\, A_i \,\}_{i \in I}$
- $\{\, B_i \,\}_{i \in I}$
- $\{\, C_1 \,\}$
- $\{\, (B_i \cap C_i) \,\}_{i \in I}$
- $\{\, (B_i \cap C_{i+1}) \,\}_{i \in I}$

Esercizio 1.5.13. *È possibile trovare un insieme A tale che $\mathcal{P}(A)$ sia una partizione di A? In caso positivo fornire un tale insieme, altrimenti dimostrare che non è possibile.*

Numeri naturali come insiemi

Spesso utilizzeremo i numeri naturali per denotare insiemi. Per ogni $n \in \mathbb{N}$ chiamiamo

$$n \text{ l'insieme } \{m \in \mathbb{N} \mid m < n\}$$

o, per enumerazione, $n = \{0, 1, \ldots, n-1\}$. Per esempio, abbiamo che

$$\begin{array}{rcl} 0 & = & \{\} \text{ (l'insieme vuoto)}, \\ 1 & = & \{0\}, \\ 2 & = & \{0, 1\}, \\ 3 & = & \{0, 1, 2\}, \\ \ldots & = & \ldots \end{array}$$

1.6 Prodotto cartesiano

Come detto, gli insiemi collezionano elementi in maniera *non ordinata*, però spesso è utile rappresentare collezioni ordinate del tipo:

$$(a_1, a_2, a_3, ..., a_n)$$

Per esempio per rappresentare delle stringhe o dei vettori.[5]

Ovviamente vorremmo che le coppie ordinate (a, b) e (b, a) siano ritenute diverse, mentre sappiamo che $\{ a, b \} = \{ b, a \}$.

Definizione 1.6.1. *Siano A e B due insiemi. Il* PRODOTTO CARTESIANO *di A per B, scritto $A \times B$, è l'insieme formato da tutte e sole le coppie ordinate (a, b) tali che $a \in A$ e $b \in B$. In formule, $A \times B = \{ (a, b) \mid a \in A, b \in B \}$.*

Osservazione 1.6.2. *La terminologia deriva da René Descartes (1596–1650) che dopo una vita avventurosa (giocatore, soldato, girovago, matematico-filosofo) ricevette un invito dalla Regina Christina di Svezia per divenire il suo tutore personale in filosofia, ma l'inverno del 1649–1650 fu particolarmente rigido e Descartes morì di polmonite a metà Febbraio.*

Esempio 1.6.3. *Dati $V = \{ A, 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K \}$ e $S = \{ \heartsuit, \diamondsuit, \clubsuit, \spadesuit \}$ il prodotto $V \times S$ definisce un classico mazzo di carte da gioco.*

Esempio 1.6.4. *L'insieme $\mathbb{R} \times \mathbb{R}$ è l'insieme delle coppie (x, y) tali che $x, y \in \mathbb{R}$. Molti studenti sono abituati a rappresentare gli elementi di questo insieme come i punti di un piano (solitamente chiamato piano cartesiano) in cui la posizione sull'asse delle x denota il primo elemento della coppia (in Figura 1.1, x_P) e la posizione sull'asse delle y denota il secondo elemento della coppia (in Figura 1.1, y_P).*

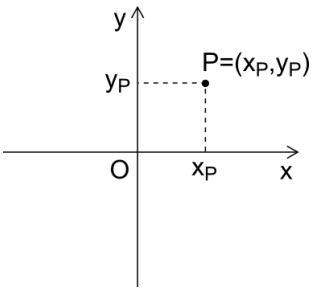


Figura 1.1: Il piano Cartesiano ($\mathbb{R} \times \mathbb{R}$) e l'elemento $P = (x_P, y_P) \in \mathbb{R} \times \mathbb{R}$.

Esempio 1.6.5. *Dati $A = \{ a, b, c \}$, $B = \{ 1, 2 \}$ e $C = \{a\}$ mostriamo gli insiemi*

$$A \times (B \times C) \text{ e } (A \times B) \times C.$$

L'insieme di sinistra è ottenuto calcolando prima

$$B \times C = \{(1, a),\ (2, a)\}$$

e poi facendo il prodotto di A per tale insieme:

$$A \times (B \times C) = \{(a, (1, a)),\ (b,\ (1, a)),\ (c, (1, a)),\ (a, (2, a)),\ (b, (2, a)),\ (c,\ (2, a))\}$$

L'insieme di destra è ottenuto calcolando prima

$$A \times B = \{(a, 1),\ (b, 1),\ (c, 1),\ (a, 2),\ (b, 2),\ (c, 2)\}$$

[5] Talvolta si usano le parentesi angolari $\langle a, b \rangle$ per rappresentare delle collezioni ordinate, invece delle tonde... ma noi non lo faremo.

e poi facendo il prodotto di tale insieme per C:

$$(A \times B) \times C = \{((a,1),a),\ ((b,1),a),\ ((c,1),a),\ ((a,2),a),\ ((b,2),a),\ ((c,2),a)\}$$

È importante notare che i due insiemi sono molto simili, ma diversi: gli elementi dell'insieme di sinistra (ad esempio $(a,(1,a))$) sono coppie in cui il secondo elemento è esso stesso una coppia (in $B \times C$). Invece, gli ementi dell'insieme di destra (ad esempio $((a,1),a)$) sono coppie in cui il primo elemento è esso stesso una coppia (in $A \times B$). Si ha pertanto che il prodotto cartesiano non è associativo, cioè

$$A \times (B \times C) \neq (A \times B) \times C$$

Nel prossimo capitolo vedremo che tali insiemi sono in biiezione.

Esercizio 1.6.6. *Dati $A = \{a,b,c\}$ e $B = \{1,2\}$ calcolare $A \times B$ e $B \times A$. (Notare che il prodotto non è commutativo, ovvero $A \times B \neq B \times A$).*

Esercizio 1.6.7. *Esistono due insiemi A e B tali che $A \times B = B \times A$?*

Esercizio 1.6.8. *Esiste un insieme A tale che $A \times A = A$?*

Esercizio 1.6.9. *Calcolare $A \times \varnothing$.*

Puzzle 1.6.10. Dire quali altre coppie comprende l'insieme

$$\{(2,1),\ (4,2),\ (6,3),\ (8,4),\ (10,5),\ (12,6),\ ...\}$$

E se invece consideriamo l'insieme riportato sotto?

$$\{(2,3),\ (4,7),\ (6,3),\ (8,4),\ (10,5),\ (12,6),\ ...\}$$

Proposizione 1.6.11. *Per tutti gli insiemi A, B vale che*

- $|\mathcal{P}(A)| = 2^{|A|}$;
- $|A \times B| = |A| \times |B|$

Il modo migliore per dimostrare la proposizione precedente è attraverso l'induzione, un argomento che vedremo nel Capitolo 5

1.7 Esercizi

Esercizio 1.7.1. *Elencare gli elementi che appartengono ai seguenti insiemi:*

1. $\{x \in \mathbb{R} \mid x^2 = 1\}$
2. $\{x \in \mathbb{N} \mid x^2 = 2\}$
3. $\{x \in \mathbb{N} \mid x < 100$ *e* x *è il quadrato di un qualche numero intero*$\}$

Esercizio 1.7.2. *Per ognuna delle seguenti coppie di insiemi, dire se $A = B$ o se $A \neq B$:*

1. $A = \{1,3,3,3,5,5,5,5,5\}$ *e* $B = \{5,3,1\}$
2. $A = \{\}$ *e* $B = \{\varnothing\}$
3. $A = \{1,\{1\}\}$ *e* $B = \mathcal{P}(\{1\})$

Esercizio 1.7.3. *Dire quali delle seguenti affermazioni sono vere e quali false:*

1. $a \in \{a\}$
2. $\{a\} \subseteq \{a\}$
3. $\{a\} \in \{a\}$
4. $\{a\} \in \{\{a\}\}$

5. $\varnothing \subseteq \{\, a \,\}$
6. $\varnothing \in \{\, a \,\}$

Esercizio 1.7.4. *Determinare l'insieme delle parti di ciascuno dei seguenti insiemi:*

1. $\{\, a \,\}$
2. $\{\, a, b \,\}$
3. $\{\, \varnothing, \{\varnothing\} \,\}$
4. $\{\, a, b, c, d \,\}$

Esercizio 1.7.5. *Dati $A = \{\, a, b, c \,\}$ e $B = \{\, a, d \,\}$ calcolare i seguenti insiemi:*

1. $A \times B$
2. $B \times A$
3. $A \cup B$
4. $A \cap B$
5. $A \setminus B$
6. $B \setminus A$

Esercizio 1.7.6. *Trovare un insieme A tale che $A \cup \mathbb{N}^+ = \mathbb{N}$, $\mathbb{N}^p \cap A = \varnothing$ e $A \cap \mathbb{P} = \{\, 11 \,\}$. Quanti insiemi esistono che soddisfano queste condizioni?*

Esercizio 1.7.7. *Tre categorie di utenti sono soci di un centro sportivo. Sapreste dire quanti sono complessivamente i soci se vi dico che:*

- *44 giocano a Tennis (T)*
- *26 Nuotano (N)*
- *31 giocano a Golf (G)*

La risposta non è 101! Non avete abbastanza elementi per rispondere perché le tre categorie non sono necessariamente disgiunte. Fatemi aggiungere che:

- *12 si dedicano al Tennis e Nuotano*
- *5 al Tennis e al Golf*
- *6 al Nuoto e al Golf*
- *4 a tutti e tre gli sport*

Quanti sono in tutto i soci? (risolvere usando diagrammi di Eulero-Venn)

Esercizio 1.7.8. *Dimostrare (per sostituzione) che per ogni A, B, C:*

1. $A \setminus \overline{A} = A$
2. $(A \setminus B) \setminus C = (A \setminus C) \setminus B$
3. $\overline{A \setminus B} = \overline{A} \cup B$
4. $\overline{A \cup (\overline{B} \cap C)} = (\overline{C} \cap \overline{A}) \cup (\overline{A} \cap B)$
5. $(A \cup \overline{B}) \cap C = (C \cap A) \cup (C \setminus B)$

Esercizio 1.7.9. *È vero che per tutti gli insiemi A, B, C, D, vale che $(A \cap D) \setminus (B \cup C) = \varnothing$? In caso affermativo fornire una dimostrazione; in caso negativo un controesempio.*

Esercizio 1.7.10. *Dimostrare il Teorema 1.4.8 usando i diagrammi di Eulero-Venn.*

Esercizio 1.7.11. *Elencare tutte le possbili partizioni dell'insieme $A = \{a, b, c\}$.*

CAPITOLO 2

Relazioni

In questo capitolo esaminiamo la nozione di *relazione*. Mostreremo svariate operazioni su relazioni e le leggi algebriche che le regolano. Considereremo poi le seguenti quattro proprietà

TOTALE	SURGETTIVA
UNIVALENTE	INIETTIVA

che sono necessarie per definire i concetti di *funzione* e *biiezione*. Alla fine del capitolo, utilizzeremo questi concetti per introdurre alcune strutture dati fondamentali in Informatica: le n-uple, le sequenze di lunghezza fissata (array) e le sequenze di lunghezza arbitraria (stringhe).

2.1 Nozioni di Base

Definizione 2.1.1. *Una* RELAZIONE R *tra l'insieme* A *e l'insieme* B *è un sottoinsieme del prodotto cartesiano* $A \times B$*, quindi* $R \subseteq A \times B$*.*

Data una relazione $R \subseteq A \times B$ e due elementi $a \in A$ e $b \in B$, se $(a, b) \in R$ diremo che a *è in relazione* R *con* b*.*

Esempio 2.1.2.

- *Se* $A = \{x, y\}$ *e* $B = \{a, b, c\}$*, il sottoinsieme*

$$\{(x, a), (x, c)\} \subseteq A \times B \tag{2.1}$$

  è una relazione.

- *Per tutti gli insiemi* A *e* B*,* $A \times B \subseteq A \times B$ *è una relazione. Questa viene chiamata* RELAZIONE COMPLETA*.*

- *Per tutti gli insiemi* A *e* B*,* $\varnothing \subseteq A \times B$ *è una relazione. Questa viene chiamata* RELAZIONE VUOTA*, e viene denotata con* $\varnothing_{A,B}$

In una relazione $R \subseteq A \times B$, A è detto INSIEME DI PARTENZA e B è detto INSIEME DI ARRIVO. Talvolta, l'insieme di partenza e l'insieme di arrivo possono coincidere: durante il corso vedremo molte relazioni di questo tipo, che sono anche argomento del Capitolo 3. Illustriamo come esempi prima le relazioni di parentela tra esseri umani e poi la relazione identità su un qualsiasi insieme A.

Esempio 2.1.3. *Sia* U *l'insieme di tutti gli esseri umani, e consideriamo le seguenti relazioni di parentela:*

- $Madre = \{(x, y) \in U \times U \mid x \text{ è madre di } y\}$,
- $Padre = \{(x, y) \in U \times U \mid x \text{ è padre di } y\}$,
- $Figlia = \{(x, y) \in U \times U \mid x \text{ è figlia di } y\}$,
- $Figlio = \{(x, y) \in U \times U \mid x \text{ è figlio di } y\}$.

A partire da queste quattro relazioni costruiremo nel seguito tutte le comuni relazioni di parentela, tipo sorella, fratello, zia, ...

Osservazione 2.1.4. *Gli autori di queste note credono nella libertà di ogni essere umano di determinare e definire la propria sessualità, i propri affetti e le proprie relazioni familiari. Rigettano quindi ogni forma di discriminazione sessista ed etero-normativa. Tuttavia in queste note le relazioni familiari considerate saranno spesso riferite a quelle della famiglia così detta* tradizionale*. Tali relazioni infatti sono degli esempi molto convenienti delle varie proprietà che desideriamo illustrare.*

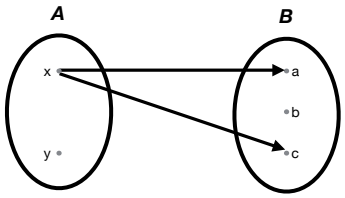
Esempio 2.1.5. *Per tutti gli insiemi A,*

$$\{(x, x) \mid x \in A\} \subseteq A \times A$$

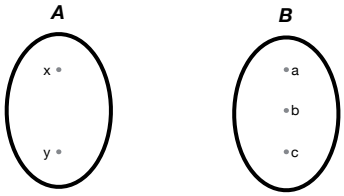
è una relazione. Questa viene chiamata RELAZIONE IDENTITÀ *su A, e viene denotata $Id_A \subseteq A \times A$.*

Come gli insiemi, anche le relazioni possono essere comodamente visualizzate attraverso dei diagrammi: data una relazione $R \subseteq A \times B$ questa viene visualizzata disegnando l'insieme A sulla sinistra, l'insieme B sulla destra e inserendo una freccia dall'elemento a all'elemento b per ogni coppia $(a, b) \in R$.

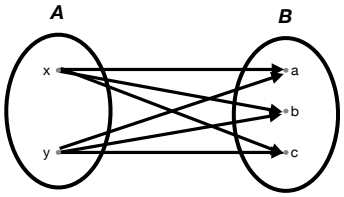
Esempio 2.1.6. *Prendiamo $A = \{x, y\}$ e $B = \{a, b, c\}$ come nell'Esempio 2.1.2. La relazione in* (2.1) *viene disegnata come*



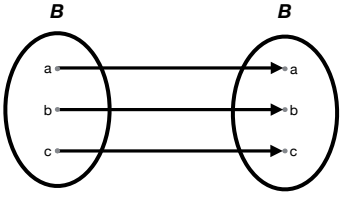
mentre la relazione vuota $\varnothing \subseteq A \times B$ come



e la relazione completa $A \times B \subseteq A \times B$ come



Invece, la relazione identità su B, $Id_B \subseteq B \times B$ è disegnata come



Esercizio 2.1.7. *Sia $A = \{x, y, z\}$ e $B = \{x, y, z, u\}$. Disegnare la relazione $\{(x, x),\ (y, y),\ (z, z)\} \subseteq A \times B$.*

Quando le relazioni sono molto grandi, o infinite, diventa complicato rappresentarle attraverso diagrammi. Per esempio, per disegnare la relazione $Madre$ su tutti gli esseri umani (Esempio 2.1.3) non basterebbero tutte le pagine di queste dispense. Quando però nelle relazioni c'è una sorta di regola, come per esempio in molte relazioni matematiche si possono usare i punti $(\dots)$ per sottindere la regola.

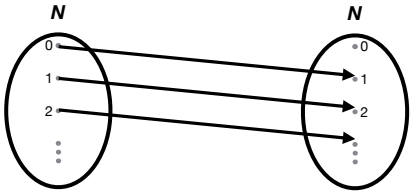
Esempio 2.1.8. *Si consideri la relazione successore*

$$Succ \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = x + 1\}$$

dove $+$ denota l'usuale operazione di somma tra numeri. Questa relazione si può anche definire per enumerazione come

$$Succ \subseteq \mathbb{N} \times \mathbb{N} = \{(0, 1),\ (1, 2),\ (2, 3),\ \dots\}$$

e rappresentare con il seguente diagramma.



Molto spesso l'insieme di partenza è il prodotto cartesiano di due insiemi, cioè $A = A_1 \times A_2$ per qualche insieme A_1, A_2.

Esempio 2.1.9. *Si consideri la relazione*

$$Plus \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N} = \{((x, y), z) \mid z = x + y\}$$

dove $+$ denota l'usuale operazione di somma tra numeri. In questo caso l'insieme di partenza è $\mathbb{N} \times \mathbb{N}$, mentre l'insieme di arrivo è $\mathbb{N}$.

Questa relazione si può anche definire per enumerazione come

$$\begin{aligned} Plus \subseteq \mathbb{N} \times \mathbb{N} = \{&((0,0),0),\ ((1,0),1),\ ((2,0),2),\ \dots \\ &((0,1),1),\ ((1,1),2),\ ((2,1),3),\ \dots \\ &\dots \\ \} \end{aligned}$$

2.2 Operazioni su relazioni

Come gli insiemi, anche le relazioni possono essere combinate in vari modi per ottenere nuove relazioni. In questa sezione illustriamo le principali operazioni su relazioni.

Operazioni insiemistiche su relazioni

Dal momento che ogni relazione è essa stessa un insieme, le relazioni possono essere combinate con tutti gli operatori insiemistici.

Esempio 2.2.1. *Si ricordino le relazioni $Madre \subseteq U \times U$ e $Padre \subseteq U \times U$ dell'Esempio 2.1.3. La relazione $Genitore \subseteq U \times U$ può essere definita come*

$$Genitore = Madre \cup Padre. \tag{2.2}$$

Infatti

$$\begin{aligned} & Madre \cup Padre \\ &= \{(x, y) \in U \times U \mid x \text{ è madre di } y\} \cup \{(x, y) \in U \times U \mid x \text{ è padre di } y\} \\ &= \{(x, y) \in U \times U \mid x \text{ è madre di } y \ \vee \ x \text{ è padre di } y\} \\ &= \{(x, y) \in U \times U \mid x \text{ è madre di } y \text{ oppure } x \text{ è padre di } y\} \\ &= \{(x, y) \in U \times U \mid x \text{ è genitore di } y\} \end{aligned}$$

Esercizio 2.2.2. *Si definisca attraverso unione, la relazione*

$$Figli* = \{(x, y) \in U \times U \mid x \text{ è figlio o figlia di } y\}.$$

Quando si compongono relazioni attraverso le operazioni insiemistiche si deve però fare sempre attenzione agli insiemi di partenza e di arrivo. Per evitare ogni tipo di complicazioni, nel nostro corso consideriamo queste operazioni solo su relazioni che hanno uno stesso insieme di partenza (A) e uno stesso insieme di arrivo (B).

Definizione 2.2.3. *Siano date due relazioni $R \subseteq A \times B$ e $S \subseteq A \times B$.*

- *La relazione $R \cup S \subseteq A \times B$ è detta* UNIONE *di R e S;*
- *La relazione $R \cap S \subseteq A \times B$ è detta* INTERSEZIONE *di R e S;*
- *La relazione $R \setminus S \subseteq A \times B$ è detta* DIFFERENZA *di R con S;*
- *La relazione $A \times B \setminus R \subseteq A \times B$ è detta il* COMPLEMENTO *di R. Il complemento di una relazione $R \subseteq A \times B$ è denotato da $\overline{R}$.*

Si noti che il complemento di R è definito in maniera apparentemente diversa dal complementro insiemistico: come insieme il complemento di R è $\mathcal{U} \setminus R$, mentre come relazione è $A \times B \setminus R$. L'idea è che quando si considerano relazioni tra A e B si fissa sempre come universo l'insieme $A \times B$.

Tutte le leggi nelle Tabelle 1.1, 1.2, 1.3 e 1.4 valgono chiaramente anche per le relazioni tra A e B ma, ancora una volta, si deve prendere $\mathcal{U} = A \times B$.

Composizione

Abbiamo appena visto che quando si combinano relazioni attraverso le operazioni insiemistiche è opportuno restringersi a relazioni con gli stessi insiemi di partenza e di arrivo.

Adesso consideriamo un'operazione che permette di combinare due relazioni, quando l'insieme di arrivo della prima relazione è lo stesso dell'insieme di partenza della seconda.

Definizione 2.2.4. *Siano $R \subseteq A \times B$ e $S \subseteq B \times C$. La* COMPOSIZIONE *di R con S è la relazione $R; S \subseteq A \times C$ così definita:*

$$R; S = \{(x, z) \in A \times C \mid \text{esiste almeno un } y \in B \text{ tale che } (x, y) \in R \text{ e } (y, z) \in S\}.$$

Esempio 2.2.5. *Siano dati $A = \{x, y, z\}$, $B = \{a, b, c, d\}$ e $C = \{1, 2\}$. Consideriamo le relazioni*

$$R = \{(x, a),\ (y, b),\ (z, c),\ (z, d)\} \subseteq A \times B$$

e

$$S = \{(a, 1),\ (a, 2),\ (c, 2),\ (d, 2)\} \subseteq B \times C.$$

Le due relazioni possono essere composte in quanto l'insieme di arrivo di R è uguale all'insieme di partenza di S. La loro composizione è la relazione

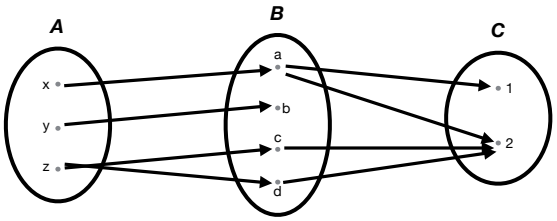
$$R; S = \{(x, 1),\ (x, 2),\ (z, 2)\} \subseteq A \times C.$$

Infatti $(x, 1) \in R; S$ perchè per l'elemento $a \in B$ si ha che $(x, a) \in R$ e $(a, 1) \in S$. Similmente, $(x, 2) \in R; S$ perchè per l'elemento $a \in B$ valgono sia $(x, a) \in R$ che $(a, 2) \in S$.

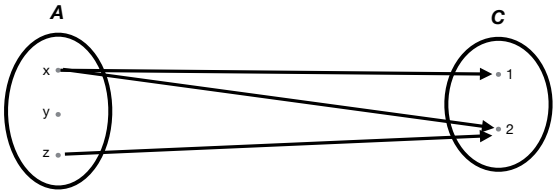
Per vedere che $(z, 2) \in R; S$ abbiamo due possibili giustificazioni: (1) per l'elemento $c \in B$ si ha che $(z, c) \in R$ e $(c, 2) \in S$; (2) per l'elemento $d \in B$ si ha che $(z, d) \in R$ e $(d, 2) \in S$.

Per vedere che nessun'altra coppia $(u, w) \in A \times C$ appartiene a $R; S$ si deve mostrare che non esiste alcun elemento $v \in B$ tale che $(u, v) \in R$ e $(v, w) \in S$. Per esempio, per vedere che $(y, 1) \notin R; S$ si può osservare che: per $a \in B$, $(y, a) \notin R$; per $b \in B$, $(b, 1) \notin S$; per $c \in B$, $(y, c) \notin R$; e per $d \in B$, $(y, d) \notin R$.

L'operazione di composizione di relazioni può sembrare un po' complicata, ma in realtà risulta estremamente naturale quando si visualizza attraverso diagrammi. Per visualizzare la composizione di $R \subseteq A \times B$ e $S \subseteq B \times C$, prima di tutto si disegnano gli insiemi A (a sinistra), B (al centro) e C (a destra), nonché le relazioni R e S, utlizzando la solita modalità delle frecce. Per esempio il seguente diagramma



visualizza le relazioni R e S dell'Esempio 2.2.5. Poi per ottenere il diagramma della relazione composta è sufficiente seguire le frecce da sinistra a destra e vedere per ogni elemento a di A e ogni elemento c di C se c'è un "percorso" da a a c. In tal caso, si aggiunge una freccia da a a c. Una volta fatto ciò per tutte le coppie $(a, c) \in A \times C$ si cancella l'insieme B dal centro e tutte le frecce ad esso collegate. Il seguente diagramma visualizza la relazione $R; S$ dell'Esempio 2.2.5.



.

Esercizio 2.2.6. *Si consideri la relazione $Succ \subseteq \mathbb{N} \times \mathbb{N}$. Si disegni il diagramma della relazione $Succ; Succ \subseteq \mathbb{N} \times \mathbb{N}$. Definire prima per enumerazione e poi per proprietà tale relazione.*

Abbiamo finora mostrato degli esempi di composizione su entità piuttosto astratte. In realtà l'operazione di composizione di relazioni ci permette di formalizzare concetti della vita quotidiana.

Esempio 2.2.7. *Si ricordi la relazione $Padre \subseteq U \times U$ dell'Esempio 2.1.3 e la relazione $Genitore \subseteq U \times U$ dell'Esempio 2.2.1. La relazione $Nonno \subseteq U \times U$ può essere definita come*

$$Nonno = Padre; Genitore.$$

Esercizio 2.2.8. *Definire per proprietà a parole la relazione $Genitore; Genitore$.*

Esercizio 2.2.9. *Definire attraverso composizione la relazione $Nonna = \{(x, y) \in U \times U \mid x$ è nonna di $y\}$.*

Esercizio 2.2.10. *Definire utilizzando solamente la composizione la relazione $Nonn * Patern* = \{(x, y) \in U \times U \mid x$ è nonno paterno o nonna paterna di $y\}$.*

I quantificatori

Nella definizione dell'operazione di composizione (Definizione 2.2.4) abbiamo utilizzato l'espressione italiana *esiste almeno*:

$$\{(x, z) \in A \times C \mid \text{esiste almeno un } y \in B \text{ tale che } (x, y) \in R \text{ e } (y, z) \in S\}$$

Questa espressione riveste un ruolo speciale nel linguaggio matematico e viene denotata dal simbolo $\exists$, chiamato QUANTIFICATORE ESISTENZIALE. In generale la formula

$$\exists a \in A \,.\, P(a)$$

si legge

esiste almeno un elemento di A tale che la proprietà P è vera.

Pertanto la composizione di relazioni può essere scritta in formule come

$$R;S = \{(x,z) \in A \times C \mid \exists y \in B\,.\,(x,y) \in R \wedge (y,z) \in S\}.$$

Un altro simbolo di fondamentale importanza è $\forall$, chiamato QUANTIFICATORE UNIVERSALE, che si legge in italiano *per ogni*. In generale la formula

$$\forall a \in A\,.\,P(a)$$

si legge

per tutti gli elementi di A vale che la proprietà P è vera.

Per dimostrare che la formula $\exists a \in A\,.\,P(a)$ è vera è sufficiente esibire un elemento $a \in A$ per cui vale la proprietà P. Dimostrare che la formula $\forall a \in A\,.\,P(a)$ è vera è molto più laborioso in generale: infatti si deve dimostrare che P vale per tutti gli $a \in A$. È importante notare che in questo caso, quando A è vuoto, non c'è niente da dimostrare: quindi la formula $\forall a \in A\,.\,P(a)$ è banalmente vera.

I quantificatori $\exists$ e $\forall$ saranno studiati in profondità nel Capitolo 8 dove presenteremo la logica del primo ordine. Fino ad allora, utilizzeremo di tanto in tanto questi simboli con il significato che abbiamo appena descritto.

Relazione opposta

Definizione 2.2.11. *Sia $R \subseteq A \times B$ una relazione. La* RELAZIONE OPPOSTA *di R è la relazione $R^{op} \subseteq B \times A$ definita come*

$$R^{op} = \{(y,x) \in B \times A \mid (x,y) \in R\}.$$

Esempio 2.2.12. *Si considerino le relazioni R e S dell'Esempio 2.2.5. Si ha che*

$$R^{op} = \{(a,x),\ (b,y),\ (c,z),\ (d,z)\} \subseteq B \times A$$

e

$$S^{op} = \{(1,a),\ (2,a),\ (2,c),\ (2,d)\} \subseteq C \times B.$$

È importante notare che R^{op} non può essere composta con S^{op} perché l'insieme di arrivo di R^{op} è A, mentre quello di partenza di S^{op} è C. Comunque si possono comporre le relazioni nel senso inverso, cioè come $S^{op};R^{op}$ perché sia l'insieme di arrivo di S^{op} che quello di partenza di R^{op} sono l'insieme B.

Esercizio 2.2.13. *Si considerino le relazioni R, S ed $R;S$ nell'Esempio 2.2.5 e le relazioni R^{op} e S^{op} nell'Esempio 2.2.12. È vera la seguente uguaglianza?*

$$(R;S)^{op} = S^{op};R^{op}$$

Esempio 2.2.14. *Si consideri la relazione $Genitore \subseteq U \times U$ introdotta nell'Esempio 2.2.1. Si ha che*

$$Genitore^{op} = Figli*$$

dove $Figli*$ è la relazione dell'Esempio 2.2.2. Infatti

Si ha che

$$\begin{aligned} Genitore^{op} &= \{(y,x) \in U \times U \mid (x,y) \in Genitore\} \\ &= \{(y,x) \in U \times U \mid x \text{ è madre o padre di } y\} \\ &= \{(y,x) \in U \times U \mid y \text{ è figlio o figlia di } x\} \\ &= \{(x,y) \in U \times U \mid x \text{ è figlio o figlia di } y\} \\ &= Figli* \end{aligned}$$

Esercizio 2.2.15. *È vero che*

$$Figli*^{op} = Genitore?$$

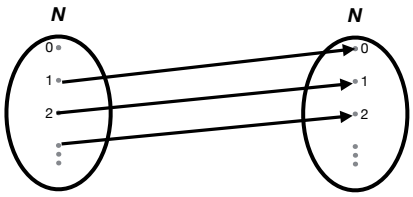
Esempio 2.2.16. *Si consideri la relazione*

$$Succ \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = x + 1\}$$

introdotta nell'Esempio 2.1.8. Si ha che

$$\begin{aligned} Succ^{op} &= \{(y, x) \in \mathbb{N} \times \mathbb{N} \mid (x, y) \in Succ\} \\ &= \{(y, x) \in \mathbb{N} \times \mathbb{N} \mid y = x + 1\} \\ &= \{(y, x) \in \mathbb{N} \times \mathbb{N} \mid y - 1 = x\} \\ &= \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid x - 1 = y\} \\ &= \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = x - 1\} \end{aligned}$$

Chiamiamo questa relazione Pred che sta per "predecessore".



È importante osservare come si può ottenere il diagramma per $Succ^{op}$ è a partire da quello di *Succ*: dato il diagramma di una qualsiasi relazione R, il diagramma di R^{op} è ottenuto come l'immagine allo specchio (cioè girandolo di 180 gradi) e invertendo la direzione delle frecce.

Concludiamo con un po' di esempi ben noti dalla matematica.

Esempio 2.2.17.

$$Double \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = 2 \times x\}$$

$$Double^{op} = Half \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = x/2\}$$

Esercizio 2.2.18. *Disegnare il diagramma delle relazioni* $Double \subseteq \mathbb{N}\times\mathbb{N}$ *e* $Double^{op} = Half \subseteq \mathbb{N}\times\mathbb{N}$

Esempio 2.2.19.

$$Square \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = x^2\}$$

$$Square^{op} = Sqrt \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = \sqrt{x}\}$$

Esercizio 2.2.20. *Disegnare il diagramma delle relazioni* $Square \subseteq \mathbb{N}\times\mathbb{N}$ *e* $Square^{op} = Sqrt \subseteq \mathbb{N}\times\mathbb{N}$

Esempio 2.2.21.

$$Exp \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = 2^x\}$$

$$Exp^{op} = Log \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = log_2(x)\}$$

Esercizio 2.2.22. *Disegnare il diagramma delle relazioni* $Exp \subseteq \mathbb{N} \times \mathbb{N}$ *e* $Exp^{op} = Log \subseteq \mathbb{N} \times \mathbb{N}$

Esercizio 2.2.23. *Definire attraverso le relazioni e le operazioni viste finora, la relazione* $Fratello = \{(x, y) \in U \times U \mid x \text{ è fratello di } y\}$. *Per "fratello" si intende il figlio di almeno uno dei genitori. Per esempio, due uomini sono considerati fratelli, se hanno la stessa madre ma non lo stesso padre.*

Esercizio 2.2.24. *Definire attraverso le relazioni e le operazioni viste finora, la relazione* $Sorella = \{(x, y) \in U \times U \mid x \text{ è fratello di } y\}$. *Anche qui per "sorella" si intende la figlia di almeno uno dei genitori.*

Esercizio 2.2.25. *Definire attraverso le relazioni e le operazioni viste finora, le relazioni* $Zia = \{(x, y) \in U \times U \mid x \text{ è zia di } y\}$ *e* $Zio = \{(x, y) \in U \times U \mid x \text{ è zio di } y\}$.

2.3 Leggi

Come per gli insiemi, anche per le relazioni esistono delle utili leggi che regolano il comportamento delle varie operazioni. Abbiamo già anticipato che, per le operazioni insiemistiche, valgono le stesse leggi studiate nella Sezione 1.4.

Teorema 2.3.1. *Per tutti gli insiemi A, B e relazioni $R \subseteq A \times B$, $S \subseteq A \times B$ e $T \subseteq A \times B$ valgono le uguaglianze nelle Tabelle 2.1, 2.2 e 2.3.*

associatività	$A \cup (S \cup T) = (A \cup S) \cup T$	$R \cap (S \cap T) = (R \cap S) \cap T$
unità	$R \cup \varnothing = R$	$R \cap (A \times B) = R$
commutatività	$R \cup S = S \cup R$	$R \cap S = S \cap R$
idempotenza	$R \cup R = R$	$R \cap R = R$
assorbimento	$R \cup (A \times B) = (A \times B)$	$R \cap \varnothing = \varnothing$

Tabella 2.1: Leggi per le operazioni insiemistiche

distributività di $\cup$ su $\cap$	$R \cup (S \cap T) = (R \cup S) \cap (R \cup T)$
distributività di $\cap$ su $\cup$	$R \cap (S \cup T) = (R \cap S) \cup (R \cap T)$
assorbimento di $\cup$ su $\cap$	$R \cup (R \cap S) = R$
assorbimento di $\cap$ su $\cup$	$R \cap (R \cup S) = R$
complemento per $\cup$	$R \cup \overline{R} = (A \times B)$
complemento per $\cap$	$R \cap \overline{R} = \varnothing$

Tabella 2.2: Leggi per le operazioni insiemistiche (2)

differenza	$R \setminus S = R \cap \overline{S}$

Tabella 2.3: Leggi per le operazioni insiemistiche (3)

La validità del Teorema 2.3.1 è garantita dai risultati della Sezione 1.4, che valgono per tutti gli insiemi e quindi, in particolare, anche per tutte le relazioni.

Il seguente teorema illustra invece alcune utili leggi per la composizione e la relazione opposta.

Teorema 2.3.2. *Per tutti gli insiemi A, B, C, D e relazioni $R \subseteq A \times B$, $S \subseteq B \times C$ e $T \subseteq C \times D$ valgono le uguaglianze nelle Tabelle 2.4 e 2.5.*

associatività	$R; (S; T) = (R; S); T$
unità	$Id_A; R = R = R; Id_B$
assorbimento	$R; \varnothing_{B,C} = \varnothing_{A,C} = \varnothing_{A,B}; S$

Tabella 2.4: Leggi per composizione di relazioni

La notazione grafica che abbiamo introdotto per le relazioni può fornire un'intuizione chiara riguardo alla validità di tali leggi. In queste note forniamo una dimostrazione discorsiva per l'associatività della composizione e lasciamo le dimostrazioni delle altre leggi come esercizio.

Dimostrazione di $R; (S; T) = (R; S); T$. Anziché dimostrare l'uguaglianza direttamente possiamo dimostrare separatamente le inclusioni

$$R; (S; T) \subseteq (R; S); T \quad \text{e} \quad R; (S; T) \supseteq (R; S); T$$

e poi concludere l'uguaglianza utilizzando l'antisimmetria di $\subseteq$ (Proposizione 1.4.13).

convoluzione	$(R^{op})^{op} = R$
op-id	$Id_A^{op} = Id_A$
op-compl	$(A \times B)^{op} = B \times A$
op-vuoto	$\varnothing_{A,B}^{op} = \varnothing_{B,A}$

Tabella 2.5: Leggi per relazione opposta

- $R;(S;T) \subseteq (R;S);T$. Per dimostrare questa inclusione, dobbiamo dimostrare che preso un generico elemento di $R;(S;T)$, tale elemento appartiene anche a $(R;S);T$. Sia $(a,d) \in R;(S;T)$: per definizione della composizione (;) esiste almeno un $b \in B$ tale che (1) $(a,b) \in R$ e (2) $(b,d) \in S;T$. Da (2) e dalla definizione di composizione si ha che esiste almeno un $c \in C$ tale che (2.1) $(b,c) \in S$ e (2.2) $(c,d) \in T$. Da (1) e (2.1) segue che (3) $(a,c) \in R;S$. Da (3) e (2.2) segue che $(a,d) \in (R;S);T$.
- $R;(S;T) \supseteq (R;S);T$. Per dimostrare questa inclusione, dobbiamo dimostrare che preso un generico elemento di $(R;S);T$, tale elemento appartiene anche a $R;(S;T)$. La dimostrazione è del tutto analoga a quella dell'inclusione precedente.

■

Esercizio 2.3.3. *Dimostrare in modo discorsivo le leggi nelle Tabelle 2.4 e 2.5.*

L'ultimo teorema illustra le leggi che regolano le interazioni fra le operazioni insiemistiche, la composizione e la relazione opposta.

Teorema 2.3.4. *Per tutti gli insiemi A, B, C, D e relazioni $R \subseteq A \times B$, $S, T \subseteq B \times C$ e $U \subseteq C \times D$ valgono le uguaglianze nelle Tabelle 2.6.*

distributività di ; su $\cup$ (sinistra)	$R;(S \cup T) = (R;S) \cup (R;T)$
distributività di ; su $\cup$ (destra)	$(S \cup T);U = (S;U) \cup (T;U)$
distributività di $\cdot^{op}$ su ;	$(R;S)^{op} = S^{op};R^{op}$
distributività di $\cdot^{op}$ su $\cup$	$(S \cup T)^{op} = S^{op} \cup T^{op}$
distributività di $\cdot^{op}$ su $\cap$	$(S \cap T)^{op} = S^{op} \cap T^{op}$
distributività di $\cdot^{op}$ su $\overline{\cdot}$	$(\overline{R})^{op} = \overline{(R^{op})}$

Tabella 2.6: Leggi di distributività

Ancora una volta, forniamo la dimostrazione di una sola delle leggi e lasciamo le altre come esercizio.

Dimostrazione di $(R;S)^{op} = S^{op};R^{op}$. Anziché dimostrare l'uguaglianza direttamente possiamo dimostrare separatamente le inclusioni

$$(R;S)^{op} \subseteq S^{op};R^{op} \quad \text{e} \quad (R;S)^{op} \supseteq S^{op};R^{op}$$

e poi concludere l'uguaglianza utilizzando l'antisimmetria di $\subseteq$ (Proposizione 1.4.13).

- $(R;S)^{op} \subseteq S^{op};R^{op}$. Per dimostrare questa inclusione, dobbiamo dimostrare che preso un generico elemento di $(R;S)^{op}$, tale elemento appartiene anche a $S^{op};R^{op}$. Sia $(c,a) \in (R;S)^{op}$: per definizione della relazione opposta si ha che $(a,c) \in R;S$. Quindi, Per definizione di composizione, esiste almeno un $b \in B$ tale che (1) $(a,b) \in R$ e (2) $(b,c) \in S$.

  Utilizzando la definizione di relazione opposta, da (1) si deduce che $(b,a) \in R^{op}$ e, da (2) che $(c,b) \in S^{op}$. Per definizione di composizione, $(c,a) \in S^{op};R^{op}$.
- $(R;S)^{op} \supseteq S^{op};R^{op}$. Per dimostrare questa inclusione, dobbiamo dimostrare che preso un generico elemento di $S^{op};R^{op}$, tale elemento appartiene anche a $(R;S)^{op}$. Sia $(c,a) \in S^{op};R^{op}$. Per definizione di composizone si ha che esiste almeno un $b \in B$ tale che (1) $(c,b) \in S^{op}$ e (2)

$(b, a) \in R^{op}$. Utilizzando la definizione di relazione opposta da (1) si deduce che $(b, c) \in S$ e da (2) che $(a, b) \in R$.

Pertanto, per definizione di composizione, $(a, c) \in R; S$ e quindi $(c, a) \in (R; S)^{op}$.

■

Esercizio 2.3.5. *Dimostrare in modo discorsivo le leggi nelle Tabella 2.6.*

Esercizio 2.3.6. *Dimostrare per sostituzione che*

$$Genitore; Genitore = (Madre; Madre) \cup (Madre; Padre) \cup (Padre; Madre) \cup (Madre; Madre)$$

Esercizio 2.3.7. *Dimostrare in modo discorsivo che per tutti gli insiemi A, B, C e relazioni $R \subseteq A \times B$ e $S \subseteq B \times C$ e $T \subseteq B \times C$ vale che*

$$\text{se } S \subseteq T, \text{ allora } R; S \subseteq R; T.$$

Esercizio 2.3.8. *Considera la seguente affermazione: per tutti gli insiemi A, B, C e relazioni $R \subseteq A \times B$ e $S \subseteq B \times C$ e $T \subseteq B \times C$ vale che*

$$\text{se } R; S \subseteq R; T \text{ allora } S \subseteq T.$$

Se è vera fornire una dimostrazione, se è falsa fornire un controesempio.

Esercizio 2.3.9. *Dimostrare in modo discorsivo che per tutti gli insiemi A, B e relazioni $R \subseteq A \times B$ e $S \subseteq A \times B$ vale che*

$$R \subseteq S \text{ se e solo se } R^{op} \subseteq S^{op}.$$

2.4 Proprietà di relazioni

In questa sezione illustriamo quattro fondamentali proprietà di relazioni. Queste proprietà appaiono ovunque, sia in Informatica che in Matematica e, nel resto del nostro corso, ci permetteranno di definire importanti concetti, quali funzioni e biiezioni.

Definizione 2.4.1. *Siano A e B due insiemi e R una relazione tra A e B. $R \subseteq A \times B$ si dice* TOTALE *se per tutti gli $a \in A$,* esiste almeno *un $b \in B$ tale che $(a, b) \in R$.*

Esempio 2.4.2. *Siano dati $A = \{x, y, z\}$ e $B = \{a, b, c, d\}$. La relazione*

$$R \subseteq A \times B = \{(x, a),\ (x, b),\ (y, a),\ (z, d)\}$$

è totale, mentre la relazione

$$S \subseteq A \times B = \{(x, c),\ (x, d),\ (z, d)\}$$

non è totale perchè per l'elemento $y \in A$ non esiste alcun elemento $u \in B$ tale che $(y, u) \in S$.

Esempio 2.4.3. *Per tutti gli insiemi A e B*

- *la relazione identità $Id_A \subseteq A \times A$ è totale;*
- *la relazione completa $A \times B \subseteq A \times B$ è totale.*

Esempio 2.4.4. *Per gli insiemi $A = \{x, y, z\}$ e $B = \{a, b, c, d\}$, la relazione vuota $\varnothing \subseteq A \times B$ non è totale. Per vederlo, basta osservare che per $x \in A$ non esiste alcun $u \in B$ tale che $(x, u) \in \varnothing$.*

Esercizio 2.4.5. *Esiste un insieme A per il quale la relazione vuota $\varnothing \subseteq A \times B$ è totale. Quale insieme è?*

Esercizio 2.4.6. *La relazione $Succ \subseteq \mathbb{N} \times \mathbb{N}$ è totale? La relazione $Pred \subseteq \mathbb{N} \times \mathbb{N}$? La relazione $Plus \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$?*

Esercizio 2.4.7. *La relazione $Madre \subseteq U \times U$ è totale? La relazione $Genitore \subseteq U \times U$? La relazione $Figlia \subseteq U \times U$? La relazione $Figli* \subseteq U \times U$?*

Definizione 2.4.8. *Siano A e B due insiemi e R una relazione tra A e B. $R \subseteq A \times B$ si dice* UNIVALENTE *se per tutti gli $a \in A$,* esiste al più *un $b \in B$ tale che $(a, b) \in R$.*

Esempio 2.4.9. *Sia $A = \{x, y, z\}$ e $B = \{a, b, c, d\}$. La relazione*

$$R \subseteq A \times B = \{(x, a),\ (x, b),\ (y, a),\ (z, d)\}$$

non è univalente perchè per l'emento $x \in A$ esistono due elementi in B che sono in relazione R con x: infatti $(x, a) \in R$ e $(x, b) \in R$. Invece, la relazione

$$S \subseteq A \times B = \{(x, c),\ (z, d)\}$$

è univalente.

Esempio 2.4.10. *Per tutti gli insiemi A e B*

- *la relazione identità $Id_A \subseteq A \times A$ è univalente;*
- *la relazione vuota $\varnothing \subseteq A \times B$ è univalente.*

Esempio 2.4.11. *Per l'insieme $A = \{x, y, z\}$ e $B = \{a, b, c, d\}$, la relazione completa $A \times B \subseteq A \times B$ non è univalente. Per vederlo, basta osservare che per $x \in A$ si ha sia che $(x, a) \in A \times B$ che $(x, b) \in A \times B$.*

Esercizio 2.4.12. *Esiste un insieme A per il quale la relazione completa $A \times B \subseteq A \times B$ è univalente. Quale insieme è?*

Esercizio 2.4.13. *La relazione $Succ \subseteq \mathbb{N} \times \mathbb{N}$ è univalente? La relazione $Pred \subseteq \mathbb{N} \times \mathbb{N}$? La relazione $Plus \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$?*

Esercizio 2.4.14. *La relazione $Madre \subseteq U \times U$ è univalente? La relazione $(Madre)^{op}$?*

È interessante notare che l'unica differenza nelle definizioni di totale e univalente consiste nel *esiste almeno* (totale) e *esiste al più* (univalente). Per una qualche proprietà P, "esiste almeno un elemento x tale che $P(x)$" significa che il numero di elementi che soddisfano P deve essere *maggiore o uguale* ad 1. Invece "esiste al più un elemento x tale che $P(x)$" significa che il numero di elementi che soddisfano P deve essere *minore o uguale* ad 1.

È importante capire questa differenza, e cosa significa in termini di dimostrazioni: quando si desidera dimostrare che "esiste almeno un elemento x tale che $P(x)$" si deve esibire un tale x; invece quando si desidera dimostrare che "esiste al più un elemento x tale che $P(x)$" si deve mostrare che, se ci sono due elementi x e y per cui vale sia $P(x)$ che $P(y)$, allora i due elementi sono lo stesso.

Osservazione 2.4.15. *Abbiamo visto che l'espressione italiana* esiste almeno *viene denotata nel linguaggio matematico dal simbolo $\exists$. Per l'espressione* esiste al più *non c'è un simbolo corrispondente. La ragione è che* esiste al più *può essere scritta in formule utilizzando $\forall$ e $=$. Più precisamente,*

$$\textit{esiste al più un } a \in A \textit{ tale che } P(a)$$

si può scrivere come

$$\forall a \in A\,.\,\forall b \in A\,.\ \textit{se } P(a) \wedge P(b),\ \textit{allora } a = b.$$

Adesso consideriamo altre due proprietà. L'unica differenza tra le due consiste ancora nell'*almeno* e nell'*al più*, ma diversamente dalle due proprietà discusse prima, riguardano l'insieme di arrivo e non quello di partenza.

Definizione 2.4.16. *Siano A e B due insiemi e R una relazione tra A e B. $R \subseteq A \times B$ si dice* SURGETTIVA *se per tutti i $b \in B$,* esiste almeno *un $a \in A$ tale che $(a, b) \in R$.*

Esempio 2.4.17. *Sia* $A = \{x, y, z\}$ *e* $B = \{a, b, c, d\}$*. La relazione*

$$R \subseteq A \times B = \{(x, a),\ (x, b),\ (y, c),\ (z, d)\}$$

è surgettiva, mentre la relazione

$$S \subseteq A \times B = \{(x, c),\ (x, b),\ (z, d)\}$$

non è surgettiva perchè per l'elemento $a \in B$ *non esiste alcun elemento* $u \in A$ *tale che* $(u, A) \in S$*.*

Esempio 2.4.18. *Per tutti gli insiemi* A *e* B

- *la relazione identità* $Id_A \subseteq A \times A$ *è surgettiva;*
- *la relazione completa* $A \times B \subseteq A \times B$ *è surgettiva.*

Esempio 2.4.19. *Per gli insiemi* $A = \{x, y, z\}$ *e* $B = \{a, b, c, d\}$*, la relazione vuota* $\varnothing \subseteq A \times B$ *non è surgettiva. Per vederlo, basta osservare che per* $a \in B$ *non esiste alcun* $u \in A$ *tale che* $(u, a) \in \varnothing$*.*

Esercizio 2.4.20. *Esiste un insieme* B *per il quale la relazione vuota* $\varnothing \subseteq A \times B$ *è surgettiva. Quale insieme è?*

Esercizio 2.4.21. *La relazione* $Succ \subseteq \mathbb{N} \times \mathbb{N}$ *è surgettiva? La relazione* $Pred \subseteq \mathbb{N} \times \mathbb{N}$*? La relazione* $Plus \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$*?*

Esercizio 2.4.22. *La relazione* $Madre \subseteq U \times U$ *è surgettiva? La relazione* $Genitore \subseteq U \times U$*? La relazione* $Figlia \subseteq U \times U$*? La relazione* $Figli* \subseteq U \times U$*?*

Definizione 2.4.23. *Siano* A *e* B *due insiemi e* R *una relazione tra* A *e* B*.* $R \subseteq A \times B$ *si dice* INIETTIVA *se per tutti i* $b \in B$*,* esiste al più *un* $a \in A$ *tale che* $(a, b) \in R$*.*

Esempio 2.4.24. *Sia* $A = \{x, y, z\}$ *e* $B = \{a, b, c, d\}$*. La relazione*

$$R \subseteq A \times B = \{(x, a),\ (x, b),\ (y, c),\ (z, d)\}$$

è iniettiva, mentre la relazione

$$S \subseteq A \times B = \{(x, c),\ (x, b),\ (z, b)\}$$

non è iniettiva perchè per l'elemento $b \in B$ *esistono due elementi in* A *in relazione* S *con* b*:* $(x, b) \in S$ *e* $(z, b) \in S$*.*

Esempio 2.4.25. *Per tutti gli insiemi* A *e* B

- *la relazione identità* $Id_A \subseteq A \times A$ *è iniettiva;*
- *la relazione vuota* $\varnothing \subseteq A \times B$ *è iniettiva.*

Esempio 2.4.26. *Per gli insiemi* $A = \{x, y, z\}$ *e* $B = \{a, b, c, d\}$*, la relazione completa* $A \times B \subseteq A \times B$ *non è iniettiva. Per vederlo, basta osservare che per* $a \in B$*, si ha che* $(x, a) \in A \times B$ *e* $(y, a) \in A \times B$*.*

Esercizio 2.4.27. *Esiste un insieme* B *per il quale la relazione completa* $A \times B \subseteq A \times B$ *è iniettiva. Quale insieme è?*

Esercizio 2.4.28. *La relazione* $Succ \subseteq \mathbb{N} \times \mathbb{N}$ *è iniettiva? La relazione* $Pred \subseteq \mathbb{N} \times \mathbb{N}$*? La relazione* $Plus \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$*?*

Esercizio 2.4.29. *La relazione* $Madre \subseteq U \times U$ *è iniettiva? La relazione* $Genitore \subseteq U \times U$*? La relazione* $Figlia \subseteq U \times U$*? La relazione* $Figli* \subseteq U \times U$*?*

La Tabella 2.7 fornisce un modo efficace per riassumere e, soprattutto, capire le quattro proprietà: totale e univalente riguardano l'insieme di partenza, mentre surgettiva e iniettiva l'insieme di arrivo. Le proprietà totale e surgettiva richiedono l'esistenza di almeno un elemento, mentre univalente e iniettiva richiedo l'esistenza di al più un elemento.

	partenza	arrivo
almeno	TOTALE	SURGETTIVA
al più	UNIVALENTE	INIETTIVA

Tabella 2.7: Tabella riassuntiva delle quattro principali proprietà

Risultati di dualità

Confrontando le definizioni di relazione totale e surgettiva è facile osservare che queste impongono lo stesso vincolo ma la prima sull'insieme di partenza, mentre la seconda sull'insieme di arrivo. Abbiamo visto che l'operazione $\cdot^{op}$ inverte l'insieme di partenza e di arrivo. Il seguente risultato collega le due proprietà attraverso la relazione opposta.

Proposizione 2.4.30. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

2.4.30.1. *$R \subseteq A \times B$ è totale se e solo se $R^{op} \subseteq B \times A$ è surgettiva.*

2.4.30.2. *$R \subseteq A \times B$ è univalente se e solo se $R^{op} \subseteq B \times A$ è iniettiva.*

Dimostrazione. Dimostriamo il primo punto e lasciamo il secondo come esercizio. Trattandosi di un se e solo se, possiamo dividere la dimostrazioni in due parti:

- Se $R \subseteq A \times B$ è totale, allora $R^{op} \subseteq B \times A$ è surgettiva. Visto che R è totale allora

  $$\text{per ogni } x \in A \text{ esiste almeno un } y \in B \text{ tale che } (x, y) \in R.$$

  Per definizione di R^{op} questo significa che

  $$\text{per ogni } x \in A \text{ esiste almeno un } y \in B \text{ tale che } (y, x) \in R^{op}.$$

  Questo significa, per definizione, che $R^{op} \subseteq B \times A$ è surgettiva.

- Se $R^{op} \subseteq B \times A$ è surgettiva, allora $R \subseteq A \times B$ è totale. Visto che $R^{op} \subseteq B \times A$ è surgettiva allora

  $$\text{per ogni } x \in A \text{ esiste almeno un } y \in B \text{ tale che } (y, x) \in R^{op}.$$

  Per definizione di R^{op} questo significa che

  $$\text{per ogni } x \in A \text{ esiste almeno un } y \in B \text{ tale che } (x, y) \in R.$$

  Questo significa, per definizione, che $R \subseteq A \times B$ è totale.

■

Esercizio 2.4.31. *Dimostrare la Proposizione 2.4.30.2.*

Proposizione 2.4.32. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

2.4.32.1. *$R \subseteq A \times B$ è surgettiva se e solo se $R^{op} \subseteq B \times A$ è totale.*

2.4.32.2. *$R \subseteq A \times B$ è iniettiva se e solo se $R^{op} \subseteq B \times A$ è univalente.*

Dimostrazione. Dimostriamo il primo punto, il secondo lo lasciamo come esercizio. Dalla Proposizione 2.4.30.1 R^{op} è totale se e solo se $(R^{op})^{op}$ è surgettiva. Ma, visto che $(R^{op})^{op} = R$, R^{op} è totale se e solo se R è surgettiva. ■

Esercizio 2.4.33. *Dimostrare la Proposizione 2.4.32.2.*

Teorema di caratterizzazione

Le quattro proprietà fondamentali che abbiamo studiato in questa sezione possono essere *caratterizzate* attraverso le operazioni su relazioni introdotte nella Sezione 2.2.

Teorema 2.4.34. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

2.4.34.1. *R è totale se e solo se $Id_A \subseteq R; R^{op}$;*

2.4.34.2. *R è univalente se e solo se $R^{op}; R \subseteq Id_B$;*

2.4.34.3. *R è surgettiva se e solo se $Id_B \subseteq R^{op}; R$;*

2.4.34.4. *R è iniettiva se e solo se $R; R^{op} \subseteq Id_A$.*

Dimostrazione. Dimostriamo il primo punto. Trattandosi di un se e solo se, possiamo dividere la dimostrazioni in due parti:

- Se R è totale, allora $Id_A \subseteq R; R^{op}$. Assumendo che R sia totale, dobbiamo dimostrare che $Id_A \subseteq R; R^{op}$, cioè che per tutti gli $a \in A$, $(a, a) \in R; R^{op}$.

  Prendiamo un qualsiasi elemento $a \in A$. Visto che R è totale, allora sappiamo che esiste almeno un $b \in B$ tale che $(a, b) \in R$. Per definizione di R^{op} abbiamo inolre che $(b, a) \in R^{op}$. Pertanto, per definizione della composizione, si ha che $(a, a) \in R; R^{op}$.

- Se $Id_A \subseteq R; R^{op}$, allora R è totale. Assumendo che $Id_A \subseteq R; R^{op}$, dobbiamo dimostrare che R è totale, cioè che per tutti gli $a \in A$, esiste almeno un $b \in B$ tale che $(a, b) \in R$.

  Prendiamo un qualsiasi elemento $a \in A$. Visto che $Id_A \subseteq R; R^{op}$, si ha che $(a, a) \in R; R^{op}$. Per definizione di composizione, questo significa che esiste almeno un $b \in B$ tale che $(a, b) \in R$ e $(b, a) \in R^{op}$.

Dimostriamo ora il secondo punto. Trattandosi di un se e solo se, possiamo dividere la dimostrazioni in due parti:

- Se R è univalente, allora $R^{op}; R \subseteq Id_B$. Assumendo che R sia univalente, dobbiamo dimostrare che $R^{op}; R \subseteq Id_B$, cioè che per tutti gli $(x, y) \in R; R^{op}$, si ha che $(x, y) \in Id_B$, vale a dire che $x = y$.

  Prendiamo una qualsiasi coppia $(x, y) \in R^{op}; R$. Per definizione di composizione, esiste un $a \in A$ tale che $(x, a) \in R^{op}$ e $(a, y) \in R$. Visto che $(x, a) \in R^{op}$, si ha che $(a, x) \in R$. Essendo R univalente, da $(a, x) \in R$ e $(a, y) \in R$ si può concludere che $x = y$.

- Se $R^{op}; R \subseteq Id_B$, allora R è univalente. Assumendo che $R^{op}; R \subseteq Id_B$, dobbiamo dimostrare che R è univalente, cioè che per tutti gli $a \in A$, esiste al più un $b \in B$ tale che $(a, b) \in R$.

  Prendiamo un qualsiasi elemento $a \in A$ ed assumiamo che $(a, x) \in R$ e $(a, y) \in R$ per qualche $x, y \in B$. Per definizione di R^{op}, si ha che $(x, a) \in R^{op}$ e, per definizione di composizione, che $(x, y) \in R^{op}; R$. Visto che $R^{op}; R \subseteq Id_B$, si ha che $x = y$. Questo prova che R è univalente.

Dimostriamo il terzo punto, utilizzando il primo punto ed i risultati di dualità.

R è surgettiva se e solo se	$R^{op} \subseteq B \times A$ è totale	(Proposizione 2.4.32.1)
se e solo se	$Id_B \subseteq R^{op}; (R^{op})^{op}$	(primo punto)
se e solo se	$Id_B \subseteq R^{op}; R$	(convoluzione)

Lasciamo la dimostrazione del quarto punto come esercizio. ■

Esercizio 2.4.35. *Dimostrare il Teorema 2.4.34.4.*

Chiusura per composizione

Il precedente teorema ci fornisce una caratterizzazione che ci sarà utile in più occasioni. Adesso mostriamo come il teorema può essere utile per dimostrare che prese due relazioni che soddisfano una delle quattro proprietà, la loro composizione soddisfa ancora tale proprietà.

Proposizione 2.4.36. *Per tutti gli insiemi A, B, C e per tutte le relazioni $R \subseteq A \times B$, $S \subseteq B \times C$ vale che:*

2.4.36.1. *Se R e S sono totali, allora $R;S$ è totale.*

2.4.36.2. *Se R e S sono univalenti, allora $R;S$ è univalente.*

2.4.36.3. *Se R e S sono surgettive, allora $R;S$ è surgettiva.*

2.4.36.4. *Se R e S sono iniettive, allora $R;S$ è iniettiva.*

Mostriamo prima una dimostrazione per sostituzione che utilizza la caratterizzazione fornita dal Teorema 2.4.34.

Dimostrazione per sostituzione. Dimostriamo solo il primo punto e lasciamo gli altri per esercizio.

1. Se $Id_A \subseteq R;R^{op}$ e $Id_B \subseteq S;S^{op}$, allora $Id_A \subseteq (R;S);(R;S)^{op}$.

$$\begin{aligned}
Id_A &\subseteq R;R^{op} && (R \text{ totale})\\
&\subseteq R;Id_B;R^{op} && (\text{unità})\\
&\subseteq R;(S;S^{op});R^{op} && (S \text{ totale})\\
&\subseteq (R;S);(S^{op};R^{op}) && (\text{associatività})\\
&\subseteq (R;S);(R;S)^{op} && (\text{distributività})
\end{aligned}$$

■

Esercizio 2.4.37. *Dimostrare per sostituzione le Proposizioni 2.4.36.2, 2.4.36.3 e 2.4.36.4.*

Esercizio 2.4.38. *Dimostrare per sostiruzione che per tutti gli insiemi A, la relazione identità $Id_A \subseteq A \times A$ è totale, univalente, surgettiva e iniettiva.*

Esercizio 2.4.39. *Dimostrare per sostiruzione che per tutti gli insiemi A, B, la relazione vuota $\varnothing \subseteq A \times B$ è univalente e iniettiva.*

Esercizio 2.4.40. *Dimostrare per sostituzione che per tutti gli insiemi A, B, la relazione completa $A \times B \subseteq A \times B$ è totale e surgettiva.*

Mostriamo adesso una dimostrazione discorsiva del Teorema 2.4.36.

Dimostrazione discorsiva. Dimostraimo i primi tre punti. Il quarto lo lasciamo come esercizio.

1. Se R e S sono totali, allora $R;S$ è totale. Assumendo che R e S siano totale, dobbiamo dimostrare che $R;S$ è totale, cioè che per tutti gli $a \in A$, esiste almeno un $c \in C$ tale che $(a,c) \in R;S$.

   Prendiamo un qualsiasi elemento $a \in A$. Visto che R è totale, allora esiste almeno un $b \in B$ tale che $(a,b) \in R$. Visto che S è totale allora esiste almeno un elemento $c \in C$ tale che $(b,c) \in S$. Per definizione di composizione abbiamo che $(a,c) \in R;S$.

2. Se R e S sono univalenti, allora $R;S$ è univalente. Assumendo che R e S siano univalenti, dobbiamo dimostrare che $R;S$ è univalente, cioè che per tutti gli $a \in A$, esiste al più un $c \in C$ tale che $(a,c) \in R;S$.

   Prendiamo un qualsiasi elemento $a \in A$ ed assumiamo che $(a,x) \in R;S$ e $(a,y) \in R;S$ per qualche $x, y \in C$. Per la definizione di composizione, si ha che devono esiste $u, v \in B$ tali che: (1) $(a,u) \in R$, (2) $(u,x) \in S$, (3) $(a,v) \in R$ e (4) $(v,y) \in R$. Visto che R è univalente, da (1) e (3) si può dedurre che $u = v$. Visto che S è univalente, da (2) e (4) e dal fatto che $u = v$ si può dedurre che $x = y$. Pertanto $R;S$ è univalente.

3. Se R e S sono surgettive, allora $R;S$ è surgettiva. Si dimostra questo risultato utilizzando quello di sopra e i risultati di dualità. Se R e S sono surgettive, allora $R^{op} \subseteq B \times A$ e $S^{op} \subseteq C \times B$ sono totali. Pertanto, per il risultato appena dimostrato, $S^{op}; R^{op}$ è totale. Quindi, per la Proposizione 2.4.30.1 $(S^{op}; R^{op})^{op}$ è surgettiva. Si osservi che

$$\begin{aligned}(S^{op}; R^{op})^{op} &= (R^{op})^{op}; (S^{op})^{op} && \text{(distributività di } .^{op} \text{ su ;)}\\ &= R; S && \text{(convoluzione)}\end{aligned}$$

Pertanto $R;S$ è surgettiva.

■

Esercizio 2.4.41. *Dimostrare la Proposizione 2.4.36.4.*

Esercizio 2.4.42. *È vero che per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$, $S \subseteq A \times B$ vale che:*

Se R e S sono totali, allora $R \cup S$ è totale?

In caso affermativo fornire una prova. In caso negativo fornire un controesempio.

Esercizio 2.4.43. *È vero che per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$, $S \subseteq A \times B$ vale che:*

Se R e S sono totali, allora $R \cap S$ è totale?

In caso affermativo fornire una prova. In caso negativo fornire un controesempio.

Esercizio 2.4.44. *È vero che per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$, $S \subseteq B \times C$ vale che:*

Se $R;S$ è totale, allora anche R e S sono totali?

In caso affermativo fornire una prova. In caso negativo fornire un controesempio.

2.5 Funzioni

Definizione 2.5.1. *Siano A e B due insiemi ed R una relazione tra di loro. $R \subseteq A \times B$ si dice una* FUNZIONE *se R è totale e univalente.*

Esempio 2.5.2. *Sia $A = \{x, y, z\}$ e $B = \{a, b, c, d\}$. La relazione*

$$R \subseteq A \times B = \{(x, a),\ (y, a),\ (z, d)\}$$

è una funzione, mentre la relazione

$$S \subseteq A \times B = \{(x, c),\ (x, d),\ (y, d),\ (z, d)\}$$

non è una funzione perché non è univalente.

È fondamentale osservare che per definizione $R \subseteq A \times B$ è totale se e solo se

per ogni $a \in A$, esiste *almeno un* $b \in B$ tale che $(a, b) \in R$

mentre $R \subseteq A \times B$ è univalente se e solo se

per ogni $a \in A$, esiste *al più un* $b \in B$ tale che $(a, b) \in R$.

Pertanto $R \subseteq A \times B$ è una funzione se e solo se

per ogni $a \in A$, esiste *esattamente un* $b \in B$ tale che $(a, b) \in R$.

Notazione 2.5.3. *Solitamente le funzioni vegono denotate da lettere minuscole, tipicamente* $f, g, h, \ldots$ *Anziché dire una funzione* tra A e B*, come nelle relazioni, si dice una funzione* da A a B*. Anziché scrivere* $f \subseteq A \times B$*, si scrive* $f\colon A \to B$*. Inoltre si scrive* $f(a)$ *per denotare l'unico elemento* $b \in B$ *tale che* $(a, b) \in f$*. Per enfatizzare che tale elemento è* b*, si scrive* $f(a) = b$*. L'insieme di tutte le funzioni da* A *a* B *è denotato da* B^A*, cioè* $B^A = \{f\colon A \to B\}$*.*

Esempio 2.5.4. *La relazione successore* $Succ \subseteq \mathbb{N} \times \mathbb{N}$ *è una funzione, così come la relazione* $Plus \subseteq (\mathbb{N} \times \mathbb{N}) \times \mathbb{N}$ *(Esempio 2.1.9). In virtù di questo, d'ora in avanti, scriveremo* $succ\colon \mathbb{N} \to \mathbb{N}$ *e* $plus\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$*. Invece la relazione* $Pred \subseteq \mathbb{N} \times \mathbb{N}$ *non è una funzione perché non è totale.*

Esempio 2.5.5. *Per tutti gli insiemi* A*, la relazione identità* $Id_A \subseteq A \times A$ *è una funzione. In virtù di questo, d'ora in avanti, scriveremo* $id_A\colon A \to A$*. La relazione vuota* $\varnothing \subseteq A \times B$ *e la relazione completa* $A \times B$ *solitamente non sono funzioni.*

Esercizio 2.5.6. *Per quale insieme* A*, la relazione vuota* $\varnothing \subseteq A \times B$ *e la relazione completa* $A \times B$ *sono funzioni?*

La notazione $f(a) = b$ può essere usata anche per definire funzioni: specificando $f(a) = b$ per ogni elemento a dell'insieme di partenza si definisce infatti una funzione.

Esempio 2.5.7. *La funzione* $R\colon A \to B$ *dell'Esempio 2.5.2 può essere definita come*

$$R(x) = a, \quad R(y) = a, \quad R(z) = d.$$

Quando la funzione è chiara dal contesto, piuttosto che scrivere $f(a) = b$, si può scrivere $a \mapsto b$.

Esempio 2.5.8. *La funzione* $R\colon A \to B$ *dell'Esempio 2.5.2 può essere definita come*

$$x \mapsto a, \quad y \mapsto a, \quad z \mapsto d.$$

Nel caso si considerino insiemi di partenza non finiti, questa notazione è abbastanza sconveniente perché necessita la specifica di $f(a) = b$ per gli infiniti elementi a dell'insieme di partenza (o i soliti punti $\ldots$).

Esempio 2.5.9. *La funzione* $succ\colon \mathbb{N} \to \mathbb{N}$ *può essere definita come*

$$succ(0) = 1, \quad succ(1) = 2, \quad succ(2) = 3, \quad \ldots$$

In questi casi un modo conveniente di specificare funzioni è quello di usare delle espressioni con variabili:

$$f(x) = \langle \text{espressione il cui valore può dipendere da } x \rangle$$

L'idea è che per calcolare l'elemento associato ad un certo valore v dalla funzione f basta sostituire v al posto di x nell'espressione e valutare il risultato.

Esempio 2.5.10.

- $f_1(x) = x$
- $f_2(x) = x + 1$
- $f_3(x) = -x$
- $f_4(x) = x/2$
- $f_5(x) = \sqrt{x}$

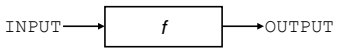
Ma attenzione: di quale insieme di partenza e di arrivo stiamo parlando? Bisogna sempre chiarire quali siano gli insiemi di riferimento!

Per esempio la funzione f_1 può essere definita per ogni insieme A: questa è difatti la funzione identità $id_A\colon A \to A$. Invece la funzione f_2 non può essere definita per ogni insieme, perchè l'espressione $x + 1$ ha significato solamente negli insiemi di numeri (come $\mathbb{N}$, $\mathbb{Z}$, $\mathbb{Q}$, $\mathbb{R}$). La funzione f_3 necessita dell'operazione di sottrazione $-$ e quindi non può essere definita sui numeri naturali,

ma può essere definita sugli interi $\mathbb{Z}$. Similmente, per f_4 non sono sufficienti gli interi ma può essere definita sui razionali $\mathbb{Q}$. Ed infine per f_5 i numeri razionali non sono sufficienti ma questa funzione può essere definita sui reali positivi $\mathbb{R}^+$.

Le funzioni che solitamente sono studiate nei corsi di Analisi Matematica sono funzioni $f\colon \mathbb{R} \to \mathbb{R}$ dove l'insieme di partenza e quello di arrivo sono quello dei numeri reali $\mathbb{R}$ (o dei loro sottoinsiemi). Per questa ragione, spesso tali insiemi sono sottointesi e non vengono specificati.

In questo corso invece, e più in generale in tutta l'Informatica, consideriamo spesso funzioni tra insiemi che non sono numerici ed è quindi fondamentale **specificare sempre l'insieme di partenza e di arrivo.** Infatti, i programmi che scriviamo con i linguaggi di programmazione spesso sono funzioni dove l'insieme di partenza rappresenta l'insieme dei possibili valori di input del programma e l'insieme di arrivo l'insieme dei valori di output.



Di seguito mostriamo un esempio fondamentale nell'informatica.

Esempio 2.5.11. *Si consideri l'insieme dei valori booleani* $Bool = \{\mathsf{t}, \mathsf{f}\}$. *La funzione* $\neg\colon Bool \to Bool$, *chiamata* NEGAZIONE, *è definita come:*

$$\neg(\mathsf{t}) = \mathsf{f} \qquad \neg(\mathsf{f}) = \mathsf{t}$$

In una funzione $f\colon A \to B$, l'insieme di partenza A può essere anche il prodotto cartesiano di due insiemi, vale a dire $A = A_1 \times A_2$ per qualche insieme A_1 e A_2. In tal caso, anziché scrivere $f(\,(a_1, a_2)\,)$ (che, formalmente denota l'elemento associato da f alla coppia $(a_1, a_2) \in A_1 \times A_2$) scriveremo più semplicemente $f(a_1, a_2)$.

Esempio 2.5.12. *Si consideri l'insieme dei valori booleani* $Bool = \{\mathsf{t}, \mathsf{f}\}$. *La funzione* $\wedge\colon Bool \times Bool \to Bool$, *chiamata* CONGIUNZIONE, *è definita come:*

$$\begin{aligned}\wedge(\mathsf{t}, \mathsf{t}) &= \mathsf{t}\\ \wedge(\mathsf{f}, \mathsf{t}) &= \mathsf{f}\\ \wedge(\mathsf{t}, \mathsf{f}) &= \mathsf{f}\\ \wedge(\mathsf{f}, \mathsf{f}) &= \mathsf{f}\end{aligned}$$

La funzione $\vee\colon Bool \times Bool \to Bool$, *chiamata* DISGIUNZIONE, *è definita come:*

$$\begin{aligned}\vee(\mathsf{t}, \mathsf{t}) &= \mathsf{t}\\ \vee(\mathsf{f}, \mathsf{t}) &= \mathsf{t}\\ \vee(\mathsf{t}, \mathsf{f}) &= \mathsf{t}\\ \vee(\mathsf{f}, \mathsf{f}) &= \mathsf{f}\end{aligned}$$

Esempio 2.5.13. *Un altro esempio di funzione in cui l'insieme di partenza è il prodotto cartesiano è la funzione* $plus\colon \mathbb{N} \times \mathbb{N} \to \mathbb{N}$. *Con la notazione descritta sopra si ha, ad esmepio che*

$$plus(2, 1) = 3$$

Le funzioni che hanno come insieme di partenza il prodotto cartesiano di un insieme sono dette BINARIE. Spesso, anziché scriverle con la notazione $f(x, y)$, chiamata NOTAZIONE PREFISSA, sono scritte con la NOTAZIONE INFISSA $x\, f\, y$. Per esempio, $plus(2, 1)$ può essere scritto $2\, plus\, 1$ (o più comunemente $2 + 1$) e $\wedge(\mathsf{t}, \mathsf{t})$ come $\mathsf{t} \wedge \mathsf{t}$.

Diversamente dalla Matematica, in Informatica capita spesso di definire funzioni in cui l'insieme di partenza è il prodotto cartesiano di insiemi diversi.

Esempio 2.5.14. *Si consideri la funzione* $f\colon Bool \times \mathbb{N} \to \mathbb{N}$ *che, ad ogni coppia* $(x, y) \in Bool \times \mathbb{N}$, *assegna* 0 *se* $x = \mathsf{f}$ *e* y *se* $x = \mathsf{t}$. *Come sottoinsieme di* $((Bool \times \mathbb{N}) \times \mathbb{N})$ *questa funzione è*

$$f = \{((\mathsf{f}, x), 0) \in (Bool \times \mathbb{N}) \times \mathbb{N} \mid x \in \mathbb{N}\} \cup \{((\mathsf{t}, x), x) \in (Bool \times \mathbb{N}) \times \mathbb{N} \mid x \in \mathbb{N}\}.$$

Altrimenti si può definire con la notazione

$$f(x,y) = \begin{cases} 0 & \text{se } x = \mathtt{f} \\ y & \text{se } x = \mathtt{t} \end{cases}$$

Durante il corso utilizzeremo questa notazione di tanto in tanto.

Composizione di funzioni

In prima approssimazione, dal momento che sono relazioni, le funzioni potrebbero essere combinate con le operazioni su relazioni che abbiamo visto nella Sezione 2.2. Sfortunatamente questo in molti casi non è possibile perché il risultato di tale operazione è una relazione ma non una funzione.

Prendiamo per esempio l'operazione di relazione opposta: la relazione opposta di una funzione è sicuramente una relazione surgettiva e iniettiva (grazie alla Proposizione 2.4.30), ma solitamente questa non è una funzione. Per un esempio concreto, il lettore può osservare che $succ\colon \mathbb{N} \to \mathbb{N}$ è una funzione, mentre $Pred = succ^{op} \subseteq \mathbb{N} \times \mathbb{N}$ non è una funzione.

In modo del tutto analogo, le operazioni insiemistiche falliscono nel restituire una funzione: se $f\colon A \to B$ e $g\colon A \to B$ sono sue funzioni, allora $f \cap g \subseteq A \times B$ sarà ancora univalente, ma in generale non sarà totale. Similmente $f \cup g \subseteq A \times B$ sarà totale, ma non univalente.

L'unica operazione che preserva la proprietà di essere funzioni è l'operazione di composizione.

Proposizione 2.5.15. *Per tutti gli insiemi A, B, C e per tutte le funzioni $f\colon A \to B$ e $g\colon B \to C$, la relazione $f;g$ è una funzione.*

Dimostrazione. Assumendo che f e g siano totali e univalenti, dobbiamo dimostrare che $f;g$ è totale e univalente.

Dalla Proposizione 2.4.36.1, si ha che $f;g$ è totale, visto che f e g sono totali. Dalla Proposizione 2.4.36.2, si ha che $f;g$ è univalente, visto che f e g sono univalenti. ■

Visto che le leggi illustrate nella Sezione 2.3 valgono per tutte le relazioni, valgono in particolare anche per le funzioni.

Proposizione 2.5.16. *Per tutti gli insiemi A, B, C, D e per tutte le funzioni $f\colon A \to B$, $g\colon B \to C$, $h\colon C \to D$ vale che:*

2.5.16.1. $f;(g;h) = (f;g);h$; *(associatività)*

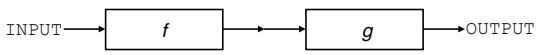
2.5.16.2. $id_A; f = f = f; id_B$. *(unità)*

È importante notare che, come per le relazioni, le funzioni possono essere composte solo se l'insieme di arrivo della prima coincide con l'insieme di partenza della seconda.

Esempio 2.5.17. *La funzione $f\colon Bool \times \mathbb{N} \to \mathbb{N}$ dell'Esempio 2.5.14 può essere composta con la funzione $succ\colon \mathbb{N} \to \mathbb{N}$. La funzione risultante $f;succ\colon Bool \times \mathbb{N} \to \mathbb{N}$ può essere definita come segue.*

$$f(x,y) = \begin{cases} 0 & \text{se } x = \mathtt{f} \\ y+1 & \text{se } x = \mathtt{t} \end{cases}$$

In Informatica si usa spesso la composizione di programmi visti come funzioni: $f;g$ significa intuitivamente che prima viene eseguito il programma f e poi l'output di questo viene passato come input al programma g.



La composizione di funzioni è molto importante anche in Matematica (per esempio, per calcolare le derivate) ma viene spesso utilizzata una notazione diversa.

Notazione 2.5.18. *La composizione $f;g$ viene talvolta indicata come $g \circ f$, oppure, omettendo $\circ$, da gf. Inoltre l'elemento $f;g(a)$ viene denotato come $g(f(a))$ oppure, omettendo le parentesi più esterne, come $gf(a)$.*

Per esempio per dire che f è la composizione della funzione seno $sin\colon \mathbb{R} \to \mathbb{R}$ e della funzione coseno $cos\colon \mathbb{R} \to \mathbb{R}$, si scrive $f(x) = cos(sin(x))$.

Teorema di caratterizzazione

Il Teorema 2.4.34 fornisce una caratterizzazione per le quattro proprietà fondamentali. Grazie a tale teorema, si ottiene banalmente una caratterizzazione delle funzioni.

Teorema 2.5.19. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

$$R \text{ è una funzione se e solo se } id_A \subseteq R; R^{op} \text{ e } R^{op}; R \subseteq id_B.$$

Esercizio 2.5.20. *Dimostrare il Teorema 2.5.19 utilizzando il Teorema 2.4.34.*

Tale caratterizzazione può essere resa più forte, rimpiazzando R^{op} con una qualsiasi relazione $S \subseteq B \times A$.

Teorema 2.5.21. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

$$\begin{gathered} R \text{ è una funzione} \\ \text{se e solo se} \\ \text{esiste } S \subseteq B \times A \text{ tale che } id_A \subseteq R; S \text{ e } S; R \subseteq id_B. \end{gathered}$$

Dimostrazione. Trattandosi di un se e solo se, possiamo dividere la dimostrazione in due parti:

- Se $id_A \subseteq R; S$ e $S; R \subseteq id_B$, allora R è una funzione. Dimostriamo prima che R è totale. Prendiamo un qualsiasi elemento $a \in A$. Visto che $id_A \subseteq R; S$, si ha che $(a, a) \in R; S$. Per definizione di composizione, questo significa che esiste almeno un $b \in B$ tale che $(a, b) \in R$ e $(b, a) \in S$.

  Dimostriamo adesso che R è univalente. Prendiamo un qualsiasi elemento $a \in A$ ed assumiamo che $(a, x) \in R$ e $(a, y) \in R$ per qualche $x, y \in B$. Seguendo lo stesso argomento di sopra si ha che esiste almeno un $b \in B$ tale che $(b, a) \in S$. Pertanto, per definizione di composizione, si ha che $(b, x) \in S; R$ e $(b, y) \in R; S$. Visto che $R; S \subseteq id_B$, si ha che $x = y$. Questo prova che R è univalente.

- Se R è una funzione allora esiste un S tale che $id_A \subseteq R; S$ e $S; R \subseteq id_B$. Per il Teorema 2.5.19, è sufficiente prendere $S = R^{op}$.

■

Esercizio 2.5.22. *Dimostrare che per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

$$\begin{gathered} R \text{ è una relazione surgettiva ed iniettiva} \\ \text{se e solo se} \\ \text{esiste } S \subseteq B \times A \text{ tale che } id_A \subseteq R; S \subseteq id_A \text{ e } id_B \subseteq S; R. \end{gathered}$$

Funzioni Parziali

Abbiamo spiegato a livello intuitivo che un programma calcola una funzione. Talvolta il calcolo può non andare a buon fine: il programma può per esempio restituire un messaggio di errore o addirittura non terminare. In questo non è vero che per tutti i valori di input il programma restituisce un output, cioè la funzione calcolata da un programma potrebbe non essere una relazione totale, ma solamente univalente.

Definizione 2.5.23. *Siano A e B due insiemi ed R una relazione tra di loro. $R \subseteq A \times B$ si dice una* FUNZIONE PARZIALE *se R è univalente.*

Per le funzioni parziali, si utilizza la stessa notazione che abbiamo descritto per le funzioni.

Esempio 2.5.24. *Si consideri* $f\colon \mathbb{R} \to \mathbb{R}$ *definita come*

$$f(x) = 1/x$$

La divisione di qualunque numero n *per* 0 *non è definita. Per questa ragione* f *è una funzione parziale.*

Come per le funzioni, le funzioni parziali possono essere composte e la loro composizione è associativa.

Proposizione 2.5.25. *Per tutti gli insiemi* A, B, C *e per tutte le funzioni parziali* $f\colon A \to B$ *e* $g\colon B \to C$*, la relazione* $f; g$ *è una funzione parziale.*

Proposizione 2.5.26. *Per tutti gli insiemi* A, B, C, D *e per tutte le funzioni parziali* $f\colon A \to B$*,* $g\colon B \to C$*,* $h\colon C \to D$ *vale che:*

2.5.26.1. $f; (g; h) = (f; g); h$*; (associatività)*

2.5.26.2. $id_A; f = f = f; id_B$*. (unità)*

Esercizio 2.5.27. *Dimostrare le Proposizioni 2.5.25 e 2.5.26.*

Funzioni surgettive ed iniettive

Il lettore avrà probabilmente sentito parlare di funzioni surgettive ed iniettive: queste sono relazioni che sono funzioni (quindi totali e univalenti) ed in aggiunta godono della proprietà surgettiva o di quella iniettiva.

Definizione 2.5.28. *Siano* A *e* B *due insiemi ed* R *una relazione tra di loro.* $R \subseteq A \times B$ *si dice una* FUNZIONE SURGETTIVA *se* R *è totale, univalente e surgettiva.*

Definizione 2.5.29. *Siano* A *e* B *due insiemi ed* R *una relazione tra di loro.* $R \subseteq A \times B$ *si dice una* FUNZIONE INIETTIVA *se* R *è totale, univalente e iniettiva.*

Come per le funzioni e le funzioni parziali, le funzioni surgettive ed iniettive possono essere composte

Proposizione 2.5.30. *Per tutti gli insiemi* A, B, C *e per tutte le funzioni surgettive* $f\colon A \to B$ *e* $g\colon B \to C$*, la relazione* $f; g$ *è una funzione surgettiva.*

Proposizione 2.5.31. *Per tutti gli insiemi* A, B, C *e per tutte le funzioni iniettive* $f\colon A \to B$ *e* $g\colon B \to C$*, la relazione* $f; g$ *è una funzione iniettiva.*

Esercizio 2.5.32. *Dimostrare le Proposizioni 2.5.30 e 2.5.31.*

2.6 Biiezioni

Le funzioni che sono allo stesso tempo surgettive ed iniettive sono particolarmente importanti e vengono dette biiezioni.

Definizione 2.6.1. *Siano* A *e* B *due insiemi ed* R *una relazione tra di loro.* $R \subseteq A \times B$ *si dice una* BIIEZIONE *se* R *è totale, univalente, surgettiva ed iniettiva.*

È importante notare che ogni biiezione è anche una funzione. Per questo utilizzeremo la stessa notazione che abbiamo usato per le funzioni.

Esempio 2.6.2. *Per ogni insieme* A*,* $id_A\colon A \to A$ *è una biiezione.*

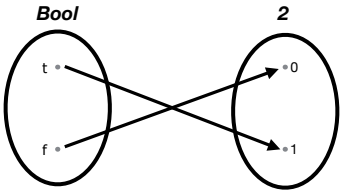
Esempio 2.6.3. *Si consideri l'insieme* $Bool = \{\mathsf{t}, \mathsf{f}\}$ *e l'insieme* $2 = \{0, 1\}$*. La relazione* $i\colon Bool \to 2$

$$\{(\mathsf{f}, 0),\ (\mathsf{t}, 1)\}$$

è una biiezione. Questa può anche essere specificata come

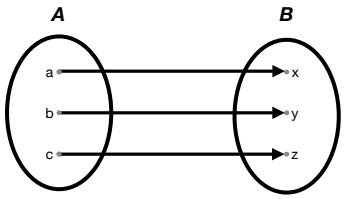
$$i(\mathsf{f}) = 0, \quad i(\mathsf{t}) = 1$$

e illustrata attraverso il seguente diagramma.



Esercizio 2.6.4. *Esiste un'altra biiezione da Bool a 2?*

Esempio 2.6.5. *Si consideri l'insieme* $A = \{a, b, c\}$ *e l'insieme* $B = \{x, y, z\}$. *La relazione*



è una biiezione da A *a* B.

Esercizio 2.6.6. *Siano* A *e* B *gli insiemi dell'Esempio 2.6.5. Esistono altre biiezioni da* A *a* B*? In caso positivo, quante sono?*

Come per le funzioni, le biiezioni possono essere composte.

Proposizione 2.6.7. *Per tutti gli insiemi* A, B, C *e per tutte le biiezioni* $f\colon A \to B$ *e* $g\colon B \to C$, *la relazione* $f;g$ *è una biiezione.*

Esercizio 2.6.8. *Dimostrare la Proposizione 2.6.7.*

Esiste però una differenza fondamentale tra funzioni e biiezioni: data una funzione $f\colon A \to B$ non necessariamente $f^{op} \subseteq B \times A$ è una funzione, invece se f è una biiezione, allora anche f^{op} lo è.

Proposizione 2.6.9. *Per tutti gli insiemi* A, B *e per tutte le biiezioni* $f\colon A \to B$, *la relazione* f^{op} *è una biiezione.*

Dimostrazione. Assumendo che f sia totale, univalente, surgettiva ed iniettiva, dobbiamo dimostrare che f^{op} ha le stesse proprietà.

- Grazie alla Proposizione 2.4.30.1, f^{op} è surgettiva perché f è totale.
- Grazie alla Proposizione 2.4.30.2, f^{op} è iniettiva perché f è univalente.
- Grazie alla Proposizione 2.4.32.1, f^{op} è totale perché f è surgettiva.
- Grazie alla Proposizione 2.4.32.2, f^{op} è univalente perché f è iniettiva.

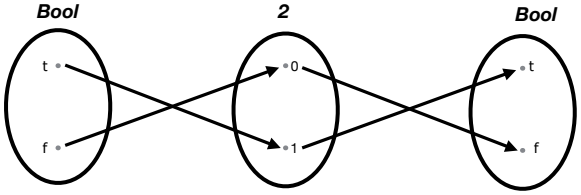
■

Esercizio 2.6.10. *Dimostrare che se* f^{op} *è una biiezione allora anche* f *lo è.*

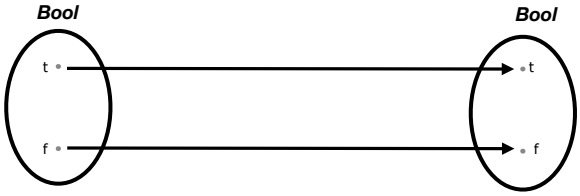
Esercizio 2.6.11. *Si ricordi la biiezione i dell'Esempio 2.6.3. Illustrare la sua biiezione opposta attraverso un diagramma.*

Esercizio 2.6.12. *Si ricordi la biiezione dell'Esempio 2.6.5. Illustrare la sua biiezione opposta attraverso un diagramma.*

Componendo una biiezione $f\colon A \to B$ con la sua biezione opposta $f^{op}\colon B \to A$ si ha che $f; f^{op} = id_A$. Per esempio, componendo la biiezione $i\colon Bool \to 2$ dell'Esempio 2.6.3 con $i^{op}\colon 2 \to Bool$



si ottiene $id_{Bool}\colon Bool \to Bool$.



Similmente componendo $f^{op}\colon B \to A$ con $f\colon A \to B$ si ottiene che $f^{op}; f = id_B$.

Questa proprietà fornisce una caratterizzazione delle biiezioni tra tutte le relazioni.

Teorema 2.6.13. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

$$R \text{ è una biiezione se e solo se } R; R^{op} = id_A \text{ e } R^{op}; R = id_B.$$

Esercizio 2.6.14. *Dimostrare il Teorema 2.6.13. Suggerimento: si utilizzi il Teorema 2.4.34.*

Teorema di caratterizzazione attraverso relazioni invertibili

La caratterizzazione fornita dall Teorema 2.6.13 può essere resa ancora più forte utilizzando la nozione di relazione inversa.

Definizione 2.6.15. *Siano $R \subseteq A \times B$ e $S \subseteq B \times A$ due relazioni. Si dice che S è l'*INVERSA *di R se $R; S = id_A$ e $S; R = id_B$. Si dice che R è* INVERTIBILE *se esiste almeno una relazione inversa di R.*

Se esiste una relazione inversa di R, allora questa è unica.

Proposizione 2.6.16. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$, $S \subseteq B \times A$ e $T \subseteq B \times A$ vale che:*

$$\text{se } S \text{ è l'inversa di } R \text{ e } T \text{ è l'inversa di } R, \text{ allora } S = T.$$

Dimostrazione.

$$\begin{aligned} S &= id_B; S && \text{(unità)} \\ &= T; R; S && (T; R = id_B) \\ &= T; id_A && (R; S = id_A) \\ &= T && \text{(unità)} \end{aligned}$$

■

Teorema 2.6.17. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

$$R \text{ è una biiezione se e solo se } R \text{ è invertibile.}$$

Dimostrazione. Trattandosi di un se e solo se possiamo dimostrare separatamente che

- Se R è una biiezione, allora R è invertibile;

- Se R è invertibile, allora R è una biiezione.

Partiamo dal primo punto. Se R è una biiezione allora, grazie al Teorema 2.6.13, $R; R^{op} = id_A$ e $R^{op}; R = id_B$. Pertanto, prendendo $S = R^{op}$ si ha che R è invertibile.

Dimostriamo adesso il secondo punto. Se esite un S tale che $id_A = R; S$ e $S; R = id_B$, allora per il Teorema 2.5.21, si ha che R è una funzione e, per l'Esercizio 2.5.22, che R è surgettiva e iniettiva. Quindi R è una biiezione. ■

Il Teorema 2.6.17 garantisce che se la relazione R è invertibile, allora è una biiezione e quindi, grazie al Teorema 2.6.13, sappiamo che R^{op} è sicuramente l'unica inversa di R.

Nel resto delle note scriveremo R^{-1} al posto di R^{op} quando R è una biiezione, per enfatizzare che R^{op} è l'inversa di R.

Insiemi in biiezione

La caratterizzazione delle biiezioni come relazioni invertibili è molto interessante: dopo aver applicato una relazione invertibile $i\colon A \to B$ ad un qualsiasi elemento in A, è sempre possibile tornare indietro all'elemento originale utilizzando $i^{-1}\colon B \to A$. Viceversa, trasformando prima con $i^{-1}\colon B \to A$ è sempre possibile tornare all'elemento orginale utilizzando i.

$$a \overset{i}{\mapsto} b \overset{i^{-1}}{\mapsto} a$$

$$b \overset{i^{-1}}{\mapsto} b \overset{i}{\mapsto} a$$

Intuitivamente l'esistenza della biiezione $i\colon A \to B$ garantisce che ci possiamo spostare avanti e indietro tra gli elementi di A e gli elementi di B senza perdere alcuna "informazione". Questo permette quasi di identificare gli insiemi A e B come se fossero lo stesso insieme.

Definizione 2.6.18. *Due insiemi A e B si dicono* IN BIIEZIONE *(o in corrispondenza uno a uno) se esiste una biiezione $i\colon A \to B$. In questo caso scriviamo $A \cong B$.*

Esempio 2.6.19. *Abbiamo mostrato che esiste una biiezione dall'insieme $Bool = \{\mathtt{t}, \mathtt{f}\}$ all'insieme $2 = \{0, 1\}$. I due insiemi sono quindi in biiezione: $Bool \cong 2$. È molto comune, in Informatica e in Logica identificare questi due insiemi, identificando $\mathtt{t}$ con 1 e $\mathtt{f}$ con 0.*

Esercizio 2.6.20. *Dimostrare che per tutti gli insiemi A, se $A \cong \varnothing$, allora $A = \varnothing$.*

Le trasformazioni invertibili giocano un ruolo chiave in moltissimi campi. Per esempio in Informatica, sono molto studiate per la Crittografia (la materia che studia come cifrare i messagi in modo sicuro) e nelle computazioni quantistiche. Purtroppo, questi argomenti non possono essere trattati in questo corso introduttivo e ci limiteremo quindi a fare delle biiezioni il loro uso più classico, cioè quello di mettere in corrispondenza uno-a-uno insiemi apparentemente distinti.

Proposizione 2.6.21. *Per tutti gli insiemi A, vale che:*

$$\mathcal{P}(A) \cong 2^A$$

Dimostrazione. Ricordiamo che $\mathcal{P}(A)$ è l'insieme delle parti di A (Definizione 1.5.2), mentre 2^A è l'insieme di tutte le funzioni $f\colon A \to 2 = \{0, 1\}$ (Notazione 2.5.3). Mostriamo per prima cosa una funzione $i\colon \mathcal{P}(A) \to 2^A$. Per ogni insieme $B \subseteq A$, definiamo la sua FUNZIONE CARATTERISTICA $\chi_B\colon A \to 2$ come

$$\chi_B(a) = \begin{cases} 1 & \text{se } a \in B \\ 0 & \text{se } a \notin B \end{cases}$$

Equivalentemente, possiamo definire questa funzione come una relazione, cioè un sottoinsieme di $A \times 2$

$$\chi_B = \{(a, 1) \in A \times 2 \mid a \in B\} \cup \{(a, 0) \in A \times 2 \mid a \notin B\}$$

La funzione $i\colon \mathcal{P}(A) \to 2^A$ mappa ogni $B \subseteq A$ nella sua funzione caratteristica χ_B.

$$\begin{array}{rccc} i\colon & \mathcal{P}(A) & \to & 2^A \\ & B \subseteq A & \mapsto & \chi_B \end{array}$$

Mostriamo ora una funzione $j\colon 2^A \to \mathcal{P}(A)$. Per ogni funzione $f\colon A \to 2$, definiamo $Sub(f) \subseteq A$ come

$$Sub(f) = \{x \in A \mid f(x) = 1\}$$

La funzione j associa ad ogni $f\colon A \to 2$ il sottoinsieme $Sub(f)$.

$$\begin{array}{rccc} j\colon & 2^A & \to & \mathcal{P}(A) \\ & f\colon A \to 2 & \mapsto & Sub(f) \end{array}$$

Per mostrare che i è una biiezione, mostriamo che j è la sua funzione inversa, cioè che

$$i; j = id_{\mathcal{P}(A)} \qquad \text{e} \qquad j; i = id_{2^A}$$

- $\underline{i; j = id_{\mathcal{P}(A)}}$ Dobbiamo dimostrare che per tutti i $B \in \mathcal{P}(A)$ vale che $j(i(B)) = id_{\mathcal{P}(A)}(B)$, cioè che $Sub(\chi_B) = B$. Ma questo è immediato perché per tutti gli $a \in A$

$$a \in Sub(\chi_B) \text{ se e solo se } \chi_B(a) = 1 \text{ se e solo se } a \in B.$$

- $\underline{j; i = id_{2^A}}$ Dobbiamo dimostrare che per tutte le $f \in 2^A$ vale che $i(j(f)) = id_{2^A}(f)$, cioè che $\chi_{Sub(f)} = f$. Ma questo è immediato perché per tutti gli $a \in A$, vale che:

$$\chi_{Sub(f)}(a) = 1 \text{ se e solo se } a \in Sub(f) \text{ se e solo se } f(a) = 1$$

e

$$\chi_{Sub(f)}(a) = 0 \text{ se e solo se } a \notin Sub(f) \text{ se e solo se } f(a) = 0.$$

■

Esempio 2.6.22. *Siano $A = \{a, b, c\}$ e $B = \{a, c\}$. La funzione caratteristica $\chi_B\colon A \to 2$ è*

$$\begin{array}{rccc} \chi_B\colon & A & \to & 2 \\ & a & \mapsto & 1 \\ & b & \mapsto & 0 \\ & c & \mapsto & 1 \end{array}$$

Esercizio 2.6.23. *Sia $A = \{a, b, c, d\}$. Illustare la funzione caratteristica dei sottoinsiemi $\varnothing \subseteq A$, $A \subseteq A$ e $\{b, d\} \subseteq A$.*

Il seguente risultato illustra un'importante biiezione che sta alla base della progragmmazione funzionale. Lasciamo la sua dimostrazione come esericzio.

Esercizio 2.6.24. *Dimostrare che per tutti gli insiemi A, B, C, vale che*

$$C^{A \times B} \cong (C^B)^A$$

Per $\cong$ valgono le stesse proprietà dell'uguaglianza insiemistica.

Proposizione 2.6.25. *Per tutti gli insiemi A, B, C vale che:*

2.6.25.1. *$A \cong A$ (riflessività);*

2.6.25.2. *Se $A \cong B$ e $B \cong C$, allora $A \cong C$ (transitività);*

2.6.25.3. *Se $A \cong B$, allora $B \cong A$ (simmetria).*

Esercizio 2.6.26. *Dimostrare la Proposizione 2.6.25. Suggerimento: per la transitività utilizzare la Proposizione 2.6.7.*

Nell'Esercizio 1.6.5 abbiamo visto che il prodotto cartesiano non è associativo, cioè che in generale

$$A \times (B \times C) \neq (A \times B) \times C$$

Comunque l'associatività vale *a meno di biiezione*, cioè i due insiemi anche se diversi sono in biiezione.

Proposizione 2.6.27. *Per tutti gli insiemi A, B, C vale che*

$$A \times (B \times C) \cong (A \times B) \times C$$

Dimostrazione. La funzione $i\colon A \times (B \times C) \to (A \times B) \times C$ è definita come

$$(a,\, (b,c)) \mapsto ((a,b),\, c)$$

mentre la funzione $j\colon (A \times B) \times C \to A \times (B \times C)$ come

$$((a,b),\, c) \mapsto (a,\, (b,c)).$$

È immediato vedere che j è la funzione inversa di i. ■

Esercizio 2.6.28. *Dimostrare che per tutti gli insiemi A, B, vale che*

$$A \times B \cong B \times A$$

Esercizio 2.6.29. *Si ricordi che $1 = \{0\}$. Dimostrare che per tutti gli insiemi A, vale che*

$$A \times 1 \cong A$$

2.7 n-uple, sequenze di lunghezza fissata e arbitraria

La Proposizione 2.6.27 ci dice che gli insiemi $A \times (B \times C)$ e $(A \times B) \times C$ anche se non sono uguali sono comunque in biiezione. Per questa ragione, prenderemo la libertà di scrivere

$$A \times B \times C$$

senza dover specificare le parentesi. Inoltre, denoteremo gli elementi di questo insieme come delle TRIPLE

$$(a, b, c)$$

dove $a \in A$, $b \in B$ e $c \in C$. Formalmente, $A \times B \times C = \{(a,b,c) \mid a \in A,\, b \in B,\, c \in C\}$. Quando il prodotto cartesiano è fra 4 insiemi A, B, C, D scriveremo

$$A \times B \times C \times D$$

e denoteremo gli elementi di questo insieme come delle QUADRUPLE

$$(a, b, c, d).$$

Per il prodotto cartesiano per 5 insiemi, faremo la stessa cosa e chiameremo i suo elementi quintuple. Lo stesso procedimento si applica ad un qualsiasi numero $n \in \mathbb{N}$ di insiemi e chiamiamo gli elementi n-uple. Per $n = 0$, esiste una sola 0-upla che denotiamo con $()$.

Le n-uple sono onnipresenti in Informatica. Infatti, mentre in matematica la maggior parte delle funzioni sono unarie o binarie, le funzioni che rappresentano programmi hanno spesso insiemi di partenza che sono il prodotto cartesiano di più di due insiemi.

Esempio 2.7.1. *Mostriamo come esempio una semplice variante dell'Esempio 2.5.14. Consideriamo la funzione $f\colon Bool \times \mathbb{N} \times \mathbb{N} \to \mathbb{N}$ definita come*

$$f(x,y,z) = \begin{cases} 0 & \text{se } x = \texttt{f} \\ y + z & \text{se } x = \texttt{t} \end{cases}$$

Questa funzione assegna ad ogni tripla $(x,y,z) \in Bool \times \mathbb{N} \times \mathbb{N}$, $y + z$ se $x = \texttt{t}$ e 0 altrimenti. Si noti che il primo elemento della tripla deve essere un elemento di $Bool$, mentre il secondo e il terzo sono numeri naturali in $\mathbb{N}$.

Un caso di n-upla particolarmente interessante è quello in cui tutti gli insiemi coinvolti nel prodotto cartesiano sono uguali.

Definizione 2.7.2. *Sia A un insieme. Una* SEQUENZA SU A DI LUNGHEZZA n *è una n-upla $(a_0, a_1, \ldots, a_{n-1})$ dove per tutti gli indici $i \in \{0, \ldots, n-1\}$, $a_i \in A$. L'insieme A^n di tutte le sequenze su A di lunghezza n è definito come:*

$$A^n = \{(a_0, a_1, \ldots, a_{n-1}) \mid \forall i \in \{0, \ldots, n-1\} \,.\, a_i \in A\}$$

Esempio 2.7.3. *Sia $A = \{a, b\}$. Allora*

$$\begin{array}{rcl} A^0 & = & \{()\} \\ A^1 & = & \{(a),\ (b)\} \\ A^2 & = & \{(a,a),\ (a,b),\ (b,a),\ (b,b)\} \\ A^3 & = & \{(a,a,a),\ (a,a,b),\ (a,b,a),\ (a,b,b),\ (b,a,a),\ (b,a,b),\ (b,b,a),\ (b,b,b)\} \\ \ldots & = & \ldots \end{array}$$

Osservazione 2.7.4. *È interessante osservare che $A^0 \cong 1$, $A^1 \cong A$, $A^2 \cong A \times A$, $A^3 \cong A \times (A \times A)$ e, più in generale $A^n \cong A \times (\cdots \times (A \times A) \ldots)$. Questo può essere espresso attraverso l'induzione, un argomento che affronteremo in modo approfondito nel Capitolo 5 (se veda l'Esempio 5.2.9):*

$$\begin{array}{rcl} A^0 & \cong & 1 \\ A^{n+1} & \cong & A \times A^n \end{array}$$

Queste strutture sono molto usate in Matematica e Fisica: quando l'insieme A è l'insieme dei numeri reali $\mathbb{R}$, una sequenza su $\mathbb{R}$ di lunghezza n è chiamata un *vettore* di $\mathbb{R}^n$ e viene solitamente rappresentata senza virgole in riga $(r_0\ r_1,\ \ldots\ r_n)$ o in colonna.

$$\begin{pmatrix} r_0 \\ r_1 \\ \vdots \\ r_{n-1} \end{pmatrix}$$

L'insieme di tutte le sequenze su $\mathbb{R}$ di lunghezza n, $\mathbb{R}^n$, forma uno *spazio vettoriale*: nel corso di Algebra Lineare studierete approfonditamente questo concetto.

In Informatica, le sequenze su A di lunghezza n sono tipicamente chiamate *array*, una struttura dati di base nella maggior parte dei linguaggi di programmazione. La sequenza

$$a = (a_0, a_1, \ldots, a_{n-1})$$

viene solitamente rappresentata come

$$a = \boxed{a_0} \boxed{a_1} \boxed{\ldots} \boxed{a_{n-1}}$$

Esempio 2.7.5. *Una sequenza su $2 = \{0, 1\}$ di lunghezza n è un array di n bits. Per esempio,*

$$a = \boxed{0} \boxed{1} \boxed{\ldots} \boxed{1}$$

Una sequenza su $\mathbb{Z}$ di lunghezza n è un array di n interi. Per esempio,

$$a = \boxed{3} \boxed{-1} \boxed{\ldots} \boxed{26}$$

Ricordandosi che ogni numero naturale $n \in \mathbb{N}$ può essere visto come l'insieme $n = \{0, \ldots n-1\}$, ogni sequenza su a di lunghezza n può essere vista come una funzione dall'insieme n ad A.

$$\begin{array}{rccc} (a_0, a_1, \ldots a_{n-1}): & n & \to & A \\ & i \in n & \mapsto & a_i \in A \end{array}$$

Viceversa, ogni funzione da n ad A può essere vista come una sequenza su A di lunghezza n.

Osservazione 2.7.6. *Per tale ragione, la notazione A^n, che denota sia l'insieme di tutte le funzioni da n ad A che l'insieme di tutte le sequenze su A di lunghezza n, non è ambigua o fuorviante.*

Osservazione 2.7.7. *Quando si pensa a un array come una funzione $a\colon n \to A$, $a(i)$ denota l'i-esimo elemento di a (a_i). Nella maggior parte dei linguaggi di programmazione, anziché scrivere $a(i)$ si scrive $a[i]$. Comunque molto spesso, gli array non sono visti come funzioni di tipo $n \to A$, ma come* funzioni parziali *di tipo $\mathbb{Z} \to A$. Infatti, è possibile passare ad un array un qualssiasi indice intero $i \in \mathbb{Z}$ e quando $i \notin n$ si ottiene un errore.*

Concludiamo questa sezione con una nozione che ritroveremo frequentemente durante tutto il corso e in particolare nel Capitolo 7.

Definizione 2.7.8. *Una* SEQUENZA SU A DI LUNGHEZZA ARBITRARIA *è una sequenza su A di lunghezza n per un qualsiasi numero naturale $n \in \mathbb{N}$. L'insieme A^* di tutte le sequenze su A di lunghezza arbitraria è definito come*

$$A^* = \bigcup_{n \in \mathbb{N}} A^n$$

Si noti che se l'insieme A ha un numero finito di elementi, allora per ogni $n \in \mathbb{N}$ l'insieme A^n è finito, ma in generale l'insieme A^* non lo è. Infatti

$$A^* = A^0 \cup A^1 \cup A^2 \cup \cdots \cup A^n \cup \ldots$$

Esercizio 2.7.9. *Prendiamo l'insieme A come $1 = \{0\}$, dimostrare che $1^* \cong \mathbb{N}$.*

Esercizio 2.7.10. *Esiste un insieme A per il quale A^* è finito? Se si, quale?*

Molto spesso le sequenze su A di lunghezza aribtraria sono chiamate *stringhe* sull'alfabeto A: anziché scrivere

$$(a_0, a_1, \ldots, a_n)$$

si scrive semplicemente

$$a_0 a_1 \ldots a_n$$

omettendo parentesi e virgole. In questa prospettiva, l'unica sequenza di lunghezza 0, prima denotata da (), viene scritta come la *stringa vuota* ε.

Esempio 2.7.11. *Prendiamo come A l'insieme AN dei caratteri alfanumerici. L'insieme AN^* contiene tutte le stringhe composte da caratteri alfanumerici, come per esempio* `si`, `123`, `anna`, `pippo`, `Pluto` *e* `topolino3`*.*

CAPITOLO 3

Relazioni su un insieme

Nel capitolo precedente abbiamo affrontato il concetto di relazione, al più alto livello di generalità. In Informatica, ma anche in Matematica e Fisica, ricoprono un ruolo di particolare interesse quelle relazioni in cui l'insieme di partenza e l'insieme di arrivo sono lo stesso insieme. In questo capitolo, ci concentreremo su questo tipo di relazioni: mostreremo la loro rappresentazione diagrammatica attraverso dei grafi e illustreremo quattro proprietà di comune interesse: riflessività, transitività, simmetria ed anti-simmetria. Vedremo poi il concetto di chiusura, con particolare enfasi sulla stella di Kleene (chiusura riflessiva e transitiva). Infine studieremo le relazioni di equivalenza e di ordinamento.

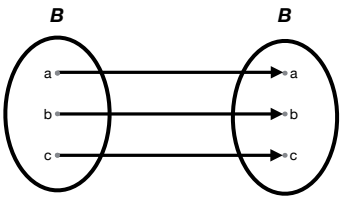
3.1 Nozioni di Base

Fino ad ora abbiamo studiato relazioni R tra un insieme A e un insieme B. In questo capitolo, ci concentremo su relazioni in cui l'insieme di partenza e quello di arrivo sono lo stesso.

Definizione 3.1.1. *Una* RELAZIONE R *sull'*INSIEME A *è un sottoinsieme del prodotto cartesiano* $A \times A$*, quindi* $R \subseteq A \times A$*.*

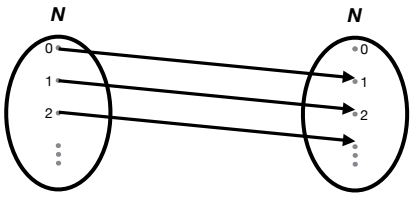
Abbiamo già visto molti esempi di relazioni di questo tipo nel Capitolo 2: le relazioni di parentela sull'insieme degli esseri umani, la relazione identità su A (per un qualsiasi insieme A), le relazioni *succ* e *Pred* sull'insieme dei numeri naturali.

La prima peculiarità di questo tipo di relazioni consiste nella loro rappresentazione diagrammatica: anziché rappresentare due insiemi, quello di partenza sulla sinistra e quello di arrivo sulla destra, visto che i due insiemi sono lo stesso, si disegnano solamente gli elementi dell'insieme e, per ogni $(a, b) \in R$, si inserisce una freccia da a a b. Per esempio, la relazione id_B (per $B = \{a, b, c\}$) oltre ad essere disegnata come



può essere disegnata come

Similmente, la relazione $succ \subseteq \mathbb{N} \times \mathbb{N}$ oltre ad essere disegnata come



può essere rappresentata come

Questo tipo di diagrammi vengono chiamati *grafi*: gli elementi di A (rappresentati da $\bullet$) vengono chiamati *nodi*, mentre le frecce vengono chiamate *archi*. I grafi rivestono un ruolo chiave in tutta l'Informatica per la loro importanza nelle applicazioni (per esempio nelle reti sociali, i cosiddetti "social") e la Teoria dei Grafi è considerata da molti una delle aree teoriche più interessanti. Affronteremo con maggiore dettaglio i grafi nel Capitolo 4 ma, per il momento, ci limiteremo ad utilizzarli come diagrammi per rappresentare relazioni su un insieme.

3.2 Proprietà di relazioni su un insieme

Tutto ciò che abbiamo visto sino ad adesso per le relazioni tra due insiemi vale chiaramente anche per le relazioni su un insieme. Ma ci sono diverse proprietà che ha senso considerare solo quando l'insieme di partenza e quello di arrivo coincidono.

Definizione 3.2.1. *Sia $R \subseteq A \times A$ una relazione su A. Si dice che R è* RIFLESSIVA *se per tutti gli elementi $a \in A$, vale che*

$$(a, a) \in R.$$

Esempio 3.2.2. *Per ogni insieme A, la relazione identità $id_A \subseteq A \times A$ e la relazione completa $A \times A \subseteq A \times A$ sono riflessive. La relazione vuota $\varnothing \subseteq A \times A$ solitamente non è riflessiva.*

Esercizio 3.2.3. *Per quale insieme A, la relazione vuota $\varnothing \subseteq A \times A$ è riflessiva?*

Esempio 3.2.4. *La relazione minore ($<$) sui naturali, cioè*

$$< = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid x < y\},$$

non è riflessiva. La relazione minore o uguale ($\leq$) sui naturali, cioè

$$\leq = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid x \leq y\},$$

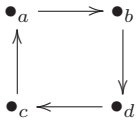
è riflessiva perché, per ogni $x \in \mathbb{N}$, $(x, x) \in \leq$.

Esempio 3.2.5. *Per $A = \{a, b, c, d\}$, la relazione $R = \{(a, a),\ (b, b),\ (a, d)\}$ non è riflessiva, mentre la relazione $S = \{(a, a),\ (b, b),\ (c, c),\ (d, d),\ (a, d)\}$ è riflessiva.*

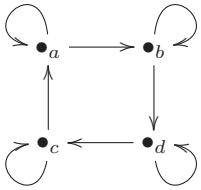
È molto facile identificare le relazioni riflessive attraverso i diagrammi: $R \subseteq A \times A$ è riflessiva se e solo se

per ogni nodo $\bullet_a$ c'è un arco 

Per esempio la relazione



non è riflessiva, mentre la relazione



è riflessiva.

Consideriamo adesso una seconda proprietà.

Definizione 3.2.6. *Sia $R \subseteq A \times A$ una relazione su A. Si dice che R è* TRANSITIVA *se per tutti gli elementi $a, b, c \in A$, vale che*

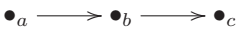
se $(a,b) \in R$ *e* $(b,c) \in R$, *allora* $(a,c) \in R$.

Esempio 3.2.7. *Per ogni insieme* A, *la relazione identità* $id_A \subseteq A \times A$, *la relazione completa* $A \times A \subseteq A \times A$ *e la relazione vuota* $\varnothing \subseteq A \times A$ *sono transitive.*

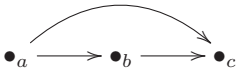
Esempio 3.2.8. *Le relazioni* $< \subseteq \mathbb{N} \times \mathbb{N}$ *e* $\leq \subseteq \mathbb{N} \times \mathbb{N}$ *sono transitive.*

Esempio 3.2.9. *Per* $A = \{a,b,c,d\}$, *la relazione* $R = \{(a,b),\ (b,c),\ (a,d)\}$ *non è transitiva, mentre la relazione* $S = \{(a,b),\ (b,c),\ (a,c),\ (a,d)\}$ *lo è.*

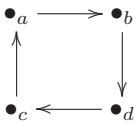
È molto facile identificare le relazioni transitive attraverso i diagrammi: $R \subseteq A \times A$ è transitiva se e solo se ogni volta che ci sono due archi

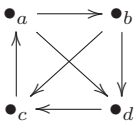


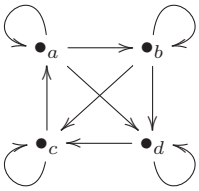
c'è anche un arco



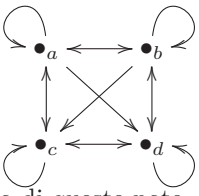
Per esempio le seguenti tre relazioni su $A = \{a,b,c,d\}$



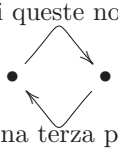




non sono transitive, mentre la relazione



è transitiva. Nel grafo di sopra e nel resto di queste note $\bullet \longleftrightarrow \bullet$ è una notazione compatta per indicare



Continuiamo la nostra esposizione con una terza proprietà.

Definizione 3.2.10. *Sia* $R \subseteq A \times A$ *una relazione su* A. *Si dice che* R *è* SIMMETRICA *se per tutti gli elementi* $a, b \in A$, *vale che*

se $(a,b) \in R$, *allora* $(b,a) \in R$.

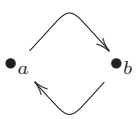
Esempio 3.2.11. *Per ogni insieme* A, *la relazione identità* $id_A \subseteq A \times A$, *la relazione completa* $A \times A \subseteq A \times A$ *e la relazione vuota* $\varnothing \subseteq A \times A$ *sono simmetriche.*

Esempio 3.2.12. *Le relazioni* $< \subseteq \mathbb{N} \times \mathbb{N}$ *e* $\leq \subseteq \mathbb{N} \times \mathbb{N}$ *non sono simmetriche.*

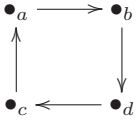
Esempio 3.2.13. *Per* $A = \{a,b,c,d\}$, *la relazione* $R = \{(a,b),\ (b,c),\ (a,d)\}$ *non è simmetrica, mentre la relazione* $S = \{(a,b),\ (b,c),\ (a,d),\ (b,a),\ (c,b),\ (d,a)\}$ *lo è.*

Come per le relazioni riflessive e transitive, è conveniente identificare le relazioni simmetriche attraverso i diagrammi: $R \subseteq A \times A$ è simmetrica se e solo se ogni volta che c'è un arco da a a b (per tutti i nodi a, b)

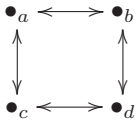
c'è anche un arco da b ad a



Per esempio, la relazione



non è simmetrica, mentre la relazione



è simmetrica.

Consideriamo adesso l'ultima proprietà.

Definizione 3.2.14. *Sia $R \subseteq A \times A$ una relazione su A. Si dice che R è* ANTI-SIMMETRICA *se per tutti gli elementi $a, b \in A$, vale che*

$$\text{se } (a,b) \in R \text{ e } (b,a) \in R, \text{ allora } a = b.$$

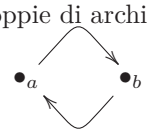
Esempio 3.2.15. *Per ogni insieme A, la relazione identità $id_A \subseteq A \times A$ e la relazione vuota $\varnothing \subseteq A \times A$ sono anti-simmetriche. La relazione completa $A \times A \subseteq A \times A$ solitamente non è anti-simmetrica.*

Esercizio 3.2.16. *Per quali insiemi A la relazione completa $A \times A \subseteq A \times A$ è anti-simmetrica?*

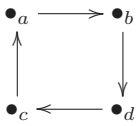
Esempio 3.2.17. *Le relazioni $< \subseteq \mathbb{N} \times \mathbb{N}$ e $\leq \subseteq \mathbb{N} \times \mathbb{N}$ sono anti-simmetriche.*

Esempio 3.2.18. *Per $A = \{a, b, c, d\}$, la relazione $R = \{(a,b),\ (b,a),\ (a,d)\}$ non è anti-simmetrica, mentre la relazione $S = \{(a,b),\ (a,d)\}$ lo è.*

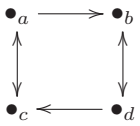
È molto facile identificare le relazioni anti-simmetriche attraverso i diagrammi: $R \subseteq A \times A$ è anti-simmetrica se e solo se non ci sono coppie di archi



Per esempio, la relazione



è anti-simmetrica, mentre la relazione



non lo è.

Teorema di caratterizzazzione

Come per le proprietà viste nella Sezione 2.4, anche le relazioni riflessive, transitive, simmetriche e anti-simmetriche possono essere caratterizzate attraverso le operazioni su relazioni.

Teorema 3.2.19. *Per tutti gli insiemi A e per tutte le relazioni su A $R \subseteq A \times A$ vale che:*

3.2.19.1. *R è riflessiva se e solo se $id_A \subseteq R$;*

3.2.19.2. *R è transitiva se e solo se $R; R \subseteq R$;*

3.2.19.3. *R è simmetrica se e solo se $R^{op} \subseteq R$;*

3.2.19.4. *R è anti-simmetrica se e solo se $R \cap R^{op} \subseteq id_A$;*

Dimostrazione. Proviamo solo il punto 2. e lasciamo gli altri per esercizio.

$$
\begin{gathered}
R; R \subseteq R \\
\text{se e solo se} \\
\text{per tutti gli } (a,c) \in A \times A, \text{ se } (a,c) \in R; R \text{ allora } (a,c) \in R \\
\text{se e solo se} \\
\text{per tutti gli } (a,c) \in A \times A, \text{ se } \exists b \in A \,.\, (a,b) \in R \wedge (b,c) \in R \text{ allora } (a,c) \in R \\
\text{se e solo se} \\
R \text{ è transitiva.}
\end{gathered}
$$

■

Esercizio 3.2.20. *Dimostrare i punti 1, 3 e 4 del Teorema 3.2.19.*

Esercizio 3.2.21. *Usando il Teorema 3.2.19, dimostrare che per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$ vale che:*

$$R \text{ è simmetrica se e solo se } R \subseteq R^{op}$$

Esercizio 3.2.22. *Usando il Teorema 3.2.19, dimostrare che per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$ vale che:*

$$R \text{ è simmetrica se e solo se } R = R^{op}$$

3.3 Chiusure

Chiusura riflessiva

Data una relazione $R \subseteq A \times A$ è sempre possibile trasformarla in una relazione riflessiva seguendo una ricetta standard.

Definizione 3.3.1. *Sia $R \subseteq A \times A$ una relazione su A. La* CHIUSURA RIFLESSIVA DI *R è la relazione $R \cup id_A$.*

Esempio 3.3.2. *Per $A = \{a,b,c,d\}$, la relazione $R = \{(a,a),\ (b,b),\ (a,d)\}$ non è riflessiva. La chiusura riflessiva di R è la relazione $R \cup id_A = \{(a,a),\ (b,b),\ (c,c),\ (d,d),\ (a,d)\}$.*

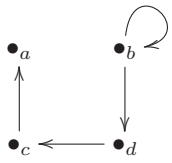
Esempio 3.3.3. *La relazione minore $<$ sui naturali, non è riflessiva. La chiusura riflessiva di $<$ è la relazione $< \cup id_{\mathbb{N}}$ che è esattamente la relazione $\leq$.*

Esempio 3.3.4. *La relazione vuota $\varnothing \subseteq A \times A$ non è riflessiva. La chiusura riflessiva di $\varnothing$ è la relazione id_A.*

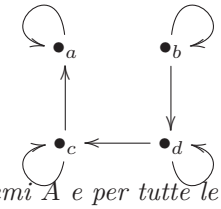
La chiusura riflessiva può essere visualizzata facilmente in modo grafico: è sufficiente inserire un arco

per ogni nodo $\bullet_x$.

Per esempio la chiusura riflessiva della relazione



è rappresentata dal seguente grafo.



Proposizione 3.3.5. *Per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$, vale che*

(1) *$R \cup id_A$ è riflessiva;*

(2) $R \subseteq R \cup id_A$;

(3) *per tutte le relazioni* $S \subseteq A \times A$, *se* $R \subseteq S$ *e* S *è riflessiva, allora* $R \cup id_A \subseteq S$.

Tale proposizione ci spiega il significato della parola *chiusura*: la chiusura riflessiva di R (1) è riflessiva, (2) contiene R e (3) è la più "piccola" tra tutte le relazioni che soddisfano (1) e (2).

Esercizio 3.3.6. *Dimostrare la Proposizione 3.3.5.*

Chiusura simmetrica

Per le relazioni simmetriche si può procedere in modo del tutto analogo.

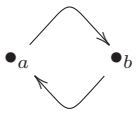
Definizione 3.3.7. *Sia* $R \subseteq A \times A$ *una relazione su* A. *La* CHIUSURA SIMMETRICA DI R *è la relazione* $R \cup R^{op}$.

Esempio 3.3.8. *Per* $A = \{a, b, c, d\}$, *la relazione* $R = \{(a, b),\ (b, c),\ (a, d)\}$ *non è simmetrica. La sua chiusura simmetrica è* $R \cup R^{op} = \{(a, b),\ (b, c),\ (a, d),\ (b, a),\ (c, b),\ (d, a)\}$.

Esempio 3.3.9. *La relazione minore o uguale* $\leq$ *sui naturali non è simmetrica. La chiusura simmetrica di* $\leq$ *è la relazione* $\leq \cup \leq^{op}$ *che è esattamente la relazione completa* $\mathbb{N} \times \mathbb{N} \subseteq \mathbb{N} \times \mathbb{N}$.

La chiusura simmetrica può essere costruita facilmente in modo grafico: è sufficiente aggiungere, per ogni arco da $\bullet_a$ a $\bullet_b$,

un arco da $\bullet_b$ ad $\bullet_a$



Proposizione 3.3.10. *Per tutti gli insiemi* A *e per tutte le relazioni* $R \subseteq A \times A$, *vale che*

(1) $R \cup R^{op}$ *è simmetrica;*

(2) $R \subseteq R \cup R^{op}$;

(3) *per tutte le relazioni* $S \subseteq A \times A$, *se* $R \subseteq S$ *e* S *è simmetrica, allora* $R \cup R^{op} \subseteq S$.

Quindi la chiusura simmetrica di R (1) è simmetrica, (2) contiene R e (3) è la più "piccola" tra tutte le relazioni che soddisfano (1) e (2).

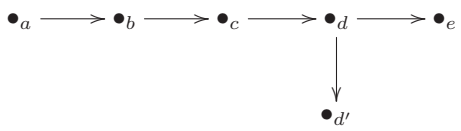
Esercizio 3.3.11. *Dimostrare la Proposizione 3.3.10.*

Chiusura transitiva

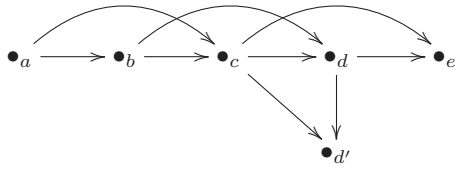
Per ottenere relazioni transitive, l'idea è di procedere in modo analogo alla chiusura simmetrica ed a quella riflessiva, ma alcune peculiarità rendono la procedura un po' più complessa.

Data una relazione $R \subseteq A \times A$, per trasformarla in una relazione transitiva, il primo tentativo consiste nell'aggiungere tutte le coppie $(a, c) \in A \times A$ tali che $\exists b \in B\,.\,(a, b) \in R \wedge (b, c) \in R$. In altre parole questo significa unire ad R la relazione $R; R$.

Per esempio, se R è la relazione

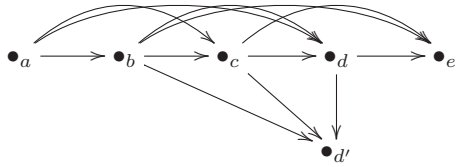


allora $R \cup (R; R)$ è la relazione



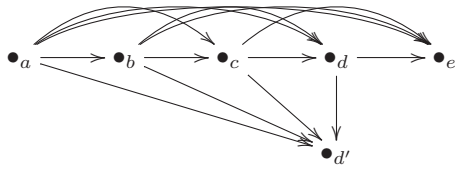
Si noti che tale relazione non è, come nei nostri desideri, transitiva: infatti $(b, c) \in R \cup (R; R)$ e $(c; d') \in R \cup (R; R)$, ma $(b, d') \notin R \cup (R; R)$.

Il secondo tentativo consiste nell'unire a $R \cup (R; R)$ la relazione $R; R; R$. La relazione $R \cup (R; R) \cup (R; R; R)$ è la seguente.



Ancora una volta la relazione ottenuta non è transitiva: $(a, b) \in R \cup (R; R) \cup (R; R; R)$ e $(b, d') \in R \cup (R; R) \cup (R; R; R)$ ma $(a, d') \notin R \cup (R; R) \cup (R; R; R)$.

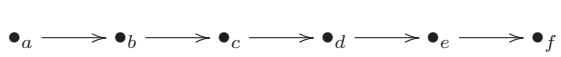
Nel terzo tentativo si unisce a $R \cup (R; R) \cup (R; R; R)$ la relazione $R; R; R; R$. La relazioni risultante $R \cup (R; R) \cup (R; R; R) \cup (R; R; R; R)$



è transitiva.

Osservazione 3.3.12. *È interessante notare che, facendo un quarto tentativo, vale a dire aggiungendo alla relazione ottenuta la relazione $R; R; R; R; R$, la relazione risultante resta la stessa.*

Si osservi che per la relazione R illustrata sopra sono stati necessari tre tentativi, ma è facile immaginare una relazione dove sono necessari più tentativi. Per esempio, per la relazione



occorrono quattro tentativi.

Come possiamo dare una ricetta generale per rendere una qualsiasi relazione R transitiva?

L'idea è di prendere una relazione ottenuta come unione infinita di tante relazioni:

$$R \cup (R; R) \cup (R; R; R) \cup (R; R; R; R) \cup \ldots \tag{3.1}$$

Per essere del tutto formali, procediamo in due passi. Innanzitutto forniamo una definizione *induttiva*:

Definizione 3.3.13. *Sia A un insieme e $R \subseteq A \times A$ una relazione su A. Per ogni numero naturale $n \in \mathbb{N}$ definiamo R^n induttivamente:*

$$R^0 = id_A \qquad R^{n+1} = R; R^n$$

Vedremo l'induzione nel dettaglio nel Capitolo 5, ma per il momento cerchiamo di capire la definizione di sopra: la relazione R^n è la composizione di R con se stessa n volte. Per esempio R^4 può essere calcolata come

$$\begin{array}{rcl} R^4 & = & R; R^3 \\ & = & R; (R; R^2) \\ & = & R; (R; (R; R^1)) \\ & = & R; (R; (R; (R; R^0))) \\ & = & R; (R; (R; (R; id_A))) \end{array}$$

che, per le leggi di associatività ed unità, è esattamente $R; R; R; R$.

Adesso per ottenere la relazione in (3.1) è sufficiente fare l'unione infinita degli R^n per tutti gli $n = 1, 2 \ldots$

$$\bigcup_{n \in \mathbb{N}^+} R^n$$

Definizione 3.3.14. *Sia $R \subseteq A \times A$ una relazione su A. La* CHIUSURA TRANSITIVA DI *R, denotata R^+, è la relazione definita come*

$$R^+ = \bigcup_{n \in \mathbb{N}^+} R^n$$

Esempio 3.3.15. *Calcoliamo la chiusura transitiva di id_A, $\varnothing \subseteq A \times A$ e $A \times A \subseteq A \times A$ per un qualsiasi insieme A.*

Per id_A si ha che

$$\begin{aligned} id_A^1 &= id_A; id_A^0 \\ &= id_A; id_A \\ &= id_A. \end{aligned}$$

Inoltre

$$\begin{aligned} id_A^2 &= id_A; id_A^1 \\ &= id_A; id_A \\ &= id_A. \end{aligned}$$

Similmente, $id_A^3 = id_A; id_A^2 = id_A; id_A = id_A$. Utilizzando l'induzione (che vedremo nel Capitolo 5), è facile dimostrare che $id_A^n = id_A$ per ogni $n \in \mathbb{N}^+$. Pertanto

$$id_A^+ = \bigcup_{n \in \mathbb{N}^+} id_A^n = id_A.$$

Per $\varnothing \subseteq A \times A$ e $A \times A \subseteq A \times A$, possiamo convincersi attraverso un simile ragionamento che $\varnothing^n = \varnothing$ e $(A \times A)^n = A \times A$ per tutti gli $n \in \mathbb{N}^+$. Pertanto

$$\varnothing^+ = \bigcup_{n \in \mathbb{N}^+} \varnothing^n = \varnothing \text{ e}$$

$$(A \times A)^+ = \bigcup_{n \in \mathbb{N}^+} (A \times A)^n = A \times A.$$

Osservazione 3.3.16. *È importante fare attenzione a non confondere la notazione R^n per una qualche relazione R con A^n per un insieme A. La prima, data nella Definizione 3.3.13, denota la relazione ottenuta componendo R con se stessa n volte. La seconda, data nella Definizione 2.7.2, denota l'insieme delle sequenze su A di lunghezza n.*

Esempio 3.3.17. *Si consideri la relazione $R = \{(a, b),\ (b, c),\ (a, d)\}$ sull'insieme $A = \{a, b, c, d\}$. Per calcolare R^+ è conveniente analizzare le relazioni R^n per $n \in \mathbb{N}^+$. Si ha che:*

$$\begin{aligned} R^1 &= R; R^0 \\ &= R; id_A \\ &= R \end{aligned} \qquad \begin{aligned} R^2 &= R; R^1 \\ &= R; R \\ &= \{(a, c)\} \end{aligned} \qquad \begin{aligned} R^3 &= R; R^2 \\ &= R; \{(a, c)\} \\ &= \varnothing \end{aligned}$$

Visto che $R^3 = \varnothing$, si ha che $R^4 = R; R^3 = R; \varnothing = \varnothing$ e, più in generale, $R^n = \varnothing$ per tutti gli $n \geq 3$. Pertanto

$$R^+ = \bigcup_{n \in \mathbb{N}^+} R^n = R^1 \cup R^2 \cup \varnothing \cup \varnothing \cup \cdots \cup \varnothing \cup \ldots$$

e quindi

$$R^+ = \{(a, b),\ (b, c),\ (a, d),\ (a, c)\}.$$

Aggiungendo una sola coppia alla relazione R il calcolo di R^+ può essere molto più complicato, come vediamo nel seguente esempio.

Esempio 3.3.18. *Si consideri la relazione* $R = \{(a,b),\ (b,c),\ (a,d),\ (c,a)\}$ *sull'insieme* $A = \{a,b,c,d\}$. *Per calcolare* R^+, *è conveniente analizzare le relazioni* R^n *per* $n \in \mathbb{N}^+$. *Si ha che* $R^1 = R; R^0 = R; id_A = R$. *Inoltre,*

$$\begin{array}{rcl} R^2 & = & R; R^1 \\ & = & R; R \\ & = & \{(a,c),\ (b,a),\ (c,b),\ (c,d)\} \end{array}$$

$$\begin{array}{rcl} R^3 & = & R; R^2 \\ & = & R; \{(a,c),\ (b,a)\ (c,b),\ (c,d)\} \\ & = & \{(a,a),\ (b,b),\ (c,c),\ (b,d)\} \end{array}$$

È importante osservare che R^4 *risulta essere inclusa nella relazione di partenza* R.

$$\begin{array}{rcl} R^4 & = & R; R^3 \\ & = & R; \{(a,a),\ (b,b),\ (c,c),\ (b,d)\} \\ & \subseteq & R; (id_A \cup \{(b,d)\}) \\ & = & (R; id_A) \cup (R; \{(b,d)\}) \\ & = & R \cup \{(a,d)\} \\ & = & R \end{array}$$

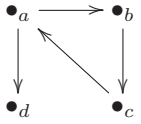
A questo punto non dobbiamo calcolare nuove relazioni perchè ognuno degli R^n *risulterà essere incluso o in* R *o in* R^2 *o in* R^3. *Per esempio,* $R^5 = R; R^4 \subseteq R; R = R^2$, $R^6 = R; R^5 \subseteq R; R^2 = R^3$ *e* $R^7 = R; R^6 \subseteq R; R^3 = R$. *Pertanto*

$$\begin{array}{rcl} R^+ = \bigcup_{n\in\mathbb{N}^+} R^n & = & R^1 \cup R^2 \cup R^3 \cup R^4 \cup R^5 \cup R^6 \cup R^7 \cup \ldots \\ & \subseteq & R^1 \cup R^2 \cup R^3 \cup R^1 \cup R^2 \cup R^3 \cup R^1 \cup \ldots \\ & = & R_1 \cup R_2 \cup R_3 \end{array}$$

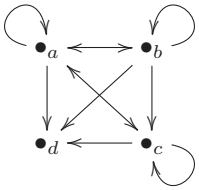
e quindi $R^+ =$

$$\{(a,b),\ (b,c),\ (a,d),\ (c,a),\ (a,c),\ (b,a),\ (c,b),\ (c,d),\ (a,a),\ (b,b),\ (c,c),\ (b,d)\}$$

Un modo efficace per semplificare il calcolo di R^+ consiste nel visualizzare la relazione R come un grafo ed aggiungere un arco da un nodo $\bullet_x$ ad un nodo $\bullet_y$ ogni volta che c'è un "percorso" da $\bullet_x$ a $\bullet_y$. Qua, per "percorso", si intende una sequenza (di lunghezza arbitraria) di archi. Per esempio la relazione R dell'Esempio 3.3.18 è disegnata come il seguente grafo.



Aggiungendo un arco per ogni percorso, si ottiene la relazione R^+.



Proposizione 3.3.19. *Per tutti gli insiemi* A *e per tutte le relazioni* $R \subseteq A \times A$, *vale che*

(1) R^+ *è transitiva;*

(2) $R \subseteq R^+$;

(3) *per tutte le relazioni* $S \subseteq A \times A$, *se* $R \subseteq S$ *e* S *è transitiva, allora* $R^+ \subseteq S$.

Analogamente alla chiusura riflessiva e simmetrica, la chiusura transitiva di R è la più "piccola" relazione transitiva che contiene R.

Esercizio 3.3.20. *Dimostrare la Proposizione 3.3.19.*

Esercizio 3.3.21. *Si ricordi la relazione Genitore $\subseteq U \times U$. Definire per proprietà (in modo intensionale) la relazione Genitore$^+$.*

Esercizio 3.3.22. *Si ricordi la relazione succ: $\mathbb{N} \to \mathbb{N}$. Definire per proprietà (in modo intensionale) la relazione succ$^+$.*

Esercizio 3.3.23. *Dimostrare che per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$, vale che*

$$\text{se } R \text{ è transitiva, allora } R^+ = R.$$

Suggerimento: utilizzare la Proposizione 3.3.19.

Esercizio 3.3.24. *Si ricordi la relazione minore $< \subseteq \mathbb{N} \times \mathbb{N}$. Definire per proprietà (in modo intensionale) la relazione $<^+$.*

Stella di Kleene

Nella definizione di R^+ si prende l'unione di tutti gli R^n per $n \in \mathbb{N}^+$. Cambiando la definizione in modo da prendere $n \in \mathbb{N}$, si ottiene la relazione $R^0 \cup R^+ = id_A \cup R^+$ che chiamiamo chiusura riflessiva e transitiva.

Definizione 3.3.25. *Sia $R \subseteq A \times A$ una relazione su A. La* CHIUSURA RIFLESSIVA E TRANSITIVA DI *R, denotata $R^\star$, è la relazione definita come*

$$R^\star = \bigcup_{n \in \mathbb{N}} R^n$$

Proposizione 3.3.26. *Per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$, vale che*

- *$R^\star$ è riflessiva e transitiva;*
- *$R \subseteq R^\star$;*
- *per tutte le relazioni $S \subseteq A \times A$, se $R \subseteq S$ e S è riflessiva e transitiva, allora $R^\star \subseteq S$.*

In altre parole, la chiusura riflessiva e transitiva di R è la più "piccola" relazione riflessiva e transitiva che contiene R.

Esercizio 3.3.27. *Si ricordi la relazione succ: $\mathbb{N} \to \mathbb{N}$. Definire per proprietà (in modo intensionale) la relazione succ*.*

Esercizio 3.3.28. *Si ricordi la relazione minore $< \subseteq \mathbb{N} \times \mathbb{N}$. Definire per proprietà (in modo intensionale) la relazione $<^\star$.*

La stella di Kleene ricopre un ruolo importante in Informatica in quanto $R^\star$ è spesso pensata come una sorta di *iterazione illimitata* di R. Per rendere questa intuizione più chiara è conveniente mostrare un esempio

Esempio 3.3.29. *Consideriamo un banalissimo programma while:*

$$\texttt{while } (x \geq 2) \texttt{ do } \{x\texttt{:=}\ x - 2\}$$

Tale programma prende in input un qualsiasi numero intero $x \in \mathbb{Z}$ e restituisce x se $x < 2$, 0 se x è pari e 1 se x è dispari. Ad esempio, partendo con $x = 6$, dopo una prima iterazione x varrà 4, poi 2 e poi 0. Con $n = 0$, il corpo del ciclo non viene eseguito perché la condizione della guardia $x \geq 2$ non è soddisfatta. Partendo con $x = 7$, nelle iterazioni successive x varrà 5, poi 3 e poi infine 1. Partendo con $x = -1$, la guardia del while è falsa e quindi non viene eseguita nessuna iterazione del ciclo while.

Il comportamento di tale programma può essere espresso utilizzando la stella di Kleene. Per rappresentare il corpo del ciclo while (il comando $x\texttt{:=}\ x-2$) definiamo la relazione $C = \{(x, x-2) \mid x \in \mathbb{Z}\} \subseteq \mathbb{Z} \times \mathbb{Z}$. Per rappresentare la condizione di guardia ($x \geq 2$), definiamo la relazione

$G = \{(x,x) \mid x \in \mathbb{Z} \text{ e } x \geq 2\} \subseteq \mathbb{Z} \times \mathbb{Z}$ *(si osservi che* $G \subseteq Id_{\mathbb{Z}}$*, quindi* G *è la* relazione di identità parziale *definita solo per i valori che soddisfano la guardia). Inoltre definiamo* $\tilde{G}$ *come* $\tilde{G} = Id_{\mathbb{Z}} \setminus G$*, quindi* $\tilde{G} = \{(x,x) \mid x \in \mathbb{Z} \text{ e } x < 2\}$.

Adesso il "comportamento" del banale programma di sopra può essere espresso come la relazione

$$W = (G;C)^\star;\tilde{G}. \tag{3.2}$$

Mostriamo che per ogni coppia $(a,b) \in \mathbb{Z}\times\mathbb{Z}$*,* $(a,b) \in W$ *se e solo se il programma while eseguito a partire da uno stato in cui* $x = a$ *termina in uno stato in cui* $x = b$.

Dalla definizione della stella di Kleene e usando la distributività della composizione rispetto all'unione (Tabella 2.6) abbiamo che

$$W = (Id_{\mathbb{Z}} \cup (G;C)^+);\tilde{G} = (Id_{\mathbb{Z}};\tilde{G}) \cup ((G;C)^+;\tilde{G}) = \tilde{G} \cup ((G;C)^+;\tilde{G})$$

Quindi per ogni valore $a < 2$ *vale* $(a,a) \in \tilde{G} \subseteq W$*, come desiderato, perché per tali valori il ciclo non viene mai eseguito e* x *mantiene il suo valore.*

Adesso per ogni $n \in \mathbb{N}^+$ *è facile convincersi che* $(G;C)^n$ *conterrà tutte e sole le coppie* (a,b) *dove* a *è il valore iniziale di* x *e* b *è il valore di* x *dopo l'*n*-sima iterazione. Ad esempio*

$$\begin{array}{lcllcl} (6,4) & \in & (G;C)^1 & (7,5) & \in & (G;C)^1 \\ (6,2) & \in & (G;C)^2 & (7,3) & \in & (G;C)^2 \\ (6,0) & \in & (G;C)^3 & (7,1) & \in & (G;C)^3 \end{array}$$

Si osservi che $(6,-2) \notin (G;C)^4$ *in quanto* $(0,-2) \notin G;C$*. Di conseguenza per* $n > 3$*, non esiste alcuna coppia* $(a,b) \in (G;C)^n$ *dove il primo elemento* a *è* 6*. Pertanto le uniche coppie in* $(G;C)^+$ *aventi come primo elemento* 6 *sono esattamente quelle elencate sopra. Lo stesso vale per* 7.

Per concludere, ricordando che $W = (G;C)^*;\tilde{G}$*, osserviamo che di tutte le coppie in* $(G;C)^+$ *aventi 6 come primo elemento, l'unica che contribuisce alla relazione* W *è* $(6,0)$*, perché* $(0,0) \in \tilde{G}$ *mentre* $(b,b) \notin \tilde{G}$ *per* $b \in \{2,4\}$*. Analogamente* $(7,1) \in W$ *e* $(7,b) \notin W$ *per* $b \neq 1$*. Poiché lo stesso ragionamento può essere fatto per ogni numero pari o dispari maggiore o uguale a 2, questo conclude la dimostrazione.*

Osservazione 3.3.30. *Il lettore potrebbe adesso pensare che ogni programma while può essere espresso come la relazione in* (3.2)*. È quindi opportuno precisare che l'espressione in* (3.2) *è solo un esempio ad hoc, e non vuole rappresentare una ricetta generale per esprire programmi while utilizzando la stella di Kleene e le operazioni su relazioni. Questo infatti è un problema abbastanza complesso che non può essere affrontato in questo corso introduttivo. Lo studente interessato potrà affrontare questo tipo di problematiche in corsi avanzati.*

Vista l'importanza della stella di Kleene, è opportuno studiare le leggi che la regolano.

Teorema 3.3.31. *Per tutti gli insiemi* A *e relazioni* $R \subseteq A \times A$ *e* $S \subseteq A \times A$ *valgono le uguaglianze nelle Tabelle 3.1.*

riflessività	$id_A \subseteq R^\star$
transività	$R^\star;R^\star \subseteq R^\star$
chiusura	$R \subseteq R^\star$
idempotenza	$(R^\star)^\star = R^\star$
$\star$-id	$id_A^\star = id_A$
$\star$-compl	$(A \times A)^\star = A \times A$
$\star$-vuoto	$\varnothing_{A,A}^\star = id_A$
distributività di $\star$ su $\cup$	$R^\star \cup S^\star \subseteq (R \cup S)^\star$
distributività di $\star$ su $\cap$	$(R \cap S)^\star \subseteq R^\star \cap S^\star$
distributività di $\star$ su $\cdot^{op}$	$(R^\star)^{op} = (R^{op})^\star$

Tabella 3.1: Leggi della stella di Kleene.

Le prime tre leggi (riflessività, transitività e chiusura) seguono immediatamente dalla Proposizione 3.3.26. Anche l'idempotenza segue dalla Proposizione 3.3.26 ma la dimostrazione richiede qualche passaggio.

Esercizio 3.3.32. *Dimostrare in maniera discorsiva la legge idempotenza nella Tabella 3.1.*

Per le leggi $\star$-id, $\star$-compl e $\star$-vuoto è sufficiente calcolare la chiusura riflessiva e transitiva di id_A, $\varnothing$ e $A \times A$.

Esercizio 3.3.33. *Calcolare $id_A^\star$, $\varnothing^\star$ e $(A \times A)^\star$, per un arbitrario insieme A.*

Le tre leggi di distributività alla fine della Tabella 3.1 richiedono invece l'utilizzo dell'induzione. Le riprenderemo nel Capitolo 5.

3.4 Relazioni di equivalenza

Definizione 3.4.1. *Sia $R \subseteq A \times A$ una relazione su A. Si dice che R è una* RELAZIONE DI EQUIVALENZA *se è riflessiva, transitiva e simmetrica.*

Esempio 3.4.2. *Per ogni insieme A, l'identità $id_A \subseteq A \times A$ e la relazione completa $A \times A \subseteq A \times A$ sono relazioni di equivalenza. La relazione vuota $\varnothing \subseteq A \times A$ in generale non è una relazione di equivalenza perchè non è riflessiva.*

Esempio 3.4.3. *Le relazioni $< \subseteq \mathbb{Z} \times \mathbb{Z}$ e $\leq \subseteq \mathbb{Z} \times \mathbb{Z}$ non sono relazioni di equivalenza, perchè non sono simmetriche.*

Esempio 3.4.4. *Per $A = \{a, b, c, d\}$, la relazione $R = \{(a, a),\ (a, b),\ (b, a),\ (c, c)\}$ non è una relazione di equivalenza perchè non è riflessiva mentre la relazione $S = \{(a, a),\ (b, b),\ (a, b),\ (b, a),\ (c, c),\ (d, d)\}$ è una relazione di equivalenza.*

Le relazioni di parentela sugli esseri umani solitamente non sono relazioni di equivalenza: queste relazioni molto spesso non sono né simmetriche né transitive (per esempio $Madre$, $Padre$, Zia). La relazione $Fratello \subseteq U \times U$ è sia simmetrica che transitiva, ma non è una relazione di equivalenza perché non è riflessiva: un essere umano non è solitamente considerato fratello di se stesso. Si può comunque definire una relazione di equivalenza molto simile.

Esempio 3.4.5. *La relazione*

$$StessaMadre = \{(x, y) \in U \times U \mid x \text{ e } y \text{ hanno la stessa madre}\}$$

è una relazione di equivalenza poiché (a) x e x hanno la stessa madre; (b) se x e y hanno la stessa madre e y e z hanno la stessa madre, allora anche x e z hanno la stessa madre; (c) se la madre di x è la madre di y, allora anche la madre di y è la madre di x.

Esercizio 3.4.6. *Dimostrare che le seguenti relazioni sono equivalenze.*

$$\begin{aligned} StessoCogn &= \{(x, y) \in U \times U \mid x \text{ e } y \text{ hanno lo stesso cognome}\} \\ StessoCompl &= \{(x, y) \in U \times U \mid x \text{ e } y \text{ hanno lo stesso compleanno}\} \\ StessiCapelli &= \{(x, y) \in U \times U \mid x \text{ e } y \text{ hanno lo stesso numero di capelli}\} \end{aligned}$$

Teorema di caratterizzazzione

Le relazioni nell'Esempio 3.4.5 e nell'Esercizio 3.4.6 seguono uno schema comune:

Definizione 3.4.7. *Sia $f\colon A \to B$ una funzione. La relazione* KERNEL DI *f è definita come*

$$Ker(f) = \{(x, y) \in A \times A \mid f(x) = f(y)\}.$$

Esempio 3.4.8. *Si consideri la funzione $capelli\colon U \to \mathbb{N}$ che associa ad ogni essere umano il suo numero di capelli. $Ker(capelli)$ è la relazione $StessiCapelli$ dell'Esercizio 3.4.6.*

Proposizione 3.4.9. *Per tutti gli insiemi A, B e per tutte le funzioni $f\colon A \to B$ vale che*

3.4.9.1. $Ker(f) = f; f^{op}$;

3.4.9.2. *$Ker(f)$ è una relazione di equivalenza.*

Dimostrazione. Per il primo punto si osservi che:

$$\begin{array}{lll} & (x,y) \in f; f^{op} & \\ \text{se e solo se} & \exists b \in B\,.\,(x,b) \in f \wedge (b,y) \in f^{op} & (\text{def di ;}) \\ \text{se e solo se} & \exists b \in B\,.\,(x,b) \in f \wedge (y,b) \in f & (\text{def di } f^{op}) \\ \text{se e solo se} & f(x) = f(y) & (f \text{ funzione}) \\ \text{se e solo se} & (x,y) \in Ker(f) & (\text{def. } Ker(f)\ ) \end{array}$$

Per il secondo punto, si dimostrano separatamente le tre proprietà per $f; f^{op}$.

- Riflessiva: $id_A \subseteq f; f^{op}$ segue subito dal fatto che che f è totale e dal Teorema 2.4.34.1.
- Transitiva: $(f; f^{op}); (f; f^{op}) = f; (f^{op}; f); f^{op} \subseteq f; f^{op}$ segue subito dal fatto che f è univalente e dal Teorema 2.4.34.2.
- Simmetrica: $f; f^{op} = (f; f^{op})^{op}$ segue subito dalle leggi di distributività e convoluzione di $\cdot^{op}$.

■

La Proposizione 3.4.9 può essere trasformata in un teorema di caratterizzazione: tutte le relazioni di equivalenza sono il kernel di una qualche funzione. Per dimostrare questo fatto usiamo la nozione di classe di equivalenza.

Definizione 3.4.10. *Sia $R \subseteq A \times A$ una relazione di equivalenza ed $a \in A$ un elemento di A. La* CLASSE DI R-EQUIVALENZA DI *a è definita come*

$$[a]_R = \{b \in A \mid (a,b) \in R\}$$

Esempio 3.4.11. *Sia S la relazione di equivalenza dell'Esempio 3.4.4. Si ha che $[a]_S = \{a,b\}$; $[b]_S = \{a,b\}$; $[c]_S = \{c\}$ e $[d]_S = \{d\}$.*

Esempio 3.4.12. *Per ogni insieme A si considerino le relazioni di equivalenza $id_A \subseteq A \times A$ e $A \times A \subseteq A \times A$. Per tutti gli $a \in A$, vale che*

$$\begin{array}{lcl} [a]_{id_A} & = & \{a\} \\ [a]_{A\times A} & = & A \end{array}$$

Proposizione 3.4.13. *Per tutti gli insiemi A, per tutte le relazioni di equivalenza $R \subseteq A \times A$, e per tutti gli elementi $a, b \in A$, le seguenti affermazioni sono equivalenti:*

3.4.13.1. $(a,b) \in R$

3.4.13.2. $[a]_R = [b]_R$

3.4.13.3. $[a]_R \cap [b]_R \neq \varnothing$

Dimostrazione. La dimostrazione procede dimostrando che:

- *L'affermazione 1 implica la 2:*

  Assumiamo che $(a,b) \in R$ e dimostriamo le due inclusioni $[a]_R \subseteq [b]_R$ e $[a]_R \supseteq [b]_R$. Facciamo vedere solo la prima, la seconda è del tutto analoga. Preso un qualsiasi elemento $c \in [a]_R$ dobbiamo far vedere che $c \in [b]_R$. Per definizione di classe di equivalenza sappiamo che $b \in [a]_R$. Dato che $c \in [a]_R$ si ha $(a,c) \in R$. Per la proprietà simmetrica delle relazioni di equivalenza si ha $(b,a) \in R$. Ma allora per la transitività deve valere $(b,c) \in R$ e quindi $c \in [b]_R$.

- *La 2 implica la 3:*

  Assumendo che $[a]_R = [b]_R$, per dimostrare che $[a]_R \cap [b]_R \neq \varnothing$ basta far vedere che $[a]_R \neq \varnothing$. Infatti, la proprietà riflessiva garantisce che $a \in [a]_R$.

- *La 3 implica la 1:*

  Assumiamo che $[a]_R \cap [b]_R \neq \varnothing$. Quindi esiste almeno un elemento $c \in [a]_R \cap [b]_R$, cioè tale che $(a, c) \in R$ e $(b, c) \in R$. Ma per la proprietà simmetrica si ha $(c, b) \in R$ e dunque per la transitiva si ha che $(a, b) \in R$.

■

Teorema 3.4.14. *Per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$ vale che R è una relazione di equivalenza se e solo se esiste un insieme B ed una funzione $f\colon A \to B$ tale che $R = Ker(f)$.*

Dimostrazione. Trattandosi di un se e solo se, possiamo dividere la dimostrazione in due parti.

- Se R è una relazione di equivalenza, allora esiste un insieme B ed una funzione $f\colon A \to B$ tale che $R = Ker(f)$.

  Prendiamo come B l'insieme di tutte le classi di R-equivalenza e come funzione $f\colon A \to B$ la funzione che associa ad ogni elemento $a \in A$ la sua classe di R-equivalenza, cioè

  $$a \mapsto [a]_R$$

  Dobbiamo dimostrare adesso che $R = Ker(f)$: $(x, y) \in Ker(f)$ se e solo se $f(x) = f(y)$ (per definizione di Ker) se e solo se $[x]_R = [y]_R$ (per definizione di f) se e solo se $(x, y) \in R$ (per la Proposizione 3.4.13).

- Se esiste un insieme B ed una funzione $f\colon A \to B$ (per un qualche insieme B) tale che $R = Ker(f)$, allora R è una relazione di equivalenza.

  Questo segue immediatamente dalla Proposizione 3.4.9.2.

■

Relazioni di equivalenza e partizioni

Data una relazione di equivalenza $R \subseteq A \times A$ si può considerare l'insieme delle classi di R-equivalenza

$$EC_R = \{[a]_R \mid a \in A\}.$$

Essendo ogni $[a]_R$ un insieme EC_R è un insieme di insiemi. Visto che gli elementi non vengono ripetuti negli insiemi, se $[a]_R = [b]_R$ allora $[a]_R$ e $[b]_R$ appariranno una volta sola in EC_R.

Esempio 3.4.15. *Sia S la relazione di equivalenza dell'Esempio 3.4.4. Si ha che $EC_S = \{\{a, b\}, \{c\}, \{d\}\}$.*

È importante notare che EC_R forma una *partizione* dell'insieme A.

Proposizione 3.4.16. *Per tutti gli insiemi A e per tutte le relazioni d'equivalenza $R \subseteq A \times A$, EC_R è una partizione.*

Dimostrazione. Per dimostrare che EC_R forma una partizione si deve dimostrare che

1. Ogni insieme $X \in EC_R$ è diverso da $\varnothing$ (insiemi non vuoti).
2. $\bigcup_{X \in EC_R} X = A$ (copertura di A);
3. Per tutti gli $X, Y \in EC_R$, se $X \neq Y$, allora $X \cap Y = \varnothing$ (insiemi disgiunti);

Per dimostrare il primo punto, si osservi che per ogni $[a]_R \in EC_R$, vale che $a \in [a]_R$ e quindi $[a]_R \neq \varnothing$.

Per dimostrare il secondo punto, si osservi che per ogni $a \in A$, vale che $a \in [a]_R$ e $[a]_R \in EC_R$. Pertanto $a \in \bigcup_{X \in EC_R} X$.

Per il terzo punto, dalla Proposizione 3.4.13, si ha che $[a]_R \neq [b]_R$ se e solo se $[a]_R \cap [b]_R = \varnothing$. ■

Viceversa, data una partizione dell'insieme A si può definire una relazione di equivalenza su A. Per mostrare questa costruzione procediamo per passi: prima si definisce una funzione e poi prendiamo il suo kernel.

Data una partizione $\mathcal{F} = \{ X_i \}_{i \in I}$ dell'insieme A, si definisce la relazione $f_{\mathcal{F}} \subseteq A \times I$ come

$$f_{\mathcal{F}} = \{(a, i) \in A \times I \mid a \in X_i\}$$

Proposizione 3.4.17. *Per tutti gli insiemi A e per tutte le partizioni $\mathcal{F} = \{ X_i \}_{i \in I}$ di A, la relazione $f_{\mathcal{F}}$ è una funzione.*

Dimostrazione. Bisogna dimostrare che $f_{\mathcal{F}}$ è totale e univalente.

- totale: visto che $\mathcal{F} = \{ X_i \}_{i \in I}$ è una partizione di A, vale che $A \subseteq \bigcup_{X \in EC_R} X$ e quindi per ogni $a \in A$ esiste almeno un X_i tale che $a \in X_i$ e quindi, per definizione di $f_{\mathcal{F}}$, si ha che $(a, i) \in f_{\mathcal{F}}$.

- univalente: visto che $\mathcal{F} = \{ X_i \}_{i \in I}$ è una partizione di A, vale che se $i \neq j$, allora $X_i \cap X_j = \varnothing$. Pertanto per ogni $a \in A$, esiste al più un $i \in I$, tale che $a \in X_i$, cioè esiste al più un $i \in I$, tale che $(x, i) \in f_{\mathcal{F}}$.

■

Adesso, la relazione di equivalenza corrispondente ad $\mathcal{F}$ è esattamente il kernel di $f_{\mathcal{F}}$.

Le corrispondenze che abbiamo illustrato definiscono una biiezione tra l'insieme delle relazioni di equivalenza su A (denotato da $ERel(A)$) e l'insieme delle partizioni su A (denotato da $Part(A)$).

Teorema 3.4.18. *Per tutti gli insiemi A, vale che*

$$ERel(A) \cong Part(A).$$

Dimostrazione. Definiamo $i \colon ERel(A) \to Part(A)$ come la funzione che mappa una relazione di equivalenza R nella partizione EC_R

$$\begin{array}{rccc} i \colon & ERel(A) & \to & Part(A) \\ & R & \mapsto & EC_R \end{array}$$

Definiamo $j \colon Part(A) \to ERel(A)$ come la funzione che mappa una partizione $\mathcal{F}$ nella relazione di equivalenza $Ker(f_{\mathcal{F}})$

$$\begin{array}{rccc} j \colon & Part(A) & \to & ERel(A) \\ & \mathcal{F} & \mapsto & Ker(f_{\mathcal{F}}) \end{array}$$

Dobbiamo dimostrare che $i; j = id_{ERel(A)}$ e $j; i = id_{Part(A)}$.

- $i; j = id_{ERel(A)}$. Dobbiamo dimostrare che per ogni relazione di equivalenza $R \subseteq A \times A$, $j(i(R)) = R$.

  Si osserva che $(x, y) \in j(i(R))$ se e solo se $(x, y) \in Ker(f_{EC_R})$ se e solo se $f_{EC_R}(x) = f_{EC_R}(y)$ se e solo se $[x]_R = [y]_R$ se e solo se $(x, y) \in R$.

- $j; i = id_{Part(A)}$. Dobbiamo dimostrare che per ogni partizione $\mathcal{F}$, $i(j(\mathcal{F})) = \mathcal{F}$.

  Si osserva che $X \in i(j(\mathcal{F}))$ se e solo se $X \in EC_{Ker(f_{\mathcal{F}})}$ se e solo se esiste un $a \in A$, tale che $X = [a]_{Ker(f_{\mathcal{F}})}$, se e solo se esiste un $a \in A$ tale che $X = [a]_{Ker(f_{\mathcal{F}})}$, se e solo se esiste un $a \in A$ tale che $X = \{b \in A \mid f_{\mathcal{F}}(b) = f_{\mathcal{F}}(a)\}$, se e solo se esite un $X_i \in \mathcal{F}$ tale che $X = X_i$.

■

3.5 Relazioni di Ordinamento

Definizione 3.5.1. *Sia $R \subseteq A \times A$ una relazione su A. Si dice che R è una* RELAZIONE DI ORDINAMENTO PARZIALE *se è riflessiva, transitiva ed anti-simmetrica.*

Esempio 3.5.2. *Per ogni insieme A, la relazione identità $id_A \subseteq A \times A$ è un ordinamento parziale. La relazione vuota $\varnothing \subseteq A \times A$ non è un ordinamento parziale perchè non è riflessiva. La relazione completa $A \times A \subseteq A \times A$ no, perché non è anti-simmetrica*

Esempio 3.5.3. *La relazione $< \subseteq \mathbb{N} \times \mathbb{N}$ non è un ordinamento parziale perché non è riflessiva mentre la relazione $\leq \subseteq \mathbb{N} \times \mathbb{N}$ è un ordinamento parziale.*

Esempio 3.5.4. *Per $A = \{a, b, c, d\}$, la relazione $R = \{(a, b),\ (b, c),\ (a, d)\}$ non è un ordinamento parziale perchè non è riflessiva e neppure transitiva, mentre la relazione $S = \{(a, b),\ (b, c),\ (a, c),\ (a, d),\ (a, a),\ (b, b),\ (c, c)\}$ è un ordinamento parziale.*

Proposizione 3.5.5. *Per ogni insieme A, l'inclusione su $\mathcal{P}(A)$*

$$\{(X, Y) \in \mathcal{P}(A) \times \mathcal{P}(A) \mid X \subseteq Y\}$$

è una relazione di ordinamento parziale.

Esercizio 3.5.6. *Dimostrare la Proposizione 3.5.5.*

Per le relazioni di ordinamento parziale spesso si usa la notazione infissa:

$$\text{si scrive } a\,R\,b \text{ per } (a, b) \in R.$$

Inoltre, le relazioni di ordinamento sono genericamente denotate dal simbolo $\sqsubseteq$. Molto spesso si utilizza il simbolo $\sqsubset$ per denotare la relazione

$$\sqsubset = \{(x, y) \mid x \sqsubseteq y \text{ e } x \neq y\}.$$

Si noti che questo è del tutto analogo alla notazione che si usa per le relazioni sui numeri *minore o uguale* ($\leq$) e *minore* ($<$).

C'è una differenza sostanziale tra l'ordinamento parziale $\leq$ sui numeri e gli altri ordinamenti parziali che abbiamo visto fin'ora. In $\leq$ vale che per ogni coppia di numeri $(n, m) \in \mathbb{N} \times \mathbb{N}$ o vale $n \leq m$ oppure vale $m \leq n$. Questa proprietà è importante in molte occasioni.

Definizione 3.5.7. *Sia $R \subseteq A \times A$ una relazione di ordinamento parziale su A. Si dice che R è un* ORDINAMENTO *se*

$$\textit{per tutti gli } (a, b) \in A \times A, \textit{ vale che } (a, b) \in R \textit{ oppure } (b, a) \in R.$$

Esercizio 3.5.8. *Mostrare che la relazione di ordinamento parziale della Proposizione 3.5.5 non è un ordinamento per tutti gli insiemi A.*

Esercizio 3.5.9 (Teorema di Caratterizzazione)**.** *Dimostrare che per tutti gli insiemi A e per tutte le relazioni $R \subseteq A \times A$, vale che*

$$R \textit{ è un ordinamento se e solo se } id_A \subseteq R,$$
$$R; R \subseteq R,\ R \cap R^{op} \subseteq id_A \textit{ e } A \times A \subseteq R \cup R^{op}.$$

Esercizio 3.5.10. *Siano A e B due insiemi e $\sqsubseteq_A \subseteq A \times A$ e $\sqsubseteq_B \subseteq B \times B$ due relazioni di ordinamento parziale. Si consideri ora la relazione $\sqsubseteq_{A \times B} \subseteq (A \times B) \times (A \times B)$ definita come*

$$(a_1, b_1) \sqsubseteq_{A \times B} (a_2, b_2) \textit{ se e solo se } a_1 \sqsubseteq_A a_2 \textit{ e } b_1 \sqsubseteq_B b_2.$$

Dimostrare che $\sqsubseteq_{A \times B}$ è una relazione di ordinamento parziale. Questa relazione è un ordinamento? In caso affermativo, dare una dimostrazione; in caso negativo fornire un controesempio.

Esercizio 3.5.11. *Siano A e B due insiemi e $\sqsubseteq_A \subseteq A \times A$ e $\sqsubseteq_B \subseteq B \times B$ due ordinamenti. Si consideri ora la relazione $\sqsubseteq_{A \times B} \subseteq (A \times B) \times (A \times B)$ definita come*

$$(a_1, b_1) \sqsubseteq_{A \times B} (a_2, b_2) \textit{ se e solo se } a_1 \sqsubset_A a_2 \quad \textit{oppure} \quad a_1 = a_2 \wedge b_1 \sqsubseteq_B b_2.$$

Dimostrare che $\sqsubseteq_{A \times B}$ è un ordinamento.

Ordinamento Lessicografico

Un esempio di ordinamento particolarmente importante è l'ordinamento lessicografico usato per ordinare le parole nei dizionari o negli elenchi.

Dato un insieme A ed un ordinamento $\sqsubseteq_A \subseteq A \times A$ si vuole definire un ordinamento $\sqsubseteq_{A^*} \subseteq A^* \times A^*$ sull'insieme delle stringhe su A.

L'idea è che l'ordinamento sul prodotto cartesiano visto nell'Esercizio 3.5.11 può essere esteso facilmente ad arbitrari insiemi A^n (l'insieme di sequenze su A di lunghezza n): date le sequenze $s = a_0 a_1 \dots a_n$ e $t = a'_0 a'_1 \dots a'_n$ in A^n si ha che

$$\begin{array}{l} s \sqsubseteq_{A^n} t \text{ se e solo se esiste un } i \in \{0, \dots, n\} \text{ tale che} \\ \text{per tutti gli indici } j < i \text{ vale che } a_j = a'_j, \text{ ed } a_i \sqsubset a'_i. \end{array}$$

Per avere un ordinameto sull'insieme di sequenze di lunghezza arbitaria (A^*), si deve solamente tenere conto della possibilità che le sequenze abbiano lunghezza diversa.

Definizione 3.5.12. *Sia $\sqsubseteq_A \subseteq A \times A$ un ordinamento sull'insieme A. L'*ORDINAMENTO LESSICOGRAFICO *$\sqsubseteq_{A^*} \subseteq A^* \times A^*$ è definito come segue.*

Per tutte le stringhe $s = a_0 a_1 \dots a_n$ e $t = a'_0 a'_1 \dots a'_m$ in A^* si ha che $s \sqsubseteq_{A^*} t$ se e solo se esiste un $i \in \mathbb{N}$ tale che per tutti i $j < i$, vale che $a_j = a'_j$ ed almeno una delle due seguenti condizioni è vera:

- *$a_i \sqsubset_A a'_i$;*
- *$i = n + 1$ e $n < m$.*

Osservazione 3.5.13. *Per chiarire la definizione può essere conveniente esprimerla attraverso la notazione matematica: $s \sqsubseteq_{A^*} t$ se e solo se*

$$\exists i \in \mathbb{N} \,.\, \forall j < i \,.\; a_j = a'_j \wedge (a_i \sqsubset a'_i \vee (i = n + 1 \wedge n < m)).$$

Esempio 3.5.14. *Si consideri l'insieme delle lettere dell'alfabeto italiano, ordinate nel modo classico:*

$$a \leq b \leq \dots \leq z$$

Vale che anna $\sqsubseteq$ zio: nella definizione si prenda $i = 0$ e si osservi che $a \sqsubset z$; anna $\sqsubseteq$ antonio: nella definizione si prenda $i = 2$ e si osservi che $an = an$ e $n \sqsubset t$; anna $\sqsubseteq$ annarella: nella definizione si prenda $i = 4$ e si osservi che $4 = 3 + 1$ e $3 < 8$.

Esercizio 3.5.15. *Considerare l'insieme delle cifre decimali ordinate nel modo ovvio*

$$0 \leq 1 \leq 2 \leq \dots \leq 9.$$

È vero che l'ordinmento lessicografico su questo insieme coincide con il consueto $\leq$ sui naturali? In caso affermativo, dare una dimostrazione; in caso negativo, fornire un controesempio.

CAPITOLO 4

Grafi

In questo capitolo descriviamo i grafi. Introdotti dal famoso matematico Eulero, la loro importanza informatica (e non solo) deriva dal fatto che una grande quantità di problemi computazionali può essere formulata mediante essi. Infatti i grafi aiutano a modellare relazioni tra coppie di elementi di un insieme, come rotte tra aeroporti o relazioni sociali. Possiamo immaginarli tramite un disegno, dove un insieme di elementi (nodi) è rappresentato da una serie di punti, cerchietti o identificatori, e le coppie di nodi che sono in relazione tra di loro sono collegate ciascuna da una linea (formando gli archi).

Il termine nodo deriva dall'idea che un grafo possa essere visto anche come una rete di corde, dove i nodi—ovvero i punti dove le corde si incontrano—rappresentano gli elementi dell'insieme, e gli archi sono segmenti di corda che rappresentano le relazioni tra nodi. Di fatti il termine rete viene a volte utilizzato come sinonimo di grafo: "la rete" (the web) è il nome dato all'enorme grafo di pagine multimediali a cui possiamo accedere tramite internet, e i cosidetti "social" sono noti come reti sociali, enfatizzando l'interconnessione sociale tra le persone. Vedremo in questo capitolo anche alcuni tipi particolari di grafi, con ulteriori esempi di grafi nel Capitolo 6.

4.1 Notazione di base

Definizione 4.1.1. *Un grafo $G = (V, E)$ è rappresentato da un insieme finito V di nodi o vertici e da un insieme finito $E \subseteq V \times V$ di archi o* lati.

Utilizzando la notazione $|X|$ per indicare la *cardinalità* o numero di elementi di un insieme finito X, assumiamo che i nodi siano etichettati con numeri consecutivi da 0 a $|V| - 1$, e indichiamo con $n = |V|$ il numero di nodi nel grafo G e con $m = |E|$ il suo numero di archi: la *dimensione* di G è quindi data dalla somma $n + m$.

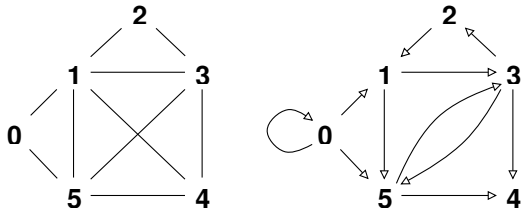


Figura 4.1: Un grafo non orientato (sinistra) e uno orientato (destra).

Nella terminologia dei capitoli precedenti, la relazione R su un insieme A (con $R \subseteq A \times A$) può essere vista come un grafo $G = (V, E)$, dove l'insieme dei nodi e degli archi sono rispettivamente definiti come $V = A$ e $E = R$, solo che invece di usare lettere per indicare gli elementi di A, usiamo numeri per indicare gli elementi di V, come mostrato a destra nella Figura 4.1.

A questo proposito è bene fare subito una distinzione riguardo agli archi. Presa una coppia $(x, y) \in R$, abbiamo visto nei capitoli precedenti che non è necessariamente vero che anche (y, x) appartenga a R, pur essendo entrambi $x, y \in A$. Passando alla notazione $G = (V, E)$ per i grafi,

l'arco $(x, y) \in E$ si dice *orientato* perché ha un verso di percorrenza (da x a y come i sensi unici nella rete stradale). I cappi (o loop) sono archi della forma (x, x).

Esempio 4.1.2. *Si consideri l'esempio in Figura 4.1. Nel grafo orientato di destra abbiamo* $V = \{0, 1, 2, 3, 4, 5\}$*, quindi ci sono* $n = 6$ *nodi (i numeri rappresentano i nodi puntiformi),* $m = 11$ *archi: l'insieme degli archi è* $E = \{(0, 0), (0, 1), (0, 5), (1, 3), (1, 5), (2, 1), (3, 2), (3, 4), (3, 5), (5, 3), (5, 4)\}$*. Notare come nello stesso grafo orientato possano essere presenti gli archi* $(3, 5)$ *e* $(5, 3)$*, perché hanno direzione opposta, e come l'arco* $(0, 0)$ *sia un cappio.*

Esite però un'altra tipologia di grafi, dove il verso degli archi non è rilevante (non ci sono frecce che ne indicano la direzione): in tal caso, è preferibile indicare l'arco con la notazione insiemistica $\{x, y\} \in E$ oppure $xy \in E$, in cui l'ordine di x e y non è rilevante. Si noti che qui i cappi (o loop) xx non vengono utilizzati nella loro forma non orientata.

Esempio 4.1.3. *Nell'esempio di Figura 4.1, il grafo non orientato di sinistra ha* $V = \{0, 1, 2, 3, 4, 5\}$*, quindi ci sono* $n = 6$ *nodi e* $m = 10$ *archi: l'insieme degli archi è* $E = \{01, 05, 12, 13, 14, 15, 23, 34, 35, 45\}$*. Si noti che* $\{1, 5\}$*,* $\{5, 1\}$*,* 15*,* 51 *sono tutte notazioni alternative per indicare il medesimo arco (che collega* 1 *e* 5*). Inoltre, non ci sono cappi in questo tipo do grafi.*

Riassumendo, utilizzeremo (x, y) per rappresentare un arco orientato *da* x *verso* y; prendendo come esempio le reti sociali, questo rappresenta una relazione non simmetrica come il "segui" su Twitter. Useremo invece l'insieme $\{x, y\}$ oppure la notazione semplificata xy per rappresentare un arco non orientato, ovvero una relazione simmetrica come l'amicizia su Facebook. Osserviamo che $\{x, y\}$ e $\{y, x\}$ rappresentano lo stesso legame tra x e y (e ciò chiaramente vale anche per xy e yx).[1] Questa terminologia si trasferisce al grafo G che sarà quindi orientato o meno; nel resto del capitolo, quando non specificato o chiaro dal contesto, un grafo si intende *non orientato senza cappi.*

4.2 Vicinato e grado dei nodi

Prendiamo in considerazione un grafo $G = (V, E)$. Due nodi $x, y \in V$ vengono detti *vicini* o *adiacenti* se connessi da un arco xy: tale arco si dice *incidente* a x e y, i quali sono chiamati *estremi* dell'arco.

Definizione 4.2.1. *L'insieme* $N(x) = \{y \mid xy \in E\}$ *dei nodi adiacenti a* x *viene detto il vicinato di* x*, e* x *si dice nodo universale quando è vicino a tutti (cioè* $N(x) \cup \{x\} = V$*) e nodo isolato quando* $N(x)$ *è vuoto. Il grado* $d_x = |N(x)|$ *di un nodo* x *è il numero di nodi a lui vicini (quindi* $d_x = 0$ *se il nodo è isolato). Con* Δ *si rappresenta il grado massimo in* G *(cioè* $\Delta = \max\{d_x \mid x \in V\}$*).*[2]

Proposizione 4.2.2 (Hand-shaking lemma (stretta di mano))**.** *Per ogni grafo non orientato* $G = (V, E)$*, vale che*

$$\sum_{x \in V} d_x = 2|E|$$

ovvero la somma dei gradi dei nodi è il doppio del numero degli archi. Inoltre, G *contiene un numero pari di nodi che hanno grado dispari (anche zero va bene).*

Esempio 4.2.3. *Prima di dimostrare la proposizione, consideriamo il grafo a sinistra nella Figura 4.1, dove* $|V| = 6$ *ed* $|E| = 10$*. Si noti che i vicinati sono* $N(0) = \{1, 5\}$*,* $N(1) = \{0, 2, 3, 4, 5\}$*,* $N(2) = \{1, 3\}$*,* $N(3) = \{1, 2, 4, 5\}$*,* $N(4) = \{1, 3, 5\}$*,* $N(5) = \{0, 1, 3, 4\}$*, e il nodo* 1 *è universale. La somma dei gradi è* $d_0 + d_1 + d_2 + d_3 + d_4 + d_5 = 2 + 5 + 2 + 4 + 3 + 4 = 20 = 2|E|$*; inoltre ci sono 2 nodi con gradi dispari (*d_1 *e* d_4*).*

Possiamo ora dimostrare la Proposizione 4.2.2.

Dimostrazione. L'intuizione è la seguente guardando al nostro esempio: l'arco $35 \in E$ incide sia sul nodo 3 che sul nodo 5 e quindi contribuisce al grado di entrambi. Per dimostrare la

[1] In inglese, spesso si usa "edge" per arco non orientato e "arc" per arco orientato, ma in italiano questa terminologia non è consolidata, anche se talvolta si usa lato per "edge".

[2] La lettera N sta per "neighborhood" e la lettera d per "degree", gli equivalenti termini inglesi.

proprietà prendiamo un arco qualsiasi $ij \in E$: poiché ij è generico e viene scelto arbitrariamente, questo vuol dire che il ragionamento seguente vale per *tutti* gli archi $ij \in E$. Osserviamo che $i \in N(j)$ e $j \in N(i)$, e ciò implica che ij contribuisce sia a $d_j = |N(j)|$ che a $d_i = |N(i)|$. Se prendiamo ora la somma dei gradi $\sum_{x \in V} d_x$, l'arco ij contribuisce due volte a tale somma: quando sommiamo d_j e quando sommiamo d_i. Pertanto la somma $\sum_{x \in V} d_x$ può essere scritta come $\sum_{x \in V} |N(x)| = \sum_{x \in V} \sum_{xy \in N(x)} 1 = \sum_{ij \in E} 2 = 2|E|$, in quanto fare la somma dei gradi è equivalente a contare due volte ogni arco. Infine, poiché tale somma è pari ($2|E|$), si può ottenere solo sommando un qualunque numero di gradi pari (anche zero), e un numero *pari* (anche zero) di gradi dispari, altrimenti la somma risultante sarebbe dispari. ■

È naturale chiedersi cosa succede per un grafo orientato $G = (V, E)$. Avendo gli archi una direzione, avremo gli archi *entranti* $(y, x) \in E$ in ogni nodo x e i suoi archi *uscenti* $(x, y) \in E$ (si noti l'ordine di x e y negli archi). Questo si riflette nelle definizioni di vicinato e grado.

Definizione 4.2.4. *L'insieme $N^+(x) = \{y \mid (x, y) \in E\}$ dei nodi rappresenta il vicinato in uscita di x (anche detto stella uscente perché formato dai suoi archi uscenti), e $N^-(x) = \{y \mid (y, x) \in E\}$ rappresenta il vicinato in ingresso di x (anche detto stella entrante perché formato dai suoi archi entranti). Il grado d'ingresso è $d_x^- = |N^-(x)|$ e il grado d'uscita è $d_x^+ = |N^+(x)|$.*

Proposizione 4.2.5. *Per ogni grafo orientato $G = (V, E)$, vale che*

$$\sum_{x \in V} d_x^- = \sum_{x \in V} d_x^+ = |E|$$

Dimostrazione. Preso un arco arbitrario $(i, j) \in E$, questo contribuisce sia a d_j^- che a d_i^+. Facendo le due somme di tali gradi, sono entrambe uguali a $|E|$. ■

Esempio 4.2.6. *Consideriamo il grafo orientato a destra nella Figura 4.1, dove $|V| = 6$ ed $|E| = 11$. Si noti che i vicinati in ingresso sono $N^-(0) = \{0\}$, $N^-(1) = \{0, 2\}$, $N^-(2) = \{3\}$, $N^-(3) = \{1, 5\}$, $N^-(4) = \{3, 5\}$, $N^-(5) = \{0, 1, 3\}$, e i vicinati in uscita sono $N^+(0) = \{0, 1, 5\}$, $N^+(1) = \{3, 5\}$, $N^+(2) = \{1\}$, $N^+(3) = \{2, 4, 5\}$, $N^+(4) = \{\}$, $N^+(5) = \{3, 4\}$. Come si può facilmente verificare, la somma dei gradi in ingresso è uguale alla somma dei gradi in uscita, entrambe uguali a $|E|$.*

Preso un grafo orientato $G = (V, E)$, ricordiamo che E è una relazione su V. È interessante collegare le quattro principali proprietà delle relazioni a condizioni sui gradi d'ingresso e d'uscita di tutti i nodi, come elencato sotto e mostrato nella Tabella 4.1:

1. E è totale se e solo se $d_x^+ \geq 1$ per ogni $x \in V$;
2. E è univalente se e solo se $d_x^+ \leq 1$ per ogni $x \in V$;
3. E è iniettiva se e solo se $d_x^- \leq 1$ per ogni $x \in V$;
4. E è surgettiva se e solo se $d_x^- \geq 1$ per ogni $x \in V$.

	$\forall x \in V.\ d_x^+ \cdots$	$\forall x \in V.\ d_x^- \cdots$
$\cdots \geq 1$	TOTALE	SURGETTIVA
$\cdots \leq 1$	UNIVALENTE	INIETTIVA

Tabella 4.1: Tabella riassuntiva per collegare le quattro proprietà della relazione E su V con i gradi d'ingresso e d'uscita del grafo orientato $G = (V, E)$

4.3 Rappresentazione

Disegnare un grafo di dimensioni contenute sul foglio o sullo schermo del computer ha degli ovvi limiti di spazio. Ciò diventa complicato per grafi di dimensioni maggiori e, pertanto, esistono due modi principali per rappresentarlo nella memoria del computer, come discusso di seguito. Per una trattazione sugli algoritmi che utilizzano queste rappresentazioni, si rimanda al corso di Programmazione e Algoritmi.

Matrice di adiacenza

La rappresentazione più immediata per un grafo $G = (V, E)$ con $n = |V|$ nodi è quella tabellare.

Definizione 4.3.1. *La* matrice di adiacenza *per G è una tabella A con n righe e n colonne, numerate da 0 a $n-1$, dove la casella che si trova all'incrocio tra la riga i e la colonna j viene chiamata* cella A_{ij} *e assume un valore binario in $\{0, 1\}$:*

- *Se G non è orientato, per ogni $i, j \in V$, vale che $A_{ij} = 1$ se l'arco ij esiste in E; altrimenti vale che $A_{ij} = 0$. Poiché gli archi non sono orientati, ij e ji indicano il medesimo arco che connette i nodi i e j, e quindi le celle A_{ij} e A_{ji} contengono il medesimo valore binario per ogni i e j, e A si dice* simmetrica.
- *Se G è orientato, vale che $A_{ij} = 1$ se l'arco (i, j) esiste in E; altrimenti vale che $A_{ij} = 0$. Si noti che A non è necessariamente simmetrica in questo caso, perché non è detto che anche l'arco (j, i) esista.*

Esempio 4.3.2. *Per i due grafi in Figura 4.1, le corrispondenti matrici di adiacenza sono mostrate di seguito.*

	0	1	2	3	4	5
0		1				1
1	1		1	1	1	1
2		1		1		
3		1	1		1	1
4		1		1		1
5	1	1		1	1	

	0	1	2	3	4	5
0	1	1				1
1				1		1
2		1				
3			1		1	1
4						
5				1	1	

Riguardo all'occupazione di memoria, notiamo che la matrice di adiacenza ha lo svantaggio di richiedere $n \times n$ celle, anche se solo poche celle hanno valore 1, quindi indipendentemente dal numero $m = |E|$ di archi.

Osservazione 4.3.3. *La rappresentazione mediante matrici di adiacenza è utile in quei contesti dove si possono usare tecniche provenienti dall'algebra lineare e dagli algoritmi numerici, che saranno discusse nei corsi successivi. Nel nostro contesto, discutiamo un esempio famoso che riguarda le pagine web. Il grafo $G = (V, E)$ per il web è orientato, dove V rappresenta l'insieme di tutte le pagine web navigabili con i nostri "browser". Nel navigare passiamo da una pagina i a una pagina j facendo click in un punto opportuno della pagina i (in termini tecnici, seguiamo un "hyperlink"): questo stabilisce una relazione implicita tra le due pagine che viene modellata con l'arco $(i, j) \in E$ (oppure ij se la direzione non viene ritenuta importante).*

Questa struttura a grafo implicita tra le pagine web permette di ottenere un formidabile risultato nei motori di ricerca, unitamente a ulteriori tecniche: classificare le innumerevoli pagine web in base alla loro importanza perché altrimenti sarebbe impossibile per un essere umano scorrerle tutte. Una delle tecniche più popolari è quella di fornire un valore di significatività alle pagine, chiamato rango, e sostanzialmente procedere con il seguente meccanismo di votazione: l'arco (i, j) viene interpretato come un voto che i effettua nei confronti di j e il contributo di tale voto viene pesato in base al rango corrente di i. Attraverso un meccanismo iterativo di voto, il rango di ogni pagina j viene aggiornato attraverso i ranghi dei suoi vicini entranti $i \in N^-(j)$: una pagina è tanto più significativa quanto più lo sono le pagine che puntano a lei. Se certe condizioni del grafo sono soddisfatte, il meccanismo si stabilizza: a quel punto il rango maggiore denota una significatività maggiore. Il meccanismo si basa sul fatto che A è vista come matrice e l'iterazione è vista come moltiplicazione tra oppurtune matrici.

Liste di adiacenza

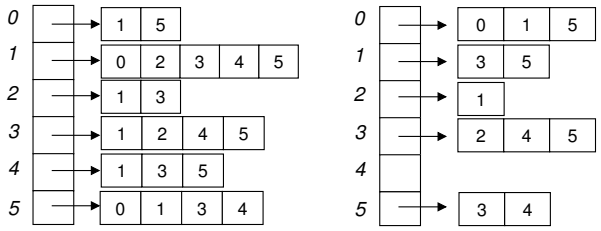
Per le reti sociali e le reti prese dal mondo reale (internet, reti stradali, reti neuronali, ecc.), spesso si utilizza una rappresentazione più compatta che permette di risparmiare spazio se il grafo è sparso,

cioè se il numero di archi è proporzionale al numero di nodi (il lettore si ponga la questione di determinare il massimo numero di archi per un grafo con n nodi). A tal fine si utilizzano le liste di adiacenza, che sono insiemi che rappresentano il vicinato per ogni nodo.

Definizione 4.3.4. *Le liste di adiacenza per G sono un array A di $n = |V|$ insiemi che rappresentano il vicinato per ogni nodo $x \in V$:*

- *Se G è non orientato, $A[x]$ rappresenta il vicinato $N(x)$.*
- *Se G è orientato, $A[x]$ rappresenta il vicinato in uscita $N^+(x)$.*

Esempio 4.3.5. *Con riferimento ai due grafi in Figura 4.1, mostriamo la loro rappresentazione mediante liste di adiacenza.*



Per il grafo orientato è utile osservare che avere tutti i vicinati in uscita è sufficiente a rappresentare E e che, da questi, è possibile ricavare tutti i vicinati in ingresso (chiaramente vale anche l'opposto, da quelli di ingresso è possibile ricavare quelli di uscita.) Si noti che il nodo 4 non ha vicini in uscita e quindi la sua lista è vuota.

La rappresentazione mediante liste di adiacenza richiede n celle di memoria, una per nodo nell'array, e poi tante celle quanti sono gli elementi nelle liste di adiacenza. Poiché la lista per il nodo i contiene $d_i = |N(i)|$ elementi, pari al suo grado, possiamo utilizzare la Proposizione 4.2.2 per ottenere che $\sum_{i \in V} d_i = 2|E| = 2m$ celle sono occupate in questo modo. In totale abbiamo quindi $n + 2m$ celle di memoria per i grafi, che rendono conveniente questa rappresentazione rispetto alle matrici di adiacenza quando $n + 2m < n^2$ (come spesso succede nei grafi reali). Analogamente, per la Proposizione 4.2.5, abbiamo $n + m$ celle per i grafi orientati.

Osservazione 4.3.6. *Un loro proficuo utilizzo è per esempio con i motori di ricerca quando devono raccogliere le pagine navigabili. Il meccanismo utilizzato per la raccolta delle pagine web raggiungibili si basa su specifici programmi chiamati "crawler" o "spider". Questi partono da un insieme predeterminato S di pagine web a cui aggiungono le pagine via via esplorate: presa una pagina $i \in S$, ne viene esaminato il contenuto per determinare il suo vicinato in uscita $N(i)$; per ogni pagina $j \in N(i)$, se la pagina j non appare in S, viene aggiunta a esso. Questo procedimento si ripete fino a che non è più possibile estendere S in questo modo. La scelta iniziale di S è determinante per raggiungere il maggior numero di pagine e non tutte le pagine sono comunque raggiunte in questo modo, si pensi per esempio a quelle del "dark web" oppure alle pagine generate dinamicamente.*

Grafi etichettati e pesati

La struttura di base di un grafo $G = (V, E)$ può essere arrichita con ulteriori informazioni, ponendo delle *etichette* sugli archi e sui nodi. Per esempio, se G rappresenta una rete sociale, vogliamo etichettare i nodi con l'identità dell'utente corrispondente e l'arco con la data in cui è stata accettata l'amicizia. In una rete stradale, vogliamo associare ai nodi le località e agli archi la distanza in km tra di esse.

Definizione 4.3.7. *Un grafo* etichettato *è una un tripla $G = (V, E, L)$ dove L (per etichettatura o "labelling") è una funzione $L : (V \cup E) \mapsto D$ che associa ad ogni nodo e arco una etichetta presa da un certo dominio D di valori (identità, date, località, distanze, ecc.). Nel caso che D sia un valore numerico (soliltamente un numero reale), il grafo etichettato si chiama* pesato *e ciascuna etichetta viene indicata come* peso *(dell'arco o del nodo).*

Un grafo etichettato (e quindi pesato) può essere orientato o meno. La sua rappresentazione è un'immediata estensione di quanto visto prima.

- Per i nodi, si costruisce un array di n etichette, dove la posizione i indica l'etichetta per il nodo i.
- Per gli archi, nella matrice di adiacenza il peso sostituisce il valore 1, usando un valore speciale `null` al posto del valore 0, per distinguere il caso in cui l'arco non esista.
- Per gli archi, nelle liste di adiancenza, invece di memorizzare i soli nodi in ciascuna lista, al posto di ogni nodo si memorizza una coppia del tipo (nodo, etichetta).

Un esempio di grafo pesato con la sua rappresentazione è illustrato in Figura 4.2.

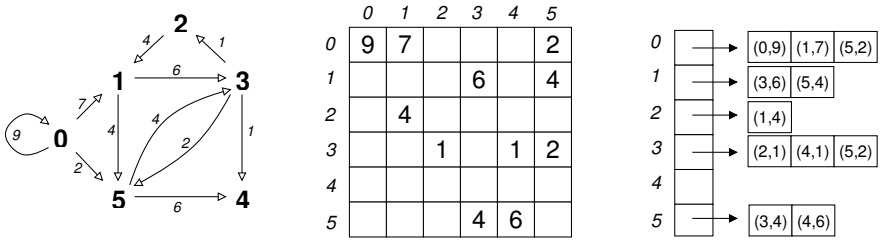


Figura 4.2: Un grafo pesato orientato, la matrice di adiacenza e le liste di adiacenza.

Isomorfismo

Nelle rappresentazioni discusse finora abbiamo assegnato una numerazione agli n nodi del grafo G. È possibile che G stesso venga rappresentato altrove con un'altra numerazione di tali n nodi. Per esempio, potrebbero esserci due applicazioni che costruiscono lo *stesso* grafo ma con numerazione *diversa* dei nodi: potremmo ricostruire una rete sociale a partire da certi individui oppure da altri, producendo due grafi che differiscono solo per come sono numerati i nodi. Come possiamo renderci conto che è lo stesso grafo?

Supponiamo infatti di avere i due grafi illustrati in Figura 4.3; ai fini della nostra discussione, chiamiamo $G_1 = (V_1, E_1)$ quello a sinistra e $G_2 = (V_2, E_2)$ quello a destra. Se guardiamo le loro matrici d'adiacenza e le loro liste d'adiacenza, queste risultano essere differenti. Possiamo però osservare che G_1 e G_2 hanno lo stesso numero di nodi e archi ($|V_1| = |V_2|$ e $|E_1| = |E_2|$); anche i gradi dei nodi sono uguali.

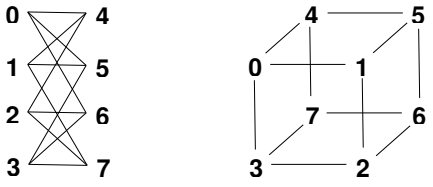


Figura 4.3: Due grafi isomorfi.

Va però detto che G_1 e G_2 sono apparentemente distinti. Per accertarci che rappresentino la stessa relazione sui nodi, non basta la nozione di uguaglianza, ma ne occorre una più robusta.

Definizione 4.3.8. *Dati due qualunque grafi $G_1 = (V_1, E_1)$ e $G_2 = (V_2, E_2)$, dove $|V_1| = |V_2|$ e $|E_1| = |E_2|$, un* isomorfismo *tra G_1 e G_2 è una biiezione $f : V_1 \mapsto V_2$ tra i loro nodi tale che, per ogni coppia di nodi $u, v \in V_1$, vale che $uv \in E_1$ se e solo se $f(u)f(v) \in E_2$ (cioè esiste il corrispondente arco in entrambi i grafi, oppure non esiste in entrambi di essi). In tal caso, G_1 e G_2 sono detti* isomorfi.

Nella Figura 4.3, i nodi di G_1 vanno rinumerati secondo la seguente biiezione f con i nodi di G_2; osserviamo che $24 \in E_1$ e $f(2)f(4) = 65 \in E_2$, $25 \notin E_1$ e $f(2)f(5) = 60 \notin E_2$, e così via, per tutte le coppie di nodi in V_1 e le loro corrispondenti coppie in V_2.

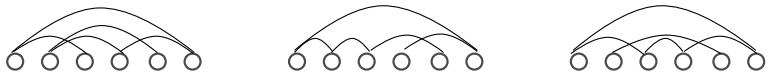
$$f \ : \quad 0 \mapsto 4, \quad 1 \mapsto 1, \quad 2 \mapsto 6, \quad 3 \mapsto 3, \quad 4 \mapsto 5, \quad 5 \mapsto 0, \quad 6 \mapsto 7, \quad 7 \mapsto 2.$$

In effetti, esiste un'enorme quantità di modi per numerare i nodi con valori distinti in $V = \{0, 1, \ldots, n-1\}$ e, quindi, altrettanti modi per rappresentare lo stesso grafo (esistono cioè più biiezioni tra V_1 e V_2 che sono isomorfismi). Per esempio, anche questa biiezione $\tilde{f}$ fornisce un isomorfismo per G_1 e G_2.

$$\tilde{f} \ : \quad 0 \mapsto 5, \quad 1 \mapsto 7, \quad 2 \mapsto 2, \quad 3 \mapsto 0, \quad 4 \mapsto 6, \quad 5 \mapsto 4, \quad 6 \mapsto 1, \quad 7 \mapsto 3.$$

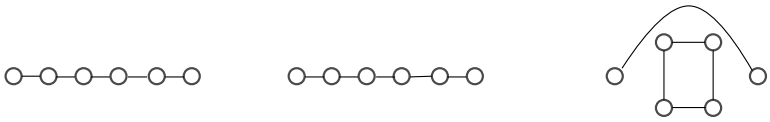
Una volta fissate le numerazioni dei nodi in V_1 e V_2, non tutte le biiezioni da V_1 a V_2 sono valide a tal fine: per esempio, la biiezione $\hat{f}$ tale che $\hat{f}(0) = 3$ e $\hat{f}(4) = 5$ non è un isomorfismo poiché $04 \in E_1$ ma $\hat{f}(0)\hat{f}(4) = 35 \notin E_2$ in Figura 4.3.

L'apparante distinzione nella Figura 4.3 nasce dall'artificio di enumerare differentemente gli *stessi nodi* ma è chiaro che la relazione tra i nodi è la medesima se ignoriamo la loro numerazione. Purtroppo non basta verificare che i gradi dei nodi corrispondano. Nell'esempio seguente, dove la numerazione dei nodi non è riportata per brevità, osserviamo che i tre grafi hanno lo stesso numero di nodi e archi, e i gradi dei nodi possono essere messi in corrispondenza. Tuttavia, solo i primi due grafi (a) e (b) sono isomorfi, mentre il terzo non lo è con i primi due.



(a) (b) (b)

Quest'osservazione si evidenzia meglio se disponiamo diversamente i nodi dei tre grafi e disegniamo i corrispondenti archi.



(a) (b) (b)

Notiamo che l'isomorfismo è una relazione di equivalenza e che il problema di decidere se due grafi *arbitrari* G_1 e G_2 di n nodi sono isomorfi equivale a trovare una loro biiezione che preserva gli archi, se esiste, ed è uno dei problemi algoritmici fondamentali con molte implicazioni (per esempio, stabilire se due grafi arbitrari *non* sono isomorfi ha delle importanti implicazioni nella crittografia). Chiaramente il problema è interessante quando $|V_1| = |V_2|$ e $|E_1| = |E_2|$, altrimenti è facile rispondere (NO).

4.4 Cammini e connettività

I grafi si prestano a modellare molti problemi computazionali e, come vedremo in seguito, Eulero li introdusse nel XVIII secolo per risolvere un "puzzle" della sua epoca che riguardava i percorsi nella città di Königsberg. Per adesso immaginiamo la mappa di una città come un grafo non orientato G, dove i nodi corrispondono a incroci e gli archi corrispondono a segmenti di strada.

Se partiamo da un nodo e ci spostiamo fino ad un altro nodo, percorriamo quello che viene chiamato un *cammino* nel grafo $G = (V, E)$.

Definizione 4.4.1. *Un* cammino P *in* G *inizia con un nodo* v_0 *e termina in un nodo* v_k*, chiamati gli* estremi *di* P*. In particolare,* P *corrisponde a un'alternanza di nodi e archi, nell'ordine in cui vengono incontrati, e viene rappresentato dalla sequenza di nodi adiacenti* $v_0, v_1, v_2, \ldots, v_{k-1}, v_k$*, tali che* $v_{i-1}v_i \in E$ *per* $i = 1, 2, \ldots, k$*. Il cammino* P *si dice di* lunghezza k *perché attraversa* k *archi.*

Esempio 4.4.2. *Nel grafo nella sinistra di Figura 4.1, la sequenza* $0, 1, 3, 4$ *corrisponde a un cammino semplice composto dagli archi* $01, 13, 34$*, dove* $v_0 = 0$ *e* $v_k = 4$ *(qui* $k = 3$ *perché ci sono 3 archi attraversati). Si noti che se* $v_0 = v_k$*, il cammino ha lunghezza* 0*: infatti, ogni cammino da un nodo a se stesso attraversa zero archi.*

Definizione 4.4.3. *Un cammino* P *(in inglese, walk) è detto* cammino semplice *(in inglese, path) se* P *passa per ogni suo nodo e arco una* sola *volta. È detto invece* percorso *(in inglese, trail) se* P *passa per lo stesso arco* al più *una volta. Un cammino* P *è detto* chiuso *se i suoi estremi sono uguali (cioè* $v_0 = v_k$*), e un cammino semplice chiuso è detto un* ciclo*, mentre un percorso chiuso è un* circuito*. (Partendo da* v_0 *si ritorna in* $v_k = v_0$ *ma ovviamente questo non viene contato due volte.)*

Esempio 4.4.4. *Nel grafo nella sinistra di Figura 4.1, la sequenza* $5, 1, 2, 3, 5, 0$ *corrisponde a un percorso perché, pur non passando per gli stessi archi, passa più di una volta per il nodo* 5*. La sequenza* $3, 5, 1, 4, 5, 3, 2$ *corrisponde a un cammino che non è semplice né un percorso, dato che l'arco non orientato* 35 *viene attraversato più di una volta.*

Nei grafi orientati si applica la stessa terminologia delle definizioni suddette, tuttavia è necessario che ogni arco $(v_{i-1}, v_i) \in E$ sia percorso nella direzione corretta (seguendo l'orientamento dalla coda verso la testa).

Esempio 4.4.5. *Nel grafo orientato nella destra di Figura 4.1, la sequenza* $0, 1, 3, 4$ *corrisponde a un cammino semplice. La sequenza* $5, 1, 3$ *non è cammino perché l'arco* $(5, 1)$ *non esiste (da* 1 *si può andare a* 5 *ma non vice versa). La sequenza* $1, 3, 2, 1$ *è un ciclo mentre* $1, 3, 5, 3, 2, 1$ *è un circuito perché passa per lo stesso nodo* 5 *due volte (chiaramente* $v_0 = v_k = 1$ *non conta) e ogni arco è attraversato al più una volta.*

Una nozione intimamente collegata a quella di cammino è la connettività tra nodi e iniziamo a considerare i grafi non orientati.

Definizione 4.4.6. *Un grafo* $G = (V, E)$ *si dice* connesso *se esiste almeno un cammino tra ogni possibile coppia di nodi distinti. Se* G *è connesso, c'è una sola* componente connessa*: quest'ultima è definita come un insieme di nodi a due a due connessi, che non può essere ulteriormente esteso ad altri nodi (è* massimale*). Se* G *non è connesso, possiamo raggruppare i nodi in componenti connesse. I nodi in ciascuna componente sono collegati da almeno un cammino; invece, prese due qualunque componenti connesse, i nodi di una componente non hanno un cammino che li conduce ai nodi dell'altra componente.*

Esempio 4.4.7. *Il grafo nella sinistra della Figura 4.1 è connesso, mentre quello in Figura 4.4 non lo è: infatti non esiste un cammino, per esempio, dal nodo* 2 *al nodo* 8*, contraddicendo il fatto che debba esistere per ogni coppia (quindi inclusa* 2 *e* 8*).*

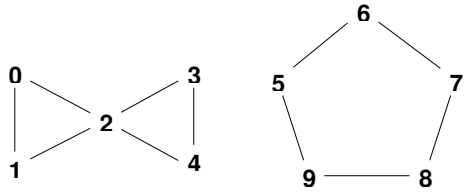


Figura 4.4: Un grafo non connesso, contenente due componenti connesse $\{0, 1, 2, 3, 4\}$ e $\{5, 6, 7, 8, 9\}$

Osserviamo nuovamente che possiamo vedere E come una relazione su V per i nodi di un grafo $G = (V, E)$. Se prendiamo $E^\star$, vale che $(x, y) \in E^\star$ se e soltanto se esiste almeno un cammino da x a y.

Proposizione 4.4.8. *Per un grafo non orientato* G*, la relazione* $E^\star$ *è di equivalenza.*

Dimostrazione. Lasciamo al lettore la dimostrazione che $E^\star$ rispetta le proprietà riflessiva, simmetrica e transitiva. ■

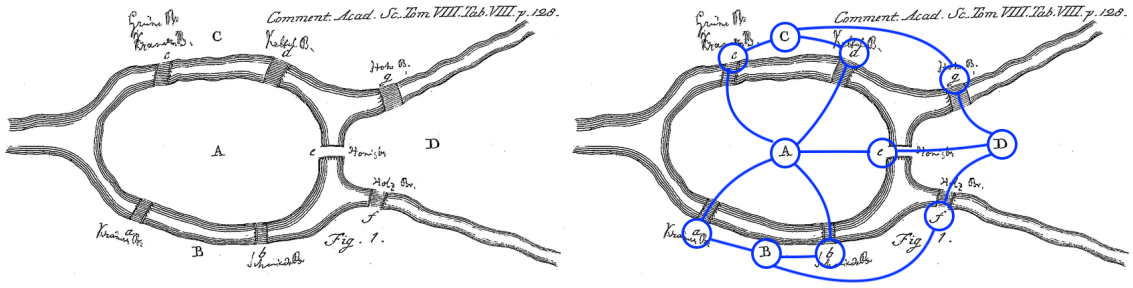


Figura 4.5: Il problema dei ponti di Königsberg nella figura originale di Eulero e la sua rappresentazione mediante un grafo.

Come abbiamo visto, una relazione di equivalenza genera classi di equivalenza. Nel caso della relazione $E^\star$ suddetta, le classi di equivalenza sono appunto le componenti connesse del grafo G, e corrispondono agli insiemi massimali di nodi connessi tra loro.

Cammini euleriani

Abbiamo ora tutti gli elementi utili a mostrare il problema affrontato da Eulero per i grafi non orientati.

Definizione 4.4.9. *Dato un grafo connesso $G = (V, E)$, un circuito* euleriano *attraversa* tutti gli archi *in E una e una sola volta (e può passare per gli stessi nodi più volte). In maniera analoga si definisce il percorso euleriano, solo che il nodo di partenza è diverso da quello di destinazione.*

Esempio 4.4.10. *L'esempio famoso è quello di disegnare una casetta su un foglio di carta senza mai sollevare la penna. Nella Figura 4.1 a sinistra, un percorso euleriano dal nodo 4 al nodo 1 è dato dalla sequenza di nodi $4, 1, 2, 3, 5, 0, 1, 3, 4, 5, 1$ che attraversa tutti gli archi esattamente una volta. Si noti che un grafo può avere più circuiti euleriani.*

Eulero si pose la questione di trovare un tale circuito perché nella città di Königsberg c'era un parco naturale ricco di torrenti che formavano un'isola A e dividevano il resto del parco in tre zone B, C, D, come illustrato nel disegno originario di Eulero e riportato nella sinistra della Figura 4.5, collegandole con sette ponti a, b, c, d, e, f. Le persone si divertivano a cercare di attraversare tutti i ponti una e una sola volta, tornando al loro punto di partenza.

Eulero formulò il problema come un grafo connesso $G = (V, E)$ in cui trovare quello che oggi chiamiamo un circuito euleriano. Considerando il disegno annotato nella destra della Figura 4.5, le zone e i ponti diventano gli 11 nodi del grafo. Per ogni zona $Z \in \{A, B, C, D\}$, creiamo un arco Zp per ogni ponte $p \in \{a, b, c, d, e, f\}$ che congiunge Z alle altre zone. Per esempio, abbiamo gli archi Cc, Cd, Cg mentre non possiamo avere Ca o CA per ovvi motivi geografici e non possiamo avere ca perché occorre passare per una zona tra i due ponti. Gli archi sono non orientati perché ciascun collegamento può essere percorso in entrambe le sue direzioni. È quindi possibile attraversare tutti i ponti come richiesto se e solo se G ammette un circuito euleriano.

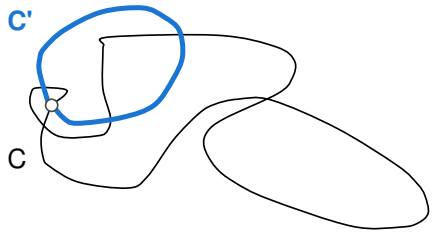
Eulero ebbe l'intuizione che la condizione necessaria e sufficiente affinché ciò avvenga è che G sia connesso e i suoi nodi abbiano tutti grado pari: pertanto il grafo G nella Figura 4.5 non contiene un circuito euleriano, in quanto presenta quattro nodi di grado dispari. Eulero dimostrò inoltre che se esattamente due nodi hanno grado dispari allora il grafo contiene un percorso euleriano: purtroppo il grafo G nella Figura 4.5 non contiene neanche un percorso euleriano.

Teorema 4.4.11. *Dato un grafo connesso G, esiste un circuito euleriano se e solo se ogni nodo ha grado pari. Esiste un percorso euleriano dal nodo x al nodo y se e solo se d_x e d_y sono gli unici gradi dispari.*

Dimostrazione. Cominciamo col dimostrare che il grafo $G = (V, E)$ contiene un circuito euleriano se e solo se ogni nodo ha grado pari.

(A) Se esiste un circuito euleriano, vuol dire che esiste un circuito che utilizza tutti gli archi di G esattamente una volta. Ogni volta che il circuito traversa un nodo utilizza due archi mai attraversati prima, ovvero il numero di archi incidenti in un nodo (cioè il suo grado) è esattamente il doppio del numero di volte che il circuite passa per il nodo, ovvero grado pari.

(B) Se ogni nodo ha grado pari, dimostriamo che esiste un circuito euleriano in G. Attraversiamo G a partire da un nodo r qualsiasi, marcando di volta in volta l'arco utilizzato per spostarsi nel nodo successivo: possiamo utilizzare solo archi non marcati (cioè mai attraversati prima) e quando attraversiamo un arco lo marchiamo per evitare di attraversarlo nuovamente. Osserviamo che in questo modo attraversiamo un percorso (non ci sono archi attraversati più volte) che prima o poi ritorna in r, creando un circuito. Infatti, se così non fosse, terminerebbe in un nodo $u \neq r$ che non ha altri archi incidenti: ma u non può essere un nodo diverso da r in quanto, se fosse stato attraversato già $k \geq 0$ volte, il suo numero di archi incidenti sarebbe $2k + 1$, quindi con grado dispari (è impossibile). Possiamo quindi dedurre che abbiamo trovato un circuito C che passa per r: se tutti gli archi sono stati attraversati da C allora C è euleriano e possiamo concludere la nostra argomentazione. Altrimenti utilizziamo l'ipotesi che G è connesso: in tal caso, deve esserci per forza un nodo r' in C che ha un arco incidente non marcato (altrimenti G non sarebbe connesso). L'idea è ripetere per r' quanto fatto per r, trovando così un altro circuito C' che passa per r', utilizzando la stessa argomentazione esposta poco fa. Notiamo che C e C' sono disgiunti sugli archi, cioè non possono attraversare uno stesso arco, perché marchiamo gli archi man mano che li attraversiamo. Inoltre C e C' si intersecano sicuramente in r' per costruzione (magari anche in ulteriori nodi, ma questo non è rilevante). Utilizziamo allora la seguente osservazione: se due circuiti disgiunti si intersecano in almeno un nodo, allora formano un unico circuito, come illustrato in figura:



Indichiamo con $C + C'$ il circuito risultante: se attraversa tutti gli archi, abbiamo finito; altrimenti troviamo un altro circuito C'' come sopra e lo componiamo con $C + C'$ ottenendo il circuito $C + C' + C''$. Procedendo in questo modo, continuiamo a comporre circuiti per ottere un unico circuito, e prima o poi attraversiamo tutti gli archi. Quindi esiste un circuito euleriano in G.

Infine, se ci sono due nodi x e y di grado dispari, possiamo aggiungere un nuovo nodo $z \notin V$ e gli archi xz e yz al grafo G, ottendo il grafo $G' = (V \cup \{z\}, E \cup \{xz, yz\})$. In tal modo i gradi dei nodi in G' sono tutti pari ed esisterà un circuito euleriano C in G' che dovrà attraversare anche xz e yz consecutivamente: rimuovendo xz e yz da C in G', otterremo il percorso euleriano in G. ■

Cammini hamiltoniani e il problema del commesso viaggiatore

Consideriamo adesso una nozione apparentemente simile e di grande importanza per l'informatica, il cui nome deriva dal matematico William R. Hamilton che la introdusse nel XIX secolo.

Definizione 4.4.12. *Dato un grafo connesso $G = (V, E)$, un ciclo hamiltoniano è un ciclo semplice che attraversa tutti i nodi in V una e una sola volta (e non è detto che attraversi tutti gli archi).*

Esempio 4.4.13. *Nella Figura 4.1 a sinistra, un ciclo hamiltoniano è dato dalla sequenza di nodi $0, 1, 2, 3, 4, 5, 0$ che attraversa tutti i nodi esattamente una volta (chiaramente il nodo 0 non conta due volte perché è sia di partenza che di arrivo). Si noti che un grafo può avere più cicli hamiltoniani. Il cammino hamiltoniano è definito analogamente, solo che il nodo x di partenza e quello y di arrivo sono diversi: in sostanza si tratta di trovare una permutazione dei nodi in V che dia luogo a un cammino semplice.*

L'apparente analogia con il circuito euleriano termina purtroppo qui: attraversare tutti gli *archi* una sola volta nel circuito euleriano, di contro ad attraversare tutti i *nodi* una sola volta nel ciclo hamiltoniano. Non esiste al momento una caratterizzazione per garantire l'esistenza o meno di un ciclo hamiltiniano in G, analogamente a quanto enunciato nel Teorema 4.4.11. Il ciclo hamiltoniano ha struttura combinatoria più complessa e sfuggente: trovare un ciclo hamiltoniano può risultare più arduo come compito e fa parte di una famiglia importante di problemi, chiamati NP-completi, la cui trattazione sarà data in un corso successivo.

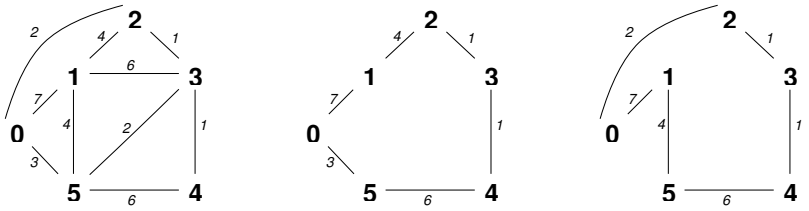
Nel caso di grafi pesati, vogliamo trovare un ciclo hamiltoniano i cui archi abbiano somma minima dei pesi, dando luogo a un famoso problema.

Definizione 4.4.14. *Dato un grafo pesato $G = (V, E, L)$, il peso di un ciclo hamiltoniano $H = v_0, v_1, \ldots, v_k$ è la somma dei pesi $L(v_{i-1}, v_i)$ degli archi $v_{i-1}, v_i \in E$ attraversati da H:*

$$\text{peso}(H) = \sum_{i=1}^{k} L(v_{i-1}, v_i) + L(v_k, v_0).$$

Nel problema del commesso viaggiatore*, si vuole trovare un ciclo hamiltoniano di peso minimo: i nodi rappresentanto le città e gli archi i collegamenti diretti tra due città aventi il peso come costo; il commesso viaggiatore vuole attraversare tutte le città una sola volta minimizzando il costo totale.*

Esempio 4.4.15. *Consideriamo il grafo pesato G mostrato a sinistra nella figura, dove i pesi sono interi positivi:*



Prendendo il ciclo hamiltoniano $H = 0, 1, 2, 3, 4, 5$, mostrato al centro, otteniamo $\text{peso}(H) = 7 + 4 + 1 + 1 + 6 + 3 = 22$ *ma non è minimo. Al commesso viaggiatore conviene infatti attraversare quello a destra $H^* = 0, 2, 3, 4, 5, 1$ con* $\text{peso}(H^*) = 2 + 1 + 1 + 6 + 4 + 7 = 21$.

Grafi orientati

Finora ci siamo principalmente concentrati sui grafi non orientati. Passando a considerare i grafi orientati, le definizioni di circuito euleriano e cammino hamiltoniano si traducono immediatamente su questi e derivano dalle definizioni di circuiti e cammini orientati; inoltre, si preserva la richiesta di attraversare tutti gli archi o tutti i nodi. Ciò che cambia significativamente è la nozione di connettività.

Definizione 4.4.16. *Un grafo orientato $G = (V, E)$ si dice* fortemente *connesso se esiste almeno un cammino (ovviamente orientato) tra ogni possibile coppia di nodi distinti. Se G è fortemente connesso, c'è una sola* componente fortemente connessa*: quest'ultima è definita come un insieme di nodi a due a due connessi in entrambe le direzioni, che non può essere ulteriormente esteso ad altri nodi (è* massimale*). Se G non è fortemente connesso, possiamo raggruppare i nodi in componenti connesse. I nodi in ciascuna componente sono collegati da almeno un cammino in entrambe le direzioni; invece, prese due qualunque componenti connesse, c'è almeno un nodo in una componente che non ha un cammino che lo conduce a un nodo dell'altra componente (potrebbe esserci nell'altra direzione ma non basta).*

Esempio 4.4.17. *Per esempio, il grafo nella destra della Figura 4.1 non è fortemente connesso perché non esiste un cammino, per esempio, dal nodo 4 al nodo 0 (mentre esiste un cammino dal nodo 0 al nodo 4 ma non basta). Ci sono tre componenti fortemente connesse: due sono formate da singoli nodi, cioè $\{0\}$ e $\{4\}$, l'altra è formata dai restanti nodi, cioè $\{1, 2, 3, 5\}$.*

Anche con i grafi orientati possiamo vedere E come una relazione su V. Se prendiamo $E^\star$, osserviamo che $(x, y) \in E^\star$ se e soltanto se esiste almeno un cammino (orientato) da x a y. Contrariamente al caso dei grafi non orientati, $E^\star$ non è simmetrica: il fatto che esista un cammino da x a y non implica necessariamente che ne esista uno da y a x. Tuttavia, $E^\star$ rispetta le proprietà riflessiva e transitiva.

Proposizione 4.4.18. *Per un grafo orientato G, la relazione $E^\star \cap (E^\star)^{op}$ genera le componenti fortemente connesse di G.*

Dimostrazione. Abbiamo che $(x, y) \in E^\star \cap (E^\star)^{op}$ se e solo se esiste un cammino da x a y e viceversa. Quindi x e y fanno parte della stessa componente connessa. D'altronde, se x e y fanno parte della stessa componente connessa, $(x, y) \in E^\star$ e $(x, y) \in (E^\star)^{op}$; quindi $(x, y) \in E^\star \cap (E^\star)^{op}$. Si noti che $E^\star \cap (E^\star)^{op}$ è una relazione di equivalenza. ■

Come si può osservare, prese due qualunque componenti fortemente connesse, queste sono completamente isolate (non ci sono archi dall'una all'altra), oppure esiste almeno un nodo di una componente fortemente connessa che non è collegato ad alcun nodo dell'altra componente fortemente connessa dell'altra componente, altrimenti le due componenti potrebbero fondersi in un'unica componente: per esempio, dal nodo 2 non si può andare al nodo 0; dal nodo 4 non si può andare né al nodo 0 né al nodo 2.

All'interno di ogni componente fortemente connessa, si nota un'interessante proprietà: presi due qualunque nodi in essa, esiste sempre un cammino chiuso (orientato) che li attraversa entrambi. Per esempio, per i nodi 2 e 5 il cammino chiuso è formato dalla sequenza $1, 5, 3, 2, 1$; per i nodi 1 e 3 il cammino chiuso corrisponde a $2, 1, 3, 2$; per i nodi 3 e 5, il cammino chiuso è dato da $3, 5, 3$.

Proposizione 4.4.19. *Un grafo $G = (V, E)$ è fortemente connesso se e solo se per ogni coppia di nodi $x, y \in V$ esiste sempre un cammino chiuso dato da $v_0, v_1, \ldots, v_k$ (dove $v_0, \ldots, v_k \in V$ e $v_0 = v_k$) tale che $x = v_i$ e $y = v_j$ (dove $0 \leq i, j \leq k$).*

Dimostrazione. (A) Sia G fortemente connesso. Presi due nodi arbitrari x e y, esiste un cammino $x_0, x_1, \ldots, x_r$ da x a y, e un cammino $y_0, y_1, \ldots, y_s$ da y a x, per definizione di fortemente connesso, dove $x_0 = x = y_s$ e $x_r = y = y_0$ indicano gli stessi due nodi. Se prendiamo la concatenatazione $x_0, x_1, \ldots, x_r, y_1, \ldots, y_s$ e la ridenominiamo come $v_0, v_1, \ldots, v_k$, osserviamo che è un cammino chiuso ($v_0 = x_0 = y_s = v_k$) e, inoltre, $x = x_0 = v_0$ (quindi $i = 0$) e $y = x_r = v_r$ (quindi $j = r$).

(B) Sia soddisfatta la proprietà che per ogni coppia di nodi $x, y \in V$ esiste sempre un cammino chiuso. È quindi possibile attraversare il cammino chiuso per andare da x a y, e da y a x. Quindi esiste sempre un cammino tra ogni coppia di nodi e G è fortemente connesso. ■

4.5 Grafi aciclici: alberi e DAG

Consideriamo una classe importante di grafi, trattando separatemente il caso in cui i grafi non sono orientati da quelli orientati.

Definizione 4.5.1. *Un grafo G (orientato o meno) si dice* ciclico *se esiste almeno un ciclo in G. Altrimenti, si dice* aciclico.

Grafi non orientati: alberi

Definizione 4.5.2. *Un grafo connesso aciclico è detto* albero. *I nodi alle estremità, ovvero i nodi di grado* 1, *vengono detti* foglie *e gli altri nodi sono talora indicati come* interni. *Se il grafo aciclico non è connesso, ciascuna sua componente connessa è un albero, e l'unione di tali alberi viene chiamata* foresta.

Esempio 4.5.3. *Consideriamo l'albero in Figura 4.6. Le foglie sono i nodi $1, 3, 4, 5, 7, 9$ e i nodi interni sono $0, 2, 6, 8$. Osserviamo che l'albero ha almeno una foglia, essendo finito. Inoltre ha $n = 10$ nodi e $m = 9$ archi: come vedremo, non è un caso che sia $m = n - 1$. Vedremo anche che, presi due nodi x e y, c'è un solo cammino semplice che li collega: per esempio, $x = 0$ e $y = 8$ hanno $0, 2, 6, 8$ in quanto gli altri cammini passano due volte per lo stesso nodo. Proviamo adesso a togliere un qualunque arco, per esempio* 26*: il risultato è che l'albero non è più connesso (e questo*

vale per ogni arco). Se invece proviamo ad aggiungere un nuovo arco tra due nodi, il grafo diventa ciclico: per esempio, collegando 0 *e* 8*, chiudiamo il cammino semplice* $0, 2, 6, 8$ *che li collega.*

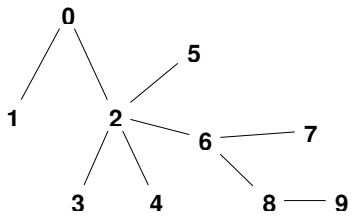


Figura 4.6: Un albero, ovvero un grafo senza cicli.

Osservazione 4.5.4. *Gli alberi sono strutture fondamentali in informatica: consentono di rappresentare relazioni gerarchiche e sono alla base di alcune delle più importanti strutture di dati. Il nome deriva dall'idea che la struttura di questo tipo di grafo possa essere immaginata simile alle diramazioni di un albero, come mostrato in Figura 4.6: tali diramazioni si originano in un nodo e non vanno mai a chiudersi su un altro nodo, perché altrimenti formerebbero un ciclo. Un esempio di albero nell'accezione informatica che tutti conosciamo è l'albero genealogico: se immaginiamo un capostipite della famiglia, come un bis-nonno, e di tracciare linee verso i figli, poi da loro ai nipoti, e così via. Otteniamo (tipicamente) un grafo senza cicli, dove le foglie sono le ultime generazioni che ancora non hanno figli. Inoltre, come vedremo in seguito, gli alberi sono implicitamente alla base di alcune fondamentali tecniche di ragionamento.*

Dimostriamo ora le proprietà osservate empiricamente nell'esempio di prima.

Proposizione 4.5.5. *Dato un grafo aciclico (albero)* $G = (V, E)$ *con* $n = |V| > 0$ *nodi, valgono le seguenti proprietà:*

(a) *Se* $n \geq 2$*, allora* G *ha almeno una foglia (ovvero un nodo di grado* 1*).*

(b) G *ha esattamente* $m = n - 1$ *archi in* E*.*

(c) *Per ogni coppia di nodi distinti* $x, y \in V$*, esiste un unico cammino semplice che collega* x *e* y*.*

(d) *Per ogni arco* $xy \in E$*, la rimozione di* xy *rende il grafo non connesso, cioè il grafo* $G' = (V, E \setminus \{xy\})$ *ha più di una componente connessa.*

(e) *Per ogni coppia di nodi distinti* $x, y \in V$ *tale che* $xy \notin E$*, l'aggiunta dell'arco* xy *crea un ciclo, cioè il grafo* $G' = (V, E \cup \{xy\})$ *è ciclico.*

Dimostrazione. In alcuni passaggi utilizziamo il fatto che, se esiste un cammino (chiuso), allora esiste un cammino semplice (ciclo), come indicato nell'Esercizio 4.8.6.

(a) Scegliamo un nodo qualsiasi x e iniziamo un cammino a partire da x. Osserviamo che non potremo mai attraversare un nodo precedentemente attraversato nel cammino, altrimenti avremmo trovato un cammino chiuso, e quindi un ciclo, che non può esistere in G perché aciclico. Tuttavia, finché ci spostiamo verso un nodo di grado almeno 2, possiamo sempre fare un altro passo verso un ulteriore nodo adiacente, visto che esiste un altro vicino oltre a quello da cui siamo arrivati. Dato che G è finito, ne consegue che dovremo prima o poi incontrare un nodo di grado 1 (altrimenti non termineremmo mai di incontrare nuovi nodi).

(b) Se $n = 1$, abbiamo $m = 0 = n - 1$ archi. Se $n > 1$, esiste sempre una foglia per il punto (a): possiamo rimuovere tale foglia e il suo unico arco incidente, ottenendo un nuovo albero con $n - 1$ nodi e $m - 1$ archi. Possiamo applicare varie volte questo passo fino ad arrivare a $n = 1$, dove troveremo un nodo isolato (e 0 archi). Visto che abbiamo ripetuto l'operazione $n - 1$ volte, ogni volta rimuovendo esattamente un arco e un nodo, e abbiamo rimosso tutti gli archi dell'albero originale e ci avanza un nodo, ne consegue che $m = n - 1$.

(c) Essendo G connesso, esiste un cammino da x a y e quindi un cammino semplice P. Per dimostrare che P è unico, procediamo *per assurdo*: supponiamo che la proprietà non sia vera e perveniamo a una contraddizione, derivando così che la proprietà debba essere vera. Specificatamente, supponiamo che P non sia unico, ma esista un altro cammino semplice $P' \neq P$. Siccome entrambi partono da x e arrivano in y, sia x' l'ultimo nodo in comune tra P e P' prima che divergano partendo da x. In maniera analoga, andando a ritroso in P e P', partendo da y, sia y' l'ultimo nodo in comune tra P e P' prima che divergano. Notiamo che x' e y' esistono sempre (male che vada $x' = x$ e $y' = y$). Ora possiamo andare da x' a y' usando parte di P e tornare indietro da y' a x' usando parte di P' a ritroso. Abbiamo cioè un cammino chiuso e quindi un ciclo, contraddicendo l'ipotesi che G è aciclico.

(d) Anche qui procediamo per assurdo, ipotizzando che la rimozione di un arco $xy \in E$ da G lasci $G' = (V, E \setminus \{xy\})$ connesso: vuol dire che esiste un cammino da x a y in G', e quindi un cammino semplice P da x a y. Poichè ogni arco di G' appare anche in G, deduciamo che P esiste anche in G. Ma poiché esiste anche xy in G, abbiamo un ciclo formato da P e xy, che contraddice l'ipotesi che G è aciclico.

(e) Questa proprietà deriva immediatamente dalle precedenti. Sappiamo che esiste un cammino semplice P da x a y: se aggiungiamo l'arco xy, otteniamo $G' = (V, E \cup \{xy\})$. Poiché tutti gli archi di G appaiono anche in G', osserviamo che P esiste anche in G': l'aggiunta di xy chiude P e crea un ciclo.

■

Quando rappresentiamo una gerarchia tramite un albero, solitamente identifichiamo un nodo di ingresso tra quelli dell'albero.

Definizione 4.5.6. *Un albero è* radicato *quando uno dei suoi nodi viene indicato come* radice r. *Dato un nodo $y \neq r$, i nodi lungo l'unico cammino che collega y a r vengono chiamati gli* antenati *di y, e il primo (quello adiacente a y) è detto il* genitore *di y. Conversamente, y viene detto* discendente *dei suoi antenati, e* figlio *del suo nodo genitore. L'albero radicato serve anche a rappresentare partizioni annidate dei suoi nodi. Ciascuna partizione viene chiamata* sottoalbero *quando è formata dai discendenti del nodo.*

Esempio 4.5.7. *Consideriamo l'albero in Figura 4.6, e scegliamo il nodo $r = 0$ come radice. Possiamo osservare come la foglia 9 sia un discendente di 2, ma non di 5, in quanto solamente 2 si trova sul cammino da 9 verso la radice 0. L'albero nella sua interezza rappresenta l'insieme $V = \{0, 1, \ldots, 9\}$. La radice r partiziona tale insieme in $\{0\}$ e i sottoalberi $\{1\}$ e $\{2, 3, 4, 5, 6, 7, 8, 9\}$; a sua volta, il nodo 2 partiziona $\{2, 3, 4, 5, 6, 7, 8, 9\}$ in $\{2\}$ e i sottoalberi $\{3\}$, $\{4\}$, $\{5\}$, $\{6, 7, 8, 9\}$; il nodo 6 partiziona $\{6, 7, 8, 9\}$ in $\{6\}$ e i sottoalberi $\{7\}$ e $\{8, 9\}$; infine, il nodo 8 partiziona $\{8, 9\}$ in $\{8\}$ e il sottoalbero $\{9\}$.*

Osservazione 4.5.8. *È interessante notare come l'inclusione tra insiemi sia collegata all'ordine parziale rappresentato da un albero. Associamo a ogni nodo il suo sottoalbero, visto come insieme di nodi. Per esempio, al nodo 2 associamo l'insieme $\{2, 3, 4, 5, 6, 7, 8, 9\}$ in Figura 4.6. Vale la relazione che u è antenato di v se e solo se l'insieme associato a u contiene quello associato a v. Il nodo 2 è antenato del nodo 8 perchè $\{2, 3, 4, 5, 6, 7, 8, 9\}$ contiene $\{8, 9\}$; vice versa, i nodi 8 e 4 non sono antenati l'uno dell'altro perché i loro insiemi sono disgiunti. In genere, notiamo che gli insiemi associati a due nodi arbitrari o sono uno contenuto nell'altro oppure sono disgiunti.*

Spesso viene richiesto che i figli di un nodo siano ordinati in un albero radicato.

Definizione 4.5.9. *Un albero radicato si dice* ordinale *se per ciascun nodo interno è possibile discriminare chi è il primo figlio, chi è il secondo figlio, e così via. Un albero radicato si dice* cardinale *o k-*ario *se ogni nodo interno ha esattamente k figli, alcuni dei quali possono essere vuoti (indicati con `null`). I figli sono numerati e chiamati figlio 0, figlio 1, ..., figlio $k-1$. L'albero è* completo *se ogni nodo interno ha tutti e k i figli non vuoti. Un caso speciale e molto importante è quando $k = 2$: in tal caso l'albero viene detto* binario*, dove il figlio 0 viene chiamato* figlio sinistro *e il figlio 1 viene chiamato* figlio destro*.*

Esempio 4.5.10. *Nella Figura 4.6, il nodo* 2 *ha come primo figlio* 3*, secondo figlio* 4*, terzo figlio* 6 *e quarto figlio* 5*. Le foglie non hanno figli. Osserviamo che gli alberi cardinali e gli alberi ordinali sono due strutture di dati* differenti*, nonostante l'apparente somiglianza: gli alberi nella Figura 4.7 sono distinti se considerati come alberi cardinali, in quanto il nodo D è il figlio destro del nodo B nel primo caso ed è il figlio sinistro nel secondo caso, mentre tali alberi sono indistinguibili come alberi ordinali in quanto D è il primo (e unico) figlio di B.*

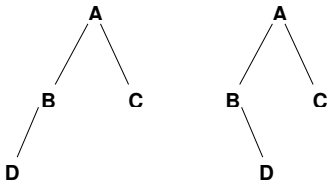


Figura 4.7: Alberi cardinali distinti che sono indistinguibili come alberi ordinali.

Grafi orientati: DAG

Nel caso di grafi orientati, un albero si chiama *arborescenza* e gli archi sono tutti orientati da padre a figlio, oppure tutti orientati da figlio a padre. Osserviamo però che questi non sono l'unico caso di grafi orientati aciclici.

Definizione 4.5.11. *Un grafo orientato aciclico è detto DAG (dall'inglese directed acyclic graph), dove i nodi con grado d'ingresso zero sono detti* sorgenti *e quelli con grado d'uscita zero sono detti* pozzi*. Nella terminologia delle relazioni, questa condizione di aciclicità si può esprimere come* $E^\star \cap id_V = \varnothing$*.*

Osservazione 4.5.12. *I DAG vengono utilizzati in quei contesti in cui esiste una dipendenza tra oggetti espressa da una relazione d'ordine: per esempio gli esami propedeutici in un corso di studi, la relazione di ereditarietà tra classi nella programmazione a oggetti, la relazione tra ingressi e uscite delle porte in un circuito logico, oppure l'ordine di valutazione delle formule in un foglio elettronico. In generale, supponiamo di avere decomposto un'attività complessa in un insieme di attività elementari e di avere individuato le relativa dipendenze. Un esempio concreto potrebbe essere la costruzione di un'automobile: volendo eseguire sequenzialmente le attività elementari (per esempio, in una catena di montaggio), bisogna soddisfare il vincolo che, se l'attività B dipende dall'attività A, allora A va eseguita prima di B. Usando i DAG per modellare tale situazione, poniamo i vertici in corrispondenza biunivoca con le attività elementari, e introduciamo un arco* (A, B) *per indicare che l'attività A va eseguita prima dell'attività B.*

Un'esecuzione in sequenza delle attività che soddisfi i vincoli di precedenza tra esse, corrisponde a un ordinamento topologico del DAG costruito su tali attività.

Definizione 4.5.13. *Dato un DAG* $G = (V, E)$*, un ordinamento topologico di G è una biiezione* $\eta : V \mapsto \{0, 1, \ldots, n-1\}$ *tale che per ogni arco* $(u, v) \in E$ *vale* $\eta(u) < \eta(v)$*. In altre parole, se disponiamo i vertici lungo una linea orizzontale in base alla loro numerazione* η*, in ordine crescente, otteniamo che gli archi risultano tutti orientati da sinistra verso destra.*

Proposizione 4.5.14. *Dato un DAG* $G = (V, E)$*, esiste sempre un suo ordinamento topologico.*

Dimostrazione. Costruiamo l'ordinamento topologico η a partire dal fondo. Deve esistere un pozzo u in G, altrimenti G sarebbe ciclico. Mettiamo tale u come ultimo nell'ordinamento η e rimuoviamo u e tutti gli archi entranti in u. Il grafo risultante G' sarà anch'esso un DAG. Prendiamo un pozzo u' e lo mettiamo subito prima di u nell'ordinamento η. Rimuoviamo u' e tutti gli archi entranti in u', e così via. A un certo punto avremo un solo nodo, che diventerà il primo elemento dell'ordinamento η. Sicccome rimuoviamo ogni volta un pozzo, tutti gli archi sono orientati nel modo richiesto, ossia $(v, u) \in E$ implica che $\eta(v) < \eta(u)$. ■

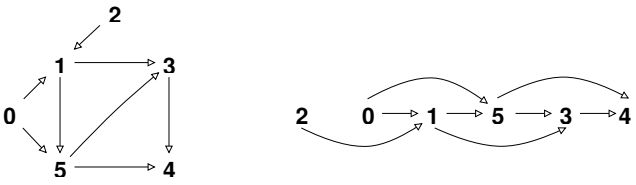


Figura 4.8: Un DAG con due sorgenti (0 e 2) e un pozzo (4). Rappresentazione grafica di un suo ordinamento topologico.

L'ordinamento topologico può anche essere visto come un ordinamento totale compatibile con l'ordinamento parziale rappresentato dal DAG. Osserviamo che ci possono essere più ordinamenti topologici per lo stesso DAG e che i grafi orientati ciclici non ammettono un ordinamento topologico.

4.6 Distanze

Analogamente alle distanze cartesiane, è possibile definire distanze nei grafi.

Definizione 4.6.1. *Una* distanza *(o metrica) su un insieme A è una funzione $d : A \times A \mapsto \mathbb{R}$ che soddisfa tre proprietà per ogni $x, y, z \in A$:*

4.6.1.1. $d(x, y) \geq 0$ *e* $d(x, x) = 0$*;*

4.6.1.2. $d(x, y) = d(y, x)$ *(simmetrica);*

4.6.1.3. $d(x, y) \leq d(x, z) + d(z, y)$ *(disuguaglianza triangolare).*

Si noti come la disuglianza triangolare sia intuitiva: un collegamento diretto da x a y non può essere peggiore di un collegamento che faccia tappa intermedia in z. Nel caso di grafi, possiamo definire la distanza sui nodi in V.

Definizione 4.6.2. *Dati due nodi $x, y \in V$, la loro* distanza $d(x, y)$ *è la* lunghezza *del cammino più breve tra x e y, chiamato* cammino minimo*.*

Esempio 4.6.3. *Nel grafo in Figura 4.1 la distanza tra i nodi* 0 *e* 4 *è 2, in quanto esistono i cammini minimi* $0, 1, 4$ *e* $0, 1, 5$*.*

La Definizione 4.6.2 è ben formata perché, sebbene tra due nodi x e y possano esserci più cammini minimi distinti, ogni cammino minimo è sempre un cammino semplice e la lunghezza è la medesima $d(x, y)$.

Proposizione 4.6.4. *Per ogni coppia di nodi $x, y \in V$, il cammino minimo è un cammino semplice.*

Dimostrazione. Procediamo per assurdo. Ipotizziamo che il cammino minimo P tra x e y non sia semplice: vuol dire che esiste un nodo z che appare almeno due volte in P. Ricordando che P è una sequenza di nodi, possiamo pertanto vedere P come $P = xP'zP''zP'''y$ dove alcuni dei cammini P', P'', P''' potrebbero essere vuoti. Osserviamo che anche $\hat{P} = xP'zP'''y$ è un cammino tra x e y, ed è sicuramente più breve di P perché passa una volta in meno per z rispetto a P. Questa è una contraddizione perché P è minimo per ipotesi. ■

Possiamo ora dimostrare che siamo in presenza di una distanza secondo la Definizione 4.6.1.

Proposizione 4.6.5. *La distanza $d()$ su grafi soddisfa la metrica data nella Definizione 4.6.1.*

Dimostrazione. Osserviamo che $d(x, y) \geq 0$ perché un cammino attraversa zero o più archi; inoltre, $d(x, x) = 0$ perché non occorre attraversare archi per rimanere in un nodo x (questo corrisponde a un cammino di lunghezza zero). Poichè i cammini non sono orientati, la lunghezza del cammino minimo tra x e y è uguale a quella del cammino minimo tra y e x (stesso cammino percorso inversamente) e quindi $d(x, y) = d(y, x)$ è simmetrica. Per la disuguaglianza triangolare vale tramite

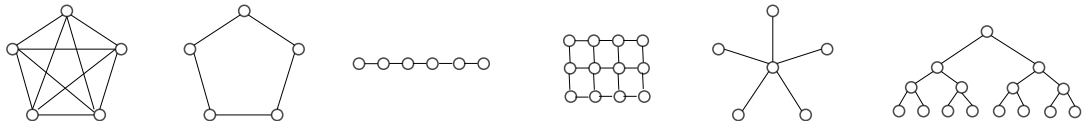


Figura 4.9: Esempi di grafi, da sinistra a destra: clique, ciclo, lineare, regolare (griglia), stella, albero completo

un ragionamento per assurdo: se fosse $d(x, y) > d(x, z) + d(z, y)$, avremmo trovato un cammino più breve del minimo, passando appunto per z, il che è impossibile per definizione di cammino minimo. ■

Osserviamo che, nel caso di grafi orientati, vale la stessa definizione basata sulla lunghezza del cammino minimo, con una grossa differenza rispeto ai grafi non orientati: la proprietà simmetrica *non* vale perchè il cammino minimo da x a y potrebbe essere di lunghezza diversa da quello da y o x, o addirittura non esistere (in tal caso si pone la distanza uguale a infinito). Le altre due proprietà valgono e si ottiene una *quasi-metrica* che viene spesso comunque chiamata distanza, perché si capsice dal contesto che è anti-simmetrica.

Basandosi sulla distanza, la seguente nozione è molto usata nello studio delle reti di calcolatori e quelle sociali.

Definizione 4.6.6. *Il* diametro *di un grafo G (orientato o meno) è la massima distanza tra coppie di nodi:*

$$\text{diam(G)} = \max_{x,y \in V} d(x, y)$$

Nel caso degli alberi, si eredita la distanza dei grafi non orientati.

Definizione 4.6.7. *Dato un albero radicato, la* profondità *di un nodo x è la sua distanza $d(x, r)$ dalla radice r, e l'*altezza *di x è la massima distanza tra x e le sue foglie discendenti. Un albero cardinale si dice* pieno *se è completo e le foglie sono tutte alla stessa distanza dalla radice.*

La profondità di un nodo è sempre minore dei suoi antenati e maggiore dei suoi discendenti: in particolare, la radice r ha profondità zero e quella degli altri nodi è sempre pari a uno più della profondità del genitore. Analogamente, ogni foglia ha altezza zero e ogni nodo interno ha altezza pari a uno più il massimo tra le altezze dei figli.

Infine, per i grafi pesati aventi pesi *non negativi*, si considera la *somma* dei pesi lungo ciascun cammino, piuttosto che la loro lunghezza. Per cammino minimo si intende quindi il *cammino pesato avente somma minima*: è importante osservare che un cammino minimo in un grafo pesato non è necessariamente quello col minor numero di archi. Per esempio in una rete stradale, è naturale pensare al peso di un arco come la lunghezza del tratto di strada corrispondente, e la lunghezza di un cammino è la somma dei pesi di tutti gli archi su di esso. Si può immediatamente verificare che la distanza pesata è una metrica nei grafi non orientati e una quasi-metrica nei grafi orientati. È naturalmente definito il diametro, essendo la distanza massima tra coppie di nodi. Invitiamo il lettore a pensare cosa può succedere con i pesi negativi: per esempio, se siamo in presenza di un ciclo in cui tutti gli archi hanno peso negativo.

4.7 Altri grafi noti

Terminiamo il capitolo con una breve rassegna informale di alcune classi di grafi famosi, illustrati in Figura 4.9. Una *clique* è un grafo in cui ogni coppia di nodi è collegata da un arco. Un grafo *d-regolare* ha ogni nodo di grado al più d, ed è *completo* se ogni nodo ha grado esattamente d (la clique è un caso particolare: d-regolare completo con $d = n - 1$). Un *ciclo* è un grafo ciclico composto da un solo ciclo. Un grafo *lineare* è un grafo aciclico composto da un solo cammino semplice. Una *stella* ha un nodo universale e gli altri nodi sono foglie. L'albero completo è descritto nella Definizione 4.6.7.

4.8 Esercizi

Esercizio 4.8.1. *Dato un grafo non orientato G con n nodi, determinare il massimo numero di archi.*

Esercizio 4.8.2. *Dato un grafo orientato $G = (V, E)$, si consideri E come una relazione su V. Mostrare che valgono le seguenti proprietà per i suoi gradi d'ingresso e d'uscita:*

1. *E è totale se e solo se $d_x^+ \geq 1$ per ogni $x \in V$;*
2. *E è univalente se e solo se $d_x^+ \leq 1$ per ogni $x \in V$;*
3. *E è iniettiva se e solo se $d_x^- \leq 1$ per ogni $x \in V$;*
4. *E è surgettiva se e solo se $d_x^- \geq 1$ per ogni $x \in V$.*

Esercizio 4.8.3. *Data la matrice di adiacenza di un grafo non orientato G, indicare cosa si ottiene sommando i valori 1 su ogni riga e colonna. E nel caso di grafo orientato?*

Esercizio 4.8.4. *Mostrare che l'inverso di un isomorfismo (da G_1 a G_2) è ancora un isomorfismo (da G_2 a G_1).*

Esercizio 4.8.5. *Dimostrare che l'isomorfismo tra grafi è una relazione di equivalenza.*

Esercizio 4.8.6. *Dimostrare che in un grafo non orientato, se esiste un cammino tra due nodi (ad es. $0, 1, 2, 1, 3$ nel grafo a destra della Figura 4.1), allora esiste un cammino semplice tra essi (ad es. $0, 1, 3$). Analogamente, se esiste un cammino chiudo, esiste un ciclo.*

Esercizio 4.8.7. *Dato un grafo $G = (V, E)$, si consideri E come una relazione simmetrica su V. Si consideri la relazione $R = E \cup id_V$. Mostrare che G è connesso se e solo se $R^\star = V \times V$.*

Esercizio 4.8.8. *Dimostrare che la relazione $E^\star$ per le componenti connesse rispetta le proprietà riflessiva, transitiva e simmetrica.*

Esercizio 4.8.9. *Dimostrare che due componenti connesse distinte non possono avere un nodo in comune.*

Esercizio 4.8.10. *Dati due insiemi $A, B \subseteq U$, l'indice di Jaccard $J(A, B) = \frac{|A \cap B|}{|A \cup B|}$ misura la similitudine tra A e B (infatti vale 1 se sono uguali e 0 se sono completamente disgiunti. Mostrare che l'indice di Jaccard è una distanza.*

Esercizio 4.8.11. *Trovare un controesempio alla disuguaglianza triangolare per la distanza sui grafi pesati nel caso che si ammettano pesi negativi.*

Esercizio 4.8.12. *Escogitare un modo per rappresentare in maniera univoca e non ambigua un albero utilizzando solo delle sequenze di parentesi.*

Esercizio 4.8.13. *Dimostrare che ogni DAG ha almeno una sorgente e almeno un pozzo.*

CAPITOLO 5

Induzione e Ricorsione

La maggior parte degli oggetti di studio dell'informatica sono definiti *induttivamente*, cioè sono definiti in termini di istanze più piccole di loro stessi. Numeri, liste, alberi binari e anche gli stessi programmi sono costruiti a partire da oggetti più piccoli dello stesso tipo. Per esempio, due programmi messi uno dopo l'altro (tipicamente con un punto e virgola nel mezzo) in modo che l'esecuzione del secondo inizi quando il primo è terminato, non sono altro che un programma definito in termini di due programmi più piccoli. Inoltre le funzioni definite su questi oggetti sono spesso definite anche esse in modo induttivo: il valore della funzione su un oggetto definito induttivamente è ottenuto componendo i valori della stessa funzione su oggetti più piccoli. La stessa tecnica si può usare più in generale per definire relazioni o proprietà su tali oggetti.

Le definizioni induttive, oltre a essere concise e matematicamente rigorose, garantiscono la correttezza di una tecnica di dimostrazione molto efficace, chiamata il *principio di induzione strutturale*, che può essere usato appunto per dimostrare proprietà di funzioni o relazioni definite su insiemi presentati induttivamente. Come vedremo, le definizioni induttive sono un caso particolare delle definizioni *ricorsive*, nelle quali in generale si permette di definire una funzione in termini del suo valore su oggetti arbitrari, non necessariamente più piccoli.

5.1 I naturali, i numeri triangolari e la formula di Gauss

Cominciamo a esplorare questi concetti presentando un semplice esempio (probabilmente già noto al lettore) che comprende (1) la definizione induttiva di un insieme (i naturali), (2) la definizione induttiva di una funzione sui naturali e (3) una dimostrazione per induzione di una semplice proprietà di tale funzione. L'esempio ha sia lo scopo di introdurre la terminologia rilevante che quello di mostrare come questi tre aspetti dell'induzione si complementano.

Cominciamo col chiederci: ma perché abbiamo bisogno di una nuova definizione dell'insieme $\mathbb{N}$ dei naturali? Il lettore attento avrà osservato che nel Capitolo 1 abbiamo introdotto questo insieme come

- $\mathbb{N} = \{0, 1, 2, ...\}$ (i **naturali**)

Questa definizione si basa sulla nostra capacità di capire la regola suggerita dai puntini sospensivi, che ci dice che aggiungendo 1 a qualunque elemento di questo insieme otteniamo il prossimo elemento dell'insieme. Ma questo tipo di definizione presenta diversi problemi,

- Come possiamo supporre che un'entità non intelligente (come un computer) possa capire questa definizione? Quanto meno dovremmo rendere esplicita la regola che genera gli elementi di questo insieme.

- Come possiamo essere sicuri che la regola che abbiamo inferito sia corretta? E se avessimo voluto definire l'insieme delle cifre decimali $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$? Oppure l'insieme delle radici dell'equazione $x^3 - 3x^2 + 2x = 0$, che sono proprio 0, 1 e 2?

- Abbiamo detto più volte che l'ordine in cui si elencano gli elementi di un insieme è irrilevante, ma allora che senso ha parlare del "prossimo elemento" di un insieme?

- Come possiamo determinare se un elemento appartiene oppure no all'insieme? Per esempio, come sappiamo che $\sqrt{9}$ appartiene ad $\mathbb{N}$ mentre $\sqrt{10}$ no?

Un modo semplice di definire una collezione infinita di oggetti consiste nel fornire una regola per generare nuovi elementi a partire da quelli esistenti, con la tecnica descritta di seguito.

Definizione induttiva di un insieme

In generale, la DEFINIZIONE INDUTTIVA DI UN INSIEME è costituita da tre componenti:

1. La CLAUSOLA BASE, che stabilisce che certi oggetti appartengono all'insieme. Questi elementi costituiscono i mattoncini per costruire altri elementi dell'insieme.
2. La CLAUSOLA INDUTTIVA, che descrive in che modo gli elementi dell'insieme possono essere usati per produrre altri elementi dell'insieme.
3. La CLAUSOLA TERMINALE, che stabilisce che l'insieme che si sta definendo non contiene altri elementi oltre a quelli ottenuti dalle due clausole precedenti. Quindi l'insieme definito è il *più piccolo* insieme che soddisfa la clausola base e quella induttiva.

Definizione 5.1.1 (Definizione induttiva di $\mathbb{N}$)**.** *L'insieme* $\mathbb{N}$ *dei numeri naturali è l'insieme di numeri che soddisfa le seguenti clausole:*

1. $0 \in \mathbb{N}$.
2. *Se* $n \in \mathbb{N}$ *allora* $(n+1) \in \mathbb{N}$.
3. *Nessun altro elemento appartiene a* $\mathbb{N}$.

Si noti che nella definizione consideriamo il concetto di "numero" e l'operazione di addizione come già conosciuti: implicitamente stiamo definendo i naturali come un sottoinsieme di un insieme di numeri più grande, come i reali. Questa definizione ci permette di mostrare per esempio che $\sqrt{9} = 3$ è un elemento di $\mathbb{N}$. Infatti dalla clausola base sappiamo che $0 \in \mathbb{N}$; quindi con la clausola induttiva otteniamo che $(0+1) = 1 \in \mathbb{N}$, e applicando ancora due volte quest'ultima otteniamo che $(1+1) = 2 \in \mathbb{N}$ e $(2+1) = 3 \in \mathbb{N}$, come desiderato. E cosa possiamo dire a proposito del numero 2.7, per esempio? Si osservi che l'insieme $\{0, 1, 1.7, 2, 2.7, 3, 3.7, 4, 4.7, \ldots\}$ soddisfa le prime due clausole della definizione e contiene il numero 2.7. Ma questo insieme non soddisfa la clausola terminale: infatti 2.7 è diverso da 0, e non può essere ottenuto da 0 aggiungendo 1 un numero qualunque di volte. In altre parole, questo insieme non è il più piccolo che soddisfa le prime due clausole: anche $\{0, 1, 2, 3, 4, \ldots\}$ le soddisfa, e non contiene 2.7. Quindi 2.7 non appartiene a $\mathbb{N}$, come sappiamo.

In modo più conciso ma del tutto equivalente, possiamo definire $\mathbb{N}$ come *il più piccolo insieme* che soddisfa:

1. $0 \in \mathbb{N}$.
2. Se $n \in \mathbb{N}$ allora $(n+1) \in \mathbb{N}$.

In questo caso abbiamo incorporato la clausola terminale nella premessa della definizione, dicendo che $\mathbb{N}$ è *il più piccolo insieme* che soddisfa le due clausole. Molto spesso le definizione induttive vengono presentate in questo modo, senza rendere esplicita la clausola terminale.

Definizione induttiva di una funzione

La definizione induttiva dell'insieme $\mathbb{N}$ può essere sfruttata per definire in modo conciso funzioni o relazioni definite su $\mathbb{N}$. Infatti per presentare la DEFINIZIONE INDUTTIVA DI UNA FUNZIONE su un insieme definito a sua volta induttivamente è sufficiente fornire (1) il valore della funzione sugli elementi che appartenono all'insieme per la clausola base, e (2) una regola per calcolare il valore della funzione sugli elementi che vi appartengono in base alla clausola induttiva, a partire dal suo valore sugli elementi che sono assunti appartenere all'insieme. Grazie alla clausola terminale della definizione induttiva siamo sicuri che i punti (1) e (2) sono sufficienti a definire la funzione per tutti

gli elementi dell'insieme. Per esempio, per definire una funzione $f : \mathbb{N} \to A$ è sufficiente fornire (1) il valore di $f(0)$ e (2) un'espressione che definisce $f(n+1)$ in termini di $f(n)$.

Per introdurre il prossimo esempio, ricordiamo che una successione di elementi di un insieme A,

$$a_0, a_1, a_2, \ldots a_n, \ldots$$

non è altro che una funzione $a : \mathbb{N} \to A$ che presentiamo scrivendo a_n invece di $a(n)$ per ogni $n \in \mathbb{N}$. Vediamo un esempio di funzione sui naturali definita induttivamente.

Definizione 5.1.2 (Definizione induttiva dei numeri triangolari)**.** *Per ogni $n \in \mathbb{N}$, il* NUMERO TRIANGOLARE *T_n è uguale alla somma di tutti i numeri minori o uguali a n:*

$$T_n = \sum_{i=0}^{n} i$$

In modo del tutto equivalente possiamo definire induttivamente la successione dei numeri triangolari (visti come una funzione $T : \mathbb{N} \to \mathbb{N}$) nel seguente modo:

1. $T_0 = 0$
2. $T_{n+1} = T_n + (n+1)$

Per esempio, $T_3 = T_2 + 3 = T_1 + 2 + 3 = T_0 + 1 + 2 + 3 = 0 + 1 + 2 + 3 = 6$. Quindi $T_3 = 6$, ma quanto vale T_n per un generico n? A questo risponde quella che è conosciuta, per motivi storici, come la *formula di Gauss*:

$$\forall n \in \mathbb{N}\,.\,\left(T_n = \frac{n \cdot (n+1)}{2}\right) \tag{5.1}$$

Come si può verificare la validità di questa formula, cioè che l'uguaglianza è vera per ogni numero $n \in \mathbb{N}$? Possiamo provare a vedere cosa succede in alcuni casi:

- Per $n = 0$ abbiamo $0 = \frac{0 \cdot 1}{2}$? OK!
- Per $n = 1$ abbiamo $0 + 1 = \frac{1 \cdot 2}{2}$? OK!
- Per $n = 2$ abbiamo $0 + 1 + 2 = \frac{2 \cdot 3}{2}$? OK!
- Per $n = 3$ abbiamo $0 + 1 + 2 + 3 = \frac{3 \cdot 4}{2}$? OK!
- ...

Ma questo non dimostra niente! In qualunque momento ci fermassimo, non avremmo dimostrato che la proprietà è valida, ma solo che l'uguaglianza è vera per un numero finito di valori (quelli esplicitamente considerati).

Il Principio di Induzione sui naturali

Per dimostrare la validità della formula di Gauss utilizzamo invece il PRINCIPIO DI INDUZIONE sui naturali. Data una generica proprietà $P(n)$ sui naturali, cioè un'asserzione che può essere vera o falsa al variare di $n \in \mathbb{N}$, il Principio di Induzione stabilisce che:

> Se (CASO BASE) $P(0)$ è vera, e se (PASSO INDUTTIVO) per ogni $n \in \mathbb{N}$ vale che se $P(n)$ è vera allora anche $P(n+1)$ lo è, allora $P(m)$ è vera per ogni $m \in \mathbb{N}$.

In modo più compatto possiamo scrivere il Principio di Induzione come una *regola di inferenza*:

$$\frac{P(0) \quad \forall n \in \mathbb{N}\,.\,(P(n) \Rightarrow P(n+1))}{\forall m \in \mathbb{N}\,.\,P(m)}\ \textbf{Principio di Induzione}$$

Introdurremo formalmente le regole di inferenza nel Capitolo 8. Informalmente, la regola dice che per dimostrare la formula sotto la linea è sufficiente dimostrare le formule elencate sopra la linea.

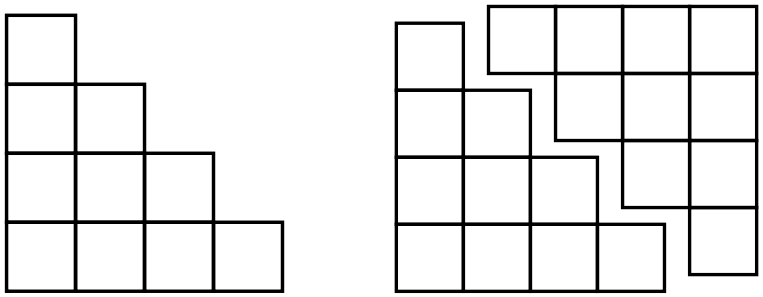


Figura 5.1: Rappresentazione grafica di T_4 e del suo doppio.

Proposizione 5.1.3 (Correttezza della formula di Gauss)**.** *La fomula* (5.1) *è valida, cioè l'uguaglianza* $T_n = \frac{n\cdot(n+1)}{2}$ *è vera per ogni* $n \in \mathbb{N}$.

Dimostrazione. Per il Principio di Induzione è sufficiente dimostrare i seguenti casi:

1. [Caso base] $T_0 = 0$ per definizione, e $\frac{0\cdot(0+1)}{2} = 0$, quindi la (5.1) è vera per $n = 0$
2. [Passo induttivo] Assumiamo che $T_n = \frac{n\cdot(n+1)}{2}$ e facciamo vedere che la (5.1) è vera per $n + 1$, cioè che
$$T_{n+1} = \frac{(n+1)\cdot(n+2)}{2}$$
Infatti abbiamo:
$$\begin{array}{rcll} T_{n+1} & = & T_n + (n+1) & \text{(Punto 2 della Definizione 5.1.2)} \\ & = & \frac{n\cdot(n+1)}{2} + (n+1) & \text{(Ipotesi induttiva)} \\ & = & \frac{n\cdot(n+1)+2\cdot(n+1)}{2} & \\ & = & \frac{(n+2)\cdot(n+1)}{2} & \end{array}$$

■

Possiamo verificare intuitivamente la correttezza della formula di Gauss anche in modo grafico. Nella parte sinistra della Figura 5.1 vediamo una rappresentazione grafica del numero triangolare T_4, che spiega anche la scelta del nome. Nella parte destra abbiamo accostato in modo opportuno due copie di T_4 ottenendo un rettangolo di dimensione $4 \cdot 5$, per cui vediamo subito che $T_4 = \frac{4\cdot5}{2}$.

5.2 Induzione sui naturali

La definizione induttiva dei naturali dell'Esempio 5.1.1 è paradigmatica. Con piccole variazioni possiamo definire opportuni sottoinsiemi di $\mathbb{N}$. Per esempio, sostituendo nella clausola 0 con 1 si ottiene, ovviamente, una definizione induttiva di $\mathbb{N}^+$. L'insieme dei numeri dispari (e analogamente quello dei numeri pari) può essere definito induttivamente come segue.

Esempio 5.2.1 (Numeri dispari)**.** *L'insieme* $\mathbb{N}^d$ *dei numeri dispari è il più piccolo insieme che soddisfa:*

1. *Clausola base:* $1 \in \mathbb{N}^d$
2. *Clausola induttiva: Se* $n \in \mathbb{N}^d$ *allora* $(n + 2) \in \mathbb{N}^d$.

Esercizio 5.2.2. *L'insieme* $\mathbb{N}$ *soddisfa sia la clausola base (*$1 \in \mathbb{N}$*) che la clausola induttiva (se* $n \in \mathbb{N}$ *allora* $(n + 2) \in \mathbb{N}$*) della definizione precedente. Perché questo non implica che* $\mathbb{N} = \mathbb{N}^d$*?*

Esercizio 5.2.3. *Fornire una definizione induttiva dell'insieme Pow-2 delle potenze di 2:*

$$\textit{Pow-2} = \{1, 2, 4, 8, 16, 32, 64, 128, \ldots\}$$

Un esempio paradigmatico di funzione definita induttivamente sui naturali è il fattoriale.

Esempio 5.2.4 (Fattoriale)**.** *Il* fattoriale *di un numero $n \in \mathbb{N}$, scritto $n!$, è il prodotto di tutti i naturali da 1 a n. Induttivamente:*

1. $0! = 1$
2. $(n+1)! = (n+1) \cdot n!$

Si noti la somiglianza con la definizione della successione di numeri triangolari: nella clausola induttiva la somma è rimpiazzata con il prodotto, e nella clausola base 0 è rimpiazzato con 1. Quest'ultima potrebbe sembrare una scelta arbitaria, ma non lo è: in entrambe le definizioni induttive il primo elemento della successione è l'elemento neutro dell'operazione usata nella clausola induttiva. Infatti 0 è l'elemento neutro della somma, mentre 1 è quello del prodotto.

Vediamo un esempio di dimostrazione per induzione di una proprietà del fattoriale.

Proposizione 5.2.5 (Il fattoriale cresce più rapidamente dell'esponenziale)**.** *Per ogni $n \in \mathbb{N}^+$ vale che $n! \geq 2^{n-1}$.*

Dimostrazione. Procediamo per induzione, osservando che poiché si assume $n \in \mathbb{N}^+$, il caso base dovrà considerare 1 e non 0.

1. [Caso base] Abbiamo per definizione che $1! = 1 \cdot 0! = 1 \cdot 1 = 1$, e $2^{1-1} = 2^0 = 1$, quindi $1! \geq 2^0$
2. [Passo induttivo] Assumendo per ipotesi che $n! \geq 2^{n-1}$, mostriamo che $(n+1)! \geq 2^n$. Infatti

$$\begin{array}{rcll} (n+1)! & = & (n+1) \cdot n! & \text{(Clausola induttiva dell'Esempio 5.2.4)} \\ & \geq & (n+1) \cdot 2^{n-1} & \text{(Ipotesi induttiva)} \\ & \geq & 2 \cdot 2^{n-1} & (n \in \mathbb{N}^+, \text{ quindi } n+1 \geq 2) \\ & = & 2^n & \end{array}$$

■

Esercizio 5.2.6. *Si consideri la successione*

1. $S_0 = 0$
2. $S_{n+1} = S_n + 1 + 2 \cdot n$

Si dimostri per induzione che $\forall n \in \mathbb{N} \,.\, S_n = n^2$.

Esercizio 5.2.7 (*)**.** *Si dimostri per induzione la validità della* formula di Nicodemo*:*

$$\forall n \in \mathbb{N} \,.\, \left(\sum_{k=0}^{n} k^3 = T_n^2 \right)$$

Quindi la somma dei cubi dei numeri naturali minori o uguali a n è uguale al quadrato dell'n-simo numero triangolare (Definizione 5.1.2).

Esempio 5.2.8 (Generalizzazione di De Morgan su insiemi)**.** *Vediamo un esempio che combina insiemi e induzione. Abbiamo già usato molte volte le leggi di De Morgan, come*

$$\overline{(A \cup B)} = \overline{A} \cap \overline{B}$$

Ma possiamo generalizzate questa legge a più insiemi? Per esempio, è vero che

$$\forall A, B, C \,.\, \left(\overline{(A \cup B \cup C)} = \overline{A} \cap \overline{B} \cap \overline{C} \right)?$$

Usando l'induzione facciamo vedere come queste leggi possano essere generalizzate a un numero finito qualunque di insiemi.

Sia

$$DM(n) = \forall A_1, ..., A_n \,.\, \left(\overline{\left(\bigcup_{i=1}^{n} A_i \right)} = \bigcap_{i=1}^{n} \overline{A_i} \right)$$

Dimostriamo che:

$$\forall n \,.\, (\, n \geq 2 \;\Rightarrow\; DM(n)\,)$$

1. *[Caso base] Dobbiamo dimostrare $DM(2)$, ovvero che presi due insiemi qualsiasi A_1 e A_2 si ha*

$$\overline{(A_1 \cup A_2)} = \overline{A_1} \cap \overline{A_2}$$

OK (è la ben nota legge di De Morgan)!

2. *[Passo induttivo] Prendiamo un generico n. Assumiamo che $DM(n)$ sia vero e dimostriamo che vale anche $DM(n+1)$.*

Se vale $DM(n)$ vuol dire che

$$\forall A_1, ..., A_n \left(\overline{\left(\bigcup_{i=1}^{n} A_i \right)} = \bigcap_{i=1}^{n} \overline{A_i} \right)$$

Vogliamo dimostrare che

$$\forall A_1, ..., A_n, A_{n+1} \left(\overline{\left(\bigcup_{i=1}^{n+1} A_i \right)} = \bigcap_{i=1}^{n+1} \overline{A_i} \right)$$

Prendiamo $n+1$ insiemi qualsiasi $A_1, ..., A_n, A_{n+1}$. Notiamo che:

$$\begin{aligned}
\overline{\left(\bigcup_{i=1}^{n+1} A_i \right)} &= \overline{\left(A_{n+1} \cup \bigcup_{i=1}^{n} A_i \right)} \\
&= \overline{A_{n+1}} \cap \overline{\left(\bigcup_{i=1}^{n} A_i \right)} \quad \textit{(per De Morgan)} \\
&= \overline{A_{n+1}} \cap \bigcap_{i=1}^{n} \overline{A_i} \quad \textit{(per ipotesi induttiva)} \\
&= \bigcap_{i=1}^{n+1} \overline{A_i}
\end{aligned}$$

Vediamo ora un esempio di uso dell'induzione che era stato anticipato in un capitolo precedente.

Esempio 5.2.9 (Sequenze e prodotto cartesiano)**.** *Nella Definizione 2.7.2 abbiamo introdotto l'insieme A^n di tutte le sequenze di lunghezza n di elementi di un dato insieme A, definito come:*

$$A^n = \{(a_0, a_1, \ldots, a_{n-1}) \mid \forall i \in \{0, \ldots, n-1\} \,.\, a_i \in A\}$$

Inoltre nell'Osservazione 2.7.4 abbiamo affermato che $A^0 \cong 1$ e che in generale

$$A^n \cong \underbrace{A \times (\cdots \times (A \times A) \ldots)}_{n \text{ volte}}$$

cioè che esiste una biiezione tra questi due insiemi. Dimostriamo per induzione che questa affermazione è vera.

1. *[Caso base] Il fatto che* $A^0 \cong 1$ *segue immediatamente dalle definizioni. Infatti* $A^0 = \{(\,)\}$ *contiene un solo elemento, la sequenza vuota. Inoltre dall'enunciato è chiaro che stiamo considerando* 1 *come un insieme e quindi (si veda la fine della Sezione 1.5) anche* $1 = \{0\}$ *contiene un solo elemento. Quindi abbiamo una ovvia biiezione* $i_0 : A^0 \to 1$ *definita da* $i_0((\,)) = 0$.

2. *[Passo induttivo] Assumiamo di avere una biiezione*

$$i_n : A^n \to \underbrace{A \times (\cdots \times (A \times A) \ldots)}_{n \text{ volte}}$$

e mostriamo che esiste una biiezione

$$i_{n+1} : A^{n+1} \to \underbrace{A \times (\cdots \times (A \times A) \ldots)}_{n+1 \text{ volte}}$$

Infatti poiché ogni sequenza di $n+1$ *elementi può essere vista come un coppia composta di un elemento e di una sequenza di n elementi, abbiamo una ovvia biiezione* $j_{n+1} : A^{n+1} \to A \times A^n$ *definita da* $(a_0, a_1, \ldots, a_n) \mapsto (a_0, (a_1, \ldots, a_n))$. *Sfruttando la biiezione* i_n *che esiste per ipotesi induttiva possiamo definire la biiezione desiderata come*

$$i_{n+1}((a_0, a_1, \ldots, a_n)) = (a_0, i_n(a_1, \ldots, a_n))$$

Nel Capitolo 3, data una relazione R su un insieme A, $R \subseteq A \times A$, abbiamo definito induttivamente la relazione R^n (Definizione 3.3.13) e abbiamo introdotto la chiusura transitiva $R^\star$ (Definizione 3.3.25). Nella Tabella 3.1 abbiamo presentato alcune leggi di distributività dell'operazione $(.)^\star$ che possono essere dimostrate usando l'induzione sui naturali. Vediamo in particolare la distributività di $\star$ su $\cup$ e la distributività di $\star$ su $\cdot^{op}$.

Esempio 5.2.10 (Distributività della stella di Kleene sull'unione di relazioni)**.** *Dimostriamo che* $R^\star \cup S^\star \subseteq (R \cup S)^\star$.
Per cominciare, dimostriamo per induzione che

$$\text{Se } R \subseteq S, \text{ allora } R^n \subseteq S^n \text{ per ogni } n \in \mathbb{N} \tag{5.2}$$

Infatti per $n = 0$ *(caso base) abbiamo* $R^0 = Id_A = S^0$. *Invece per* $n = m + 1$ *(passo induttivo) abbiamo*

$$\begin{array}{lcll} R^n & = & R; R^m & \\ & \subseteq & S; R^m & \textit{(Ipotesi: } R \subseteq S\textit{)} \\ & \subseteq & S; S^m & \textit{(Ipotesi induttiva: } R^m \subseteq S^m\textit{)} \\ & = & S^n & \end{array}$$

Mostriamo ora che

$$\text{Se } R \subseteq S, \text{ allora } R^\star \subseteq S^\star \tag{5.3}$$

Infatti abbiamo:

$$R^\star \quad = \quad \bigcup_{n \in \mathbb{N}} R^n \quad \textit{(Definizione 3.3.25)} \quad \subseteq \quad \bigcup_{n \in \mathbb{N}} S^n \quad \textit{(per (5.2))} \quad = \quad S^\star$$

Infine dimostriamo che $R^\star \cup S^\star \subseteq (R \cup S)^\star$. *Infatti abbiamo*

- $R \subseteq (R \cup S)$, *e quindi* $R^\star \subseteq (R \cup S)^\star$ *(per (5.3))*
- $S \subseteq (R \cup S)$, *e quindi* $S^\star \subseteq (R \cup S)^\star$ *(per (5.3))*
- *e quindi* $R^\star \cup S^\star \subseteq (R \cup S)^\star$ *(per proprietà dell'unione)*

Esempio 5.2.11 (Distributività della stella di Kleene sull'opposto di una relazione)**.** *Dimostriamo che* $(R^\star)^{op} = (R^{op})^\star$
Vediamo prima che

$$R; R^n = R^n; R \quad \textit{per ogni } n \in \mathbb{N} \tag{5.4}$$

Infatti se $n = 0$ *(caso base) abbiamo* $R; R^0 = R; Id_A = R = Id_A; R = R^0; R$.
Per $n = m + 1$ *(passo induttivo) abbiamo*

$$\begin{array}{rcll} R; R^{m+1} & = & R; (R; R^m) & \\ & = & R; (R^m; R) & \textit{(Ipotesi induttiva)} \\ & = & (R; R^m); R & \textit{(associatività)} \\ & = & R^{m+1}; R & \end{array}$$

Dimostriamo ora per induzione che

$$(R^n)^{op} = (R^{op})^n \quad \textit{per ogni } n \in \mathbb{N} \tag{5.5}$$

Infatti se $n = 0$ *(caso base) abbiamo* $(R^0)^{op} = (Id_A)^{op} = Id_A = (R^{op})^0$.
Per $n = m + 1$ *(passo induttivo) abbiamo*

$$\begin{array}{rcll} (R^n)^{op} & = & (R^{m+1})^{op} & \\ & = & (R; R^m)^{op} & \\ & = & (R^m)^{op}; R^{op} & \textit{(Distrib. di } \cdot^{op}\textit{)} \\ & = & (R^{op})^m; R^{op} & \textit{(Ipotesi induttiva)} \\ & = & R^{op}; (R^{op})^m & \textit{(per (5.4))} \\ & = & R^{op}; (R^{op})^{m+1} = (R^{op})^n & \end{array}$$

Infine possiamo concludere osservando che

$$(R^\star)^{op} = \left(\bigcup_{n\in\mathbb{N}} R^n\right)^{op} \textit{(Def. 3.3.25)} = \bigcup_{n\in\mathbb{N}} (R^n)^{op} \textit{(Dist. di } \cdot^{op} \textit{ su } \cup\textit{)} = \bigcup_{n\in\mathbb{N}} (R^{op})^n \textit{(per (5.5))} = (R^{op})^\star$$

Esercizio 5.2.12. *Ora che abbiamo tutti gli strumenti necessari, si dimostri per induzione sul numero di nodi il punto (b) della Proposizione 4.5.5, che afferma che ogni albero con n nodi ha esattamente* $n - 1$ *archi.*

Esempio 5.2.13 (Numeri di Fibonacci)**.** *Una successione numerica molto famosa è quella dei* NUMERI DI FIBONACCI*, che prendono il nome dal matematico pisano che li introdusse per risolvere il famoso* problema dei conigli*.*

Il problema consiste nello studiare la crescita di una popolazione di conigli su un'isola, assumendo che:

- *Al mese* 0 *viene posta una sola coppia di conigli (maschio e femmina) appena nati;*
- *Una coppia di conigli si riproduce solo quando entrambi hanno più di due mesi;*
- *Passati due mesi di età, ogni coppia produce un'altra coppia di conigli ogni mese;*
- *I conigli non muoiono mai, hanno tutto il cibo necessario e sull'isola non ci sono predatori;*
- Qual è il numero f_n di coppie di conigli dopo n mesi?

Ragioniamo sul problema:

- *Al mese* 1 *c'è una sola coppia* ($f_1 = 1$)*;*
- *Al mese* 2 *c'è ancora una sola coppia* ($f_2 = 1$)*;*
- *Al mese* 3 *l'unica coppia presente sull'isola ne produce un'altra* ($f_3 = 2$)*;*

- *Al mese 4 la coppia più vecchia ne produce un'altra, mentre l'altra coppia è ancora troppo giovane ($f_4 = 3$);*
- *Al mese n ci sono ancora tutte le coppie che esistevano già al mese $n-1$ e di queste, tutte (e sole) le coppie che esistevano al mese $n-2$ adesso possono generare figli ($f_n = f_{n-1} + f_{n-2}$);*

Quindi la successione di Fibonacci è definita induttivamente come segue:

1. $f_1 = 1$
2. $f_2 = 1$
3. $f_n = f_{n-1} + f_{n-2} \qquad se\ n > 2$

Si osservi che nella definizione induttiva della successione di Fibonacci la clausola base determina due valori, f_1 e f_2, e la clausola induttiva definisce un elemento usando i due elementi precedenti. Questo mostra che le definizioni induttive di funzioni possono assumere una varietà di forme. Ma chi ci garantisce che una definizione così data sia corretta? Torneremo su questo problema nella Sezione 5.4: per il caso in esame ci convinciamo facilemente che la definizione è ben data perché f_n è definita in termini dei due valori *precedenti* della sequenza, e la clausola viene applicata solo per $n > 2$, quindi tal valori esistono.

La successione di Fibonacci gode di moltissime proprietà interessanti ed è argomento di studi approfonditi nella teoria dei numeri. Vediamo un esempio e lasciamo il successivo come esercizio per il lettore.

Esempio 5.2.14. *Sia f_i l'i-esimo numero di Fibonacci. Dimostriamo per induzione che*

$$\forall n \in \mathbb{N}^+ . \left(\sum_{i=1}^{n} f_i^2 = f_n \cdot f_{n+1} \right)$$

1. [Caso base] *Poiché l'asserto è per tutti gli $n \in \mathbb{N}^+$, per il caso base dobbiamo considerare $n = 1$. Quindi dobbiamo dimostrare che $\sum_{i=1}^{1} f_i^2 = f_1 \cdot f_2$. Ma questo segue immediatamente dalla definizione di f_1 e f_2: $\sum_{i=1}^{1} f_i^2 = f_1^2 = 1 = f_1 \cdot f_2$.*

2. [Passo induttivo] *Assumiamo per ipotesi induttiva che valga $\sum_{i=1}^{n} f_i^2 = f_n \cdot f_{n+1}$. Allora*

$$\begin{array}{rcll} \sum_{i=1}^{n+1} f_i^2 & = & \sum_{i=1}^{n} f_i^2 + f_{n+1}^2 & \textit{(Togliamo l'ultimo termine dalla sommatoria)} \\ & = & f_n \cdot f_{n+1} + f_{n+1}^2 & \textit{(Ipotesi induttiva)} \\ & = & (f_n + f_{n+1}) \cdot f_{n+1} & \textit{(Distributività di } \cdot \textit{ rispetto a } +\textit{, al contrario)} \\ & = & f_{n+2} \cdot f_{n+1} & \textit{(Clausola induttiva della successione di Fibonacci)} \end{array}$$

Esercizio 5.2.15. *Per concludere questa sezione, vediamo se riusciamo a convincervi che tutti i gatti hanno lo stesso colore con la seguente dimostrazione induttiva.*

Sia $C(n) =$ "preso un qualsiasi insieme di n gatti, questi hanno tutti lo stesso colore" *Vogliamo dimostare che $\forall n \in \mathbb{N}^+ . C(n)$. Procediamo per induzione su n:*

- [Caso base] *Dobbiamo dimostrare che vale $C(1)$. Ma chiaramente, se l'insieme è composto di un solo gatto allora la proprietà è banalmente soddisfatta.*
- [Passo induttivo] *Prendiamo un generico n. Assumiamo che $C(n)$ sia vero e dimostriamo che vale anche $C(n+1)$.*

  Allineiamo gli $n+1$ gatti dell'insieme e consideriamo i due sottoinsiemi contenenti i primi n gatti e gli ultimi n gatti

  Per ipotesi induttiva, i primi n gatti hanno lo stesso colore. Per ipotesi induttiva, gli ultimi n gatti hanno lo stesso colore.

  Ma allora per transitività tutti gli $n+1$ gatti hanno lo stesso colore!

Il Principio di Induzione Forte sui naturali

Qualche volta il Principio di Induzione sui naturali non è abbastanza "potente" per completare la dimostrazione di una certa proprietà $P(n)$ in maniera agevole, perché per dimostrare che $P(n+1)$ è vera non basta l'ipotesi induttiva che $P(n)$ è vera, ma è conveniente assumere che $P(m)$ valga anche per altri valori minori di m.

Il Principio di Induzione Forte sui naturali ci permette di rafforzare le ipotesi del passo induttivo e portare avanti la dimostrazione in modo più semplice. Esso stabilisce che:

> Se per ogni $n \in \mathbb{N}$ vale che se $P(0), P(1), \ldots, P(n-1)$ sono vere allora anche $P(n)$ lo è, allora $P(m)$ è vera per ogni $m \in \mathbb{N}$.

In modo più compatto, come regola di inferenza:

$$\frac{\forall n \,.\, (P(0) \ \wedge \ P(1) \ \wedge \ \cdots \ \wedge \ P(n-1) \Rightarrow P(n))}{\forall m \,.\, P(m)} \ \textbf{Induzione Forte}$$

Si noti che il caso base non è scomparso, ma è inglobato nell'unica premessa della regola. Infatti per $n = 0$ il fatto che $P(0)$ valga deve essere dimostrato senza poter assumere alcuna premessa. Come si vede, per dimostrare che $P(n)$ valga, nel passo induttivo possiamo assumere non solo che $P(n-1)$ sia vero, ma che lo siano anche $P(0)$, $P(1)$, $P(2)$, ..., $P(n-2)$, cioè preso un n generico abbiamo più ipotesi a disposizione per dimostrare $P(n)$. Nonstante l'apparenza, non è difficile mostrare che il Principio di Induzione Forte e quello ordinario sono equivalenti, nel senso che permettono di dimostrare le stesse proprietà, ma questo va oltre gli obbiettivi di queste note.

Esempio 5.2.16 (Il Teorema fondamentale dell'Aritmentica)**.** *Usando il Principio di Induzione Forte sui naturali dimostriamo che* ogni intero n maggiore di 1 o è un numero primo oppure può essere scritto come prodotto di numeri primi. *Questo enunciato è parte del* Teorema fondamentale dell'Aritmetica *che stabilisce anche l'unicità di tale decomposizione.*

Procediamo quindi per induzione forte su n. Assumiamo che $n > 1$ e che ogni intero k con $1 < k < n$ o è primo oppure è esprimibile come prodotto di numeri primi. Se n stesso è primo non c'è niente da dimostrare, quindi supponiamo che $n = a \cdot b$, con $1 < a < n$ e $1 < b < n$. Per l'ipotesi induttiva ognuno tra a e b o è primo oppure è esprimibile come prodotto di primi. Ma allora poiché $n = a \cdot b$ anche n è esprimibile come un prodotto di numeri primi.

Esercizio 5.2.17. *Si consideri la funzione*

$$f(n) = \begin{cases} 0, & \text{se } n = 0 \\ 2 \cdot f(\frac{n}{2}), & \text{se } n \text{ è pari} \\ f(n-1) + 1, & \text{se } n \text{ è dispari.} \end{cases}$$

Si dimostri per induzione forte che $f(n) = n$ per ogni $n \in \mathbb{N}$.

Esercizio 5.2.18. *Sia f_i l'i-esimo numero di Fibonacci. Dimostrare per induzione forte che la seguente* formula di Binet *è corretta per ogni $n \in \mathbb{N}^+$:*

$$f_n = \frac{1}{\sqrt{5}} \cdot \left(\frac{1+\sqrt{5}}{2}\right)^n - \frac{1}{\sqrt{5}} \cdot \left(\frac{1-\sqrt{5}}{2}\right)^n$$

5.3 Induzione su insiemi sintattici

L'insieme $\mathbb{N}$ dei numeri naturali, come presentato nella Definizione 5.1.1, è inteso contenere valori *semantici*, non oggetti *sintattici*, nel senso che esso contiene tali numeri indipendentemente dal modo in cui li scriviamo. Per esempio 3, $\sqrt{9}$, $((0+1)+1)+1$, e 11 (in notazione binaria) sono quattro rappresentazioni sintattiche diverse dello stesso elemento $3 \in \mathbb{N}$. Inoltre si osservi che abbiamo utilizzato liberamente le classiche operazioni (somma, moltiplicazione, elevazione a potenza, ...), dando per scontato che il lettore sappia come sono definite queste operazioni sui naturali.

Tuttavia in Informatica siamo spesso interessati a definire insiemi (tipicamente infiniti) di oggetti puramente sintattici, perché di fatto i programmi che scriviamo possono manipolare solo ben

definite rappresentazioni sintattiche degli oggetti del mondo reale. In questa sezione presentiamo alcuni di questi insiemi sfruttando la tecnica di definizione per induzione, e tratteremo solo in modo informale, quando conveniente, la loro associazione con valori semantici.

Seguendo lo stesso schema delle sezioni precedenti, per ognuno di questi insiemi sfrutteremo la definizione induttiva per definire delle funzioni su di essi. Inoltre dimostreremo opportune proprietà di tali funzioni utilizzando il **Principio di Induzione Strutturale**, che è del tutto analogo al Principio di Induzione sui naturali che abbiamo già visto, ma riferito ad una arbitraria definizione induttiva.

Stringhe

Come primo esempio rivisitiamo una definizione del Capitolo 2. Dato un insieme A, abbiamo introdotto nella Definizione 2.7.8 l'insieme $A^\star$ delle sequenze di lunghezza arbitraria di elementi di A come

$$A^* = \bigcup_{n\in\mathbb{N}} A^n$$

cioè come l'unione, al variare di $n \in \mathbb{N}$, degli insiemi A^n che contengono tutte le sequenze di lunghezza n di elementi di A. Spesso questa definizione viene usata nel contesto della teoria dei linguaggi formali, dove tipicamente A è un insieme di *simboli* o *caratteri*, chiamato *alfabeto*, e le sequenze sono chiamate *stringhe*. Vediamo una definizione induttiva del tutto equivalente.

Definizione 5.3.1. *Sia A un alfabeto, cioè un insieme finito di simboli. L'insieme $A^\star$ delle* STRINGHE *su A è il più piccolo insieme che soddisfa:*

1. [CLAUSOLA BASE] $\varepsilon \in A^\star$, *dove ε è la* stringa vuota.[1]
2. [CLAUSOLA INDUTTIVA] *Se $w \in A^\star$ e $a \in A$ allora $wa \in A^\star$.*

Dalla definizione segue che in generale una stringa ha la forma $\varepsilon a_0 a_1 \dots a_k$, dove $a_0, a_1, \dots, a_k$ sono simboli in A, ma spesso la scriveremo $a_0 a_1 \dots a_k$ eliminando la ε iniziale. L'insieme A^+ delle STRINGHE NON VUOTE su A è definito come $A^+ = A^\star \setminus \{\varepsilon\}$.

Esempio 5.3.2. *Se $A = \{a, b\}$ allora $A^\star$ è l'insieme seguente:*

$$A^\star = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \dots\}$$

Infatti,

- *per la clausola base, $\varepsilon \in A^\star$;*
- *per la clausola induttiva, poiché sappiamo che $\varepsilon \in A^\star$ e $\{a, b\} \subseteq A$, possiamo inferire che $\{\varepsilon a, \varepsilon b\} = \{a, b\} \subseteq A^\star$.*
- *applicando di nuovo la clausola induttiva, sapendo che $\{\varepsilon, a, b\} \subseteq A^\star$ possiamo dedurre che $\{a, b, aa, ab, ba, bb\} \subseteq A^\star$;*

e ogni nuova applicazione della clausola induttiva aggiunge nuove stringhe all'insieme.

Avremmo potuto definire gli insiemi $A^\star$ e A^+ in tanti modi equivalenti. Per esempio, avremmo potuto usare wa invece di (oppure insieme a) aw nella clausola induttiva; oppure avremmo potuto presentare A^+ con una definizione induttiva, invece di definirlo in termini di $A^\star$.

Esercizio 5.3.3. . *Fornire una definizione induttiva di A^+.*

Vediamo ora due semplici funzioni definite per induzione sulle stringhe: la *lunghezza* e la *concatenazione*.

Definizione 5.3.4. *La* LUNGHEZZA *di una stringa $w \in A^\star$, scritta $|w|$, è definita induttivamente nel modo seguente:*

1. $|\varepsilon| = 0$

[1] Assumiamo che $\varepsilon \notin A$.

2. $|wa| = |w| + 1$, *per ogni* $w \in A^\star$ *e* $a \in A$.

Queste clausole definiscono univocamente la funzione $|\_| : A^\star \to \mathbb{N}$.

La CONCATENAZIONE *di due stringhe* $u, w \in A^\star$*, scritta* $u \cdot w$*, è definita per induzione su* w *nel modo seguente:*

1. $u \cdot \varepsilon = u$

2. $u \cdot wa = (u \cdot w)a$

Queste clausole definiscono univocamente la funzione $\_ \cdot \_ : A^\star \times A^\star \to A^\star$.

Le parentesi tonde nella clausola induttiva della concatenazione indicano, come al solito, l'ordine di valutazione degli operatori. Infatti possiamo leggere $u \cdot wa$ come $u \cdot (wa)$ (la concatenazione di u con la stringa ottenuta aggiungendo a w il simbolo a), mentre $(u \cdot w)a$ è la stringa che si ottiene concatenando prima u e w, e poi aggiungendo a in fondo.

Per esempio, se $A = \{a, b\}$, abbiamo

$$\begin{array}{rcll} abb \cdot ba & = & abb \cdot \varepsilon ba & \text{(Rendiamo esplicito } \varepsilon \text{ nella seconda stringa)} \\ & = & (abb \cdot \varepsilon b)a & \text{(Def. di concatenazione, clausola induttiva)} \\ & = & ((abb \cdot \varepsilon)b)a & \text{(Def. di concatenazione, clausola induttiva)} \\ & = & ((abb)b)a & \text{(Def. di concatenazione, clausola base)} \\ & = & abbba & \text{(Rimuoviamo parentesi non necessarie)} \end{array}$$

Vediamo ora come grazie alla definizione induttiva delle stringhe su un alfabeto A e delle funzioni lunghezza e concatenazione possiamo dimostrare delle semplici proprietà, sfruttando il *Principio di Induzione Strutturale.*

Principio di Induzione Strutturale

Data una generica proprietà $P(x)$ su un insieme X definito induttivamente, cioè un'asserzione che può essere vera o falsa al variare di $x \in X$, il PRINCIPIO DI INDUZIONE STRUTTURALE stabilisce che:

> Se (CASO BASE) $P(x)$ è vera per tutti gli elementi x che appartengono a X per la *clausola base*, e se (PASSO INDUTTIVO) per ogni modo in cui la *clausola induttiva* prevede che si possano combinare alcuni elementi $x_0, \ldots, x_k$ di X per ottenere un nuovo elemento $x \in X$ vale che: se $P(x_0), \ldots, P(x_k)$ sono vere allora $P(x)$ è vera, allora $P(z)$ è vera per ogni $z \in X$

Da questo enunciato risulta evidente che il Principio di Induzione sui naturali introdotto nella Sezione 5.1 non è altro che il Principio di Induzione Strutturale applicato alla definizione induttiva dei numeri naturali. Nel caso specifico delle stringhe, quest'ultimo ci garantisce che se $P(\varepsilon)$ è vera, e se assumendo che $P(w)$ sia vera possiamo mostrare che anche $P(wa)$ lo è per ogni $a \in A$, allora $P(u)$ è vera per ogni $u \in A^\star$.

Proposizione 5.3.5 (lunghezza della concatenazione di due stringhe)**.** *Per tutte le stringhe* $u, w \in A^\star$ *vale che* $|u \cdot w| = |u| + |w|$.

Dimostrazione. Procediamo per induzione strutturale su $w \in A^\star$.

1. [CASO BASE] $(w = \varepsilon)$

$$\begin{array}{rcll} |u \cdot w| & = & |u \cdot \varepsilon| & \\ & = & |u| & \text{(Def. di concatenazione, clausola base)} \\ & = & |u| + 0 & \\ & = & |u| + |\varepsilon| & \text{(Def. di lunghezza, clausola base)} \\ & = & |u| + |w| & \end{array}$$

2. [Passo induttivo] Supponiamo che $w = va$ e assumiamo come ipotesi induttiva che per ogni $u \in A^\star$ valga $|u \cdot v| = |u| + |v|$.

$$\begin{array}{rcll} |u \cdot w| & = & |u \cdot va| & \\ & = & |(u \cdot v)a| & \text{(Def. di concatenazione, clausola induttiva)} \\ & = & |(u \cdot v)| + 1 & \text{(Def. di lunghezza, clausola induttiva)} \\ & = & |u| + |v| + 1 & \text{(Ipotesi induttiva)} \\ & = & |u| + |va| & \text{(Def. di lunghezza, clausola base)} \\ & = & |u| + |w| & \end{array}$$

■

Esercizio 5.3.6. *Si dimostri per induzione strutturale che per ogni $w \in A^\star$ e per ogni $a \in A$ valgono le seguenti uguaglianze:*

1. $\varepsilon \cdot w = w$
2. $w \cdot a = wa$

Proposizione 5.3.7. *Sia A un alfabeto che contiene (almeno) due simboli distinti a e b. Allora per qualunque stringa $w \in A^\star$ vale che $a \cdot w \neq w \cdot b$.*

Dimostrazione. Procediamo per induzione strutturale.

1. [Caso base] ($w = \varepsilon$) In questo caso abbiamo:

$$\begin{array}{rcll} a \cdot w & = & a \cdot \varepsilon & \\ & = & a & \text{(Def. di concatenazione, clausola base)} \\ & \neq & b & \text{(Ipotesi: } a \neq b\text{)} \\ & = & \varepsilon \cdot b & \text{(Esercizio 5.3.6)} \\ & = & w \cdot b & \end{array}$$

2. [Passo induttivo] Supponiamo che $w = vc$ e assumiamo come ipotesi induttiva che $a \cdot v \neq v \cdot b$. Consideriamo due sottocasi. Se $c = b$, quindi l'ultimo simbolo di w è proprio b, allora

$$\begin{array}{rcll} a \cdot w & = & a \cdot vb & \\ & = & (a \cdot v)b & \text{(Def. di concatenazione, clausola base)} \\ & \neq & (v \cdot b)b & \text{(Ipotesi induttiva)} \\ & = & vbb & \text{(Esercizio 5.3.6)} \\ & = & vb \cdot b & \text{(Esercizio 5.3.6)} \\ & = & w \cdot b & \end{array}$$

   Se invece $w = vc$ e $c \neq b$, allora la tesi $a \cdot w \neq w \cdot b$ è banalmente vera perché le due stringhe terminano con caratteri diversi.

■

I naturali, sintatticamente

Vediamo una definizione induttiva e puramente sintattica dei naturali: chiameremo l'insieme così definito, $s\mathbb{N}$, l'insieme dei *naturali sintattici*.

Definizione 5.3.8 (Definizione induttiva di $\mathbb{N}$)**.** *L'insieme $s\mathbb{N}$ dei* numeri naturali sintattici *è il più piccolo insieme che soddisfa le due clausole seguenti:*

1. $Z \in s\mathbb{N}$.
2. *Se* $x \in s\mathbb{N}$ *allora* $S(x) \in s\mathbb{N}$.

Nella clausola base abbiamo scelto la *costante* Z per ricordare *zero* e nella clausola induttiva abbiamo scelto il *costruttore* $S(\_)$ per ricordare la funzione *successore*. Dalla definizione segue che ogni elemento di $s\mathbb{N}$ avrà la struttura

$$\underbrace{S(S(\cdots(S}_{n\in\mathbb{N}}(Z))\cdots))$$

cioè Z preceduta da un numero finito S. Questo stabilisce una biiezione tra gli insiemi $\mathbb{N}$ e $s\mathbb{N}$.

Esercizio 5.3.9. *Sia* $val : s\mathbb{N} \to \mathbb{N}$ *la funzione definita induttivamente come*

1. $val(Z) = 0$
2. $val(S(x)) = val(x) + 1$

Si dimostri che val *è una biiezione.*

Ricordiamo che nella Sezione 5.1 abbiamo assunto per conosciute le operazioni sui naturali. Mostriamo ora come possiamo definire sui naturali sintattici l'operazione di addizione, in modo induttivo.

Definizione 5.3.10 (Somma di naturali sintattici)**.** *La funzione* $add : s\mathbb{N} \times s\mathbb{N} \to s\mathbb{N}$ *è definita per induzione strutturale sul secondo argomento nel seguente modo:*

1. $add(x, Z) = x$
2. $add(x, S(y)) = S(add(x, y))$

Per esempio, abbiamo che

$$\begin{aligned} add(S(S(S(Z))), S(S(Z))) &= S(add(S(S(S(Z))), S(Z))) && \text{(Clausola induttiva)} \\ &= S(S(add(S(S(S(Z))), Z))) && \text{(Clausola induttiva)} \\ &= S(S(S(S(S(Z))))) && \text{(Clausola base)} \end{aligned}$$

Proposizione 5.3.11. *Per ogni coppia di elementi* $x, y \in s\mathbb{N}$ *vale che* $val(add(x, y)) = val(x) + val(y)$.

Dimostrazione. Procediamo per induzione strutturale sul secondo argomento.

1. $(y = Z)$:

$$\begin{aligned} val(add(x, Z)) &= val(x) && \text{(Per } add\text{, clausola base)} \\ &= val(x) + 0 \\ &= val(x) + val(Z) && \text{(Per } val\text{, clausola base)} \end{aligned}$$

2. $(y = S(z))$:

$$\begin{aligned} val(add(x, S(z))) &= val(S(add(x, z))) && \text{(Per } add\text{, clausola induttiva)} \\ &= val(add(x, z)) + 1 && \text{(Per } val\text{, clausola induttiva)} \\ &= val(x) + val(z) + 1 && \text{(Per ipotesi induttiva)} \\ &= val(x) + val(S(z)) && \text{(Per } val\text{, clausola induttiva)} \end{aligned}$$

■

Esercizio 5.3.12. *Sulla falsariga della Definzione 5.3.10 e sfruttando la funzione add, definire per induzione strutturale le funzioni* $mul : s\mathbb{N} \times s\mathbb{N} \to s\mathbb{N}$ *e* $exp : s\mathbb{N} \times s\mathbb{N} \to s\mathbb{N}$ *in modo tale che* $val(mul(x, y)) = val(x) \cdot val(y)$ *e* $val(exp(x, y)) = val(x)^{val(y)}$

Liste

Le liste sono un tipo di dati di uso comune nei linguaggi di programmazione. Esse consentono la memorizzazione di sequenze di dati di lunghezza variabile (a differenza degli array), normalmente di tipo omogeneo. Vediamo una definizione induttiva delle liste di elementi di un insieme dato A.

Definizione 5.3.13 (Liste)**.** *L'insieme L_A delle liste di elementi di A è il più piccolo insieme che soddisfa:*

1. $[\,] \in L_A$. $[\,]$ *(la* lista vuota*)*
2. *Se $a \in A$ e $lst \in L_A$ allora $a\colon lst \in L_A$*

La costante $[\,]$ è chiamata la *lista vuota* mentre Il simbolo ":" rappresenta l'operazione di aggiungere un elemento all'inizio della lista. Per esempio, se $A = \{a, b, c, d, e\}$ la lista $b\colon e\colon a\colon [\,]$ (che possiamo scrivere in modo equivalente come $(b\colon (e\colon (a\colon [\,])))$) è ottenuta applicando per prima la clausola base, e poi tre volte quella induttiva con, nell'ordine, gli elementi a, e e b. In molti linguaggi di programmazione è possibile scrivere questa lista come $[b, e, a]$, quindi con una sintassi più leggera.

Si noti che nella definizione data la clausola induttiva aggiunge un elemento *all'inizio* della lista, mentre nella Definizione 5.3.1 la clausola induttiva aggiunge un simbolo *alla fine* di una stringa. In entrambi i casi avremmo potuto scegliere diversamente, ma è opportuno sottolineare che normalmente nei linguaggi di programmazione le liste sono definite induttivamente nel modo da noi proposto.

Vediamo alcune funzioni su liste, definite induttivamente. Queste sono operazioni di tipo *strutturale*, nel senso che operano solo sulla struttura della lista indipendentemente dal tipo degli elementi. Assumiamo che gli elementi appartengano a un insieme fissato A, e che a sia un generico elemento di A.

Definizione 5.3.14 (operazioni strutturali su liste)**.** *La* lunghezza *di una lista, $len(lst)$, è definita per induzione strutturale come*

1. $len([\,]) = 0$
2. $len(a\colon lst) = len(lst) + 1$

Queste clausole definiscono univocamente la funzione $len : L_A \to \mathbb{N}$.

La concatenazione *di due liste, $app(lst_1, lst_2)$ (app è abbreviazione di "append"), è definita per induzione strutturale sul primo argomento come*

1. $app([\,], lst_2) = lst_2$
2. $app(a\colon lst_1, lst_2) = a\colon app(lst_1, lst_2)$

Queste clausole definiscono univocamente la funzione $app : L_A \times L_A \to L_A$.

La funzione $rev : L_A \to L_A$ (rev è abbreviazione di "reverse") che applicata a una lista lst restituisce la lista ottenuta invertendo l'ordine degli elementi di lst è definita per induzione strutturale come

1. $rev([\,]) = [\,]$
2. $rev(a\colon lst) = app(rev(lst), a\colon [\,])$

Le definizioni di *len* e di *app* sono abbastanza simili alle funzioni $|\_|$ e $\_\cdot\_$ su stringhe introdotte nella Definizione 5.3.4, ma tengono conto della diversa struttura delle clausole induttive evidenziata sopra. La definizione di *rev* sfrutta *app*: la clausola induttiva può essere letta come "per invertire una lista non vuota, basta togliere il primo elemento, invertire la lista rimanente e inserire l'elemento tolto in fondo". Per esempio, vediamo come si valuta $rev([b, e, a])$:

$$
\begin{array}{lcll}
rev([b,e,a]) & = & rev(b\colon e\colon a\colon [\,]) & \text{(Esplicitiamo la lista)} \\
& = & app(rev(e\colon a\colon [\,]), b\colon [\,]) & \text{(Def. di } rev\text{, clausola induttiva)} \\
& = & app(app(rev(a\colon [\,]), e\colon [\,]), b\colon [\,]) & (rev\text{, clausola induttiva)} \\
& = & app(app(app(rev([\,]), a\colon [\,]), e\colon [\,]), b\colon [\,]) & (rev\text{, clausola induttiva)} \\
& = & app(app(app([\,], a\colon [\,]), e\colon [\,]), b\colon [\,]) & (rev\text{, clausola base)} \\
& = & app(app(a\colon [\,], e\colon [\,]), b\colon [\,]) & (app\text{, clausola base)} \\
& = & app(a\colon (app([\,], e\colon [\,]), b\colon [\,]) & (app\text{, clausola induttiva)} \\
& = & app(a\colon (e\colon [\,]), b\colon [\,]) & (app\text{, clausola base)} \\
& = & a\colon app(e\colon [\,], b\colon [\,]) & (app\text{, clausola induttiva)} \\
& = & a\colon e\colon app([\,], b\colon [\,]) & (app\text{, clausola induttiva)} \\
& = & a\colon e\colon b\colon [\,] & (app\text{, clausola base)} \\
& = & [a,e,b] &
\end{array}
$$

Quindi applicando le definizioni date abbiamo avuto bisogno di circa 10 passi per calcolare l'inverso di una lista di tre elementi. Si può dimostrare che per invertire una lista di n elementi sono necessari un numero di passi dell'ordine di n^2. Anche se la complessità computazionale della valutazione delle funzioni esula dagli argomenti di questo corso, è opportuno presentare una diversa definizione induttiva della funzione $reverse$ che ha un'efficienza maggiore.

Esempio 5.3.15. *La funzione* $rev1 : L_A \times L_A \to L_A$ *è definita per induzione strutturale sul primo argomento come*

1. $rev1([\,], lst) = lst$
2. $rev1(a\colon lst_1, lst_2) = rev1(lst_1, a\colon lst_2)$

Per invertire una lista lst *basta valutare* $rev1(lst, [\,])$*. Per convincerci della correttezza della definizione, descriviamo la valutazione di* $rev1([b,e,a], [\,])$*:*

$$
\begin{array}{lcll}
rev1([b,e,a],[\,]) & = & rev1(b\colon e\colon a\colon [\,],[\,]) & \textit{(Esplicitiamo la lista)} \\
& = & rev1(e\colon a\colon [\,], b\colon [\,]) & \textit{(Clausola induttiva)} \\
& = & rev1(a\colon [\,], e\colon b\colon [\,]) & \textit{(Clausola induttiva)} \\
& = & rev1([\,], a\colon e\colon b\colon [\,]) & \textit{(Clausola induttiva)} \\
& = & a\colon e\colon b\colon [\,] & \textit{(Clausola base)} \\
& = & [a,e,b] &
\end{array}
$$

Quindi abbiamo ottenuto il risultato in 4 passi: per invertire una lista di n *elementi con* $rev1$ *è necessario un numero di passi dell'ordine di* n*.*

Naturalmente possiamo usare il Principo di Induzione Strutturale per dimostrare opportune proprietà sulle liste.

Proposizione 5.3.16 (proprietà delle funzioni su liste)**.** *Le seguenti uguaglianze sono vere per ogni* $lst, lst_1, lst_2, lst_3 \in L_A$*:*

5.3.16.1. $app(lst, [\,]) = lst$

5.3.16.2. $app(app(lst_1, lst_2), lst_3) = app(lst_1, app(lst_2, lst_3))$

5.3.16.3. $rev(app(lst_1, lst_2)) = app(rev(lst_2), rev(lst_1))$

5.3.16.4. $len(app(lst_1, lst_2)) = len(lst_1) + len(lst_2)$

5.3.16.5. $len(rev(lst)) = len(lst))$

Presentiamo la dimostrazione delle prime tre proprietà, lasciando al lettore le ultime due come esercizio.

Dimostrazione di 5.3.16.1. Dimostriamo per induzione strutturale su lst che $app(lst, [\,]) = lst$.

1. [Caso base] ($lst = [\,]$).

$$app([\,],[\,]) \quad = \quad [\,] \quad \text{(Clausola base } app\text{)}$$

2. [Passo induttivo] ($lst = a\colon lst'$). Assumiamo che $app(lst', [\,])) = lst'$, per dimostrare che $app(a\colon lst', [\,]) = a\colon list'$:

$$\begin{aligned} app(a\colon lst', [\,]) &= a\colon app(lst'_1, [\,]) && \text{(Clausola induttiva } app\text{)} \\ &= a\colon lst' && \text{(Ipotesi induttiva)} \end{aligned}$$

■

Dimostrazione di 5.3.16.2. Dimostriamo per induzione strutturale su lst_1 che $app(app(lst_1, lst_2), lst_3) = app(lst_1, app(lst_2, lst_3))$.

1. [Caso base] ($lst_1 = [\,]$).

$$\begin{aligned} app(app([\,], lst_2), lst_3) &= app(lst_2, lst_3) && \text{(Clausola base } app\text{)} \\ &= app([\,], app(lst_2, lst_3)) && \text{(Clausola base } app\text{)} \end{aligned}$$

2. [Passo induttivo] ($lst_1 = a\colon lst'_1$). Assumiamo che $app(app(lst'_1, lst_2), lst_3) = app(lst'_1, app(lst_2, lst_3))$, per dimostrare che $app(app(a\colon lst'_1, lst_2), lst_3) = app(a\colon lst'_1, app(lst_2, lst_3))$:

$$\begin{aligned} app(app(a\colon lst'_1, lst_2), lst_3) &= app(a\colon app(lst'_1, lst_2), lst_3) && \text{(Clausola induttiva } app\text{)} \\ &= a\colon app(app(lst'_1, lst_2), lst_3) && \text{(Clausola induttiva } app\text{)} \\ &= a\colon app(lst'_1, app(lst_2, lst_3)) && \text{(Ipotesi induttiva)} \\ &= app(a\colon lst'_1, app(lst_2, lst_3)) && \text{(Clausola induttiva } app\text{)} \end{aligned}$$

■

Dimostrazione di 5.3.16.3. Dimostriamo per induzione strutturale su lst_1 che $rev(app(lst_1, lst_2)) = app(rev(lst_2), rev(lst_1))$.

1. [Caso base] ($lst_1 = [\,]$).

$$\begin{aligned} rev(app([\,], lst_2)) &= rev(lst_2) && \text{(Clausola base } app\text{)} \\ &= rev(app(lst_2, [\,])) && \text{(Per la (5.3.16.1))} \end{aligned}$$

2. [Passo induttivo] ($lst_1 = a\colon lst'_1$). Assumiamo che $rev(app(lst'_1, lst_2)) = app(rev(lst_2), rev(lst'_1))$, per dimostrare che $rev(app(a\colon lst'_1, lst_2)) = app(rev(lst_2), rev(a\colon lst'_1))$:

$$\begin{aligned} rev(app(a\colon lst'_1, lst_2)) &= rev(a\colon app(lst'_1, lst_2)) && \text{(Clausola induttiva app)} \\ &= app(rev(app(lst'_1, lst_2)), a\colon [\,]) && \text{(Clausola induttiva } rev\text{)} \\ &= app(app(rev(lst_2), rev(lst'_1)), a\colon [\,]) && \text{(Ipotesi induttiva)} \\ &= app(rev(lst_2), app(rev(lst'_1), a\colon [\,])) && \text{(Per la (5.3.16.2))} \\ &= app(rev(lst_2), rev(a\colon lst'_1))) && \text{(Clausola induttiva } rev\text{)} \end{aligned}$$

■

Come detto sopra, le funzioni appena viste manipolano solo la struttura di una lista, non gli elementi. Ma se sull'insieme A sono definite delle operazioni, le possiamo usare per definire altre funzioni interessanti sulle liste in L_A. Vediamo alcuni esempi.

Definizione 5.3.17 (funzioni su liste)**.** *Dato un insieme di simboli A, la funzione $lengthList : L_{A^\star} \to L_{\mathbb{N}}$ è definita per induzione strutturale come*

1. $lengthList([\,]) = [\,]$
2. $lengthList(w\colon lst) = |w|\colon lengthList(lst)$

La somma *di una lista di naturali è la funzione $sumList : L_{\mathbb{N}} \to \mathbb{N}$ definita per induzione strutturale come*

1. $sumList([\,]) = 0$
2. $sumList(n\colon lst) = sumList(lst) + n$

*L'*inserimento ordinato *di un numero naturale in una lista è la funzione $ins : \mathbb{N} \times L_{\mathbb{N}} \to L_{\mathbb{N}}$ definita per induzione strutturale come*

1. $ins(n, [\,]) = n\colon [\,]$
2. $ins(n, m\colon lst) = \begin{cases} m\colon ins(n, lst) & \text{se } n > m \\ n\colon m\colon lst & \text{altrimenti} \end{cases}$

La prima funzione, *lengthList*, valutata su di una lista di stringhe restituisce la lista delle relative lunghezze. Vediamo un esempio di valutazione, assumendo che $A = \{a, b\}$.

$$\begin{aligned} lengthList([abc, \varepsilon]) &= lengthList(abc\colon \varepsilon\colon [\,]) && \text{(Esplicitiamo la lista)} \\ &= |abc|\colon lengthList(\varepsilon\colon [\,]) && \text{(Clausola induttiva)} \\ &= 3\colon |\varepsilon|\colon lengthList([\,]) && \text{(Clausola induttiva, def. di } len) \\ &= 3\colon 0\colon [\,] && \text{(Clausola base, def. di } len) \\ &= [3, 0] \end{aligned}$$

Quindi *lenghts* applica la funzione $|\cdot|$ a tutti gli elementi delle lista passata come argomento, restituendo la lista dei risultati. Molte altre funzioni su liste possono essere definite in modo analogo.

Esercizio 5.3.18.

1. *Definire per induzione strutturale la funzione $sqrList : L_{\mathbb{N}} \to L_{\mathbb{N}}$ che applicata a una lista lst restituisce la lista dei quadrati degli elementi di lst, nello stesso ordine.*
2. *Definire per induzione strutturale la funzione $preds : \mathbb{N} \to L_{\mathbb{N}}$ che per ogni $n \in \mathbb{N}$ restituisce la lista dei suoi predecessori $[n-1, n-2, \ldots, 0]$. Definire quindi la funzione $predsList : L_{\mathbb{N}} \to L_{L_{\mathbb{N}}}$ che applica $preds$ a tutti gli elementi di una lista di naturali, restituendo la lista dei risultati.*

La seconda funzione della Definizione 5.3.17, *sumList*, calcola la somma di tutti i numeri della lista. Vediamo un esempio di valutazione.

$$\begin{aligned} sumList([9, 5, 28]) &= sumList(9\colon 5\colon 28\colon [\,]) && \text{(Esplicitiamo la lista)} \\ &= sumList(5\colon 28\colon [\,]) + 9 && \text{(Clausola induttiva)} \\ &= sumList(28\colon [\,]) + 5 + 9 && \text{(Clausola induttiva)} \\ &= sumList([\,]) + 28 + 14 && \text{(Clausola induttiva)} \\ &= 0 + 42 && \text{(Clausola base)} \\ &= 42 \end{aligned}$$

Anche *sumList* è un esempio particolare di una famiglia di funzioni su liste che possono essere definite secondo lo stesso schema.

Esercizio 5.3.19. *Definire per induzione strutturale le seguenti funzioni:*

1. $maxList : L_N \to \mathbb{N}$*, che restitutisce il massimo di una lista di naturali (*-1 *se la lista è vuota).*

1. $concat : L_{A^\star} \to A^\star$*, che restituisce la concatenanazione di tutte le stringhe della lista data, nell'ordine. Per esempio,* $concat([ba, \varepsilon, aba]) = [baaba]$.

La terza funzione della Definizione 5.3.17, *ins*, assume che la lista di naturali passata come secondo argomento sia ordinata in modo crescente (ogni numero è minore o uguale al successivo), e inserisce nella lista il primo argomento garantendo che la lista risultante sia ancora ordinata. Vediamo un esempio di valutazione.

$$\begin{aligned}
ins(5,[1,3,7]) &= ins(5,1\colon 3\colon 7\colon [\,]) && \text{(Esplicitiamo la lista)}\\
&= 1\colon ins(5,3\colon 7\colon [\,]) && \text{(Clausola induttiva, } 5>1)\\
&= 1\colon 3\colon ins(5,7\colon [\,]) && \text{(Clausola induttiva, } 5>3)\\
&= 1\colon 3\colon 5\colon (7\colon [\,]) && \text{(Clausola induttiva, } 5\leq 7)\\
&= [1,3,5,7]
\end{aligned}$$

Alberi binari

Un altro tipo di dati ampiamente usato in Informatica sono gli *alberi binari*. Secondo la classificazione presentata nella Definizione 4.5.9, si tratta di *alberi radicati cardinali con k=2*, ma ne presentiamo qui una definizione induttiva che caratterizza un insieme di alberi leggermente diverso. Infatti mentre per la definizione vista nel Capitolo 4 un albero ha almeno un nodo, nella nostra definizione un albero può essere anche vuoto. In informatica entrambe queste definizione sono ampiamente usate, anche se in contesti leggermente diversi: la prima nella Teoria dei Grafi, quella che introduciamo ora in Algoritmica.

Definizione 5.3.20 (Alberi binari)**.** *L'insieme BT degli* alberi binari *è il più piccolo insieme che soddisfa:*

1. $\lambda \in BT$*, l'albero vuoto;*

2. *se* $t_1, t_2 \in BT$ *allora* $N(t_1, t_2) \in BT$.

Quindi un albero binario può essere l'albero vuoto, che non contiene nessun nodo, oppure un *nodo* $N(t_1, t_2)$ con due *sottoalberi* t_1 e t_2, chiamati anche rispettivamente il *sottoalbero sinistro* e il *sottoalbero destro*. Un nodo è una *foglia* se entrambi i suoi sottoalberi sono vuoti, altrimenti è un *nodo interno*.[2] Le "relazioni di parentela" tra nodi di un albero binario (figlio, genitore, antenato, discendente) sono definite esattamente come nel Capitolo 4. Nel resto di questa sezione, poiché parleremo solo di alberi *binari*, spesso ometteremo la qualificazione "binario" considerandola sottointesa.

La Figura 5.2 mostra una possibile rappresentazione grafica degli alberi appena definiti. L'albero vuoto (a) non ha una rappresentazione esplicita. La parte centrale, (b), mostra l'albero binario

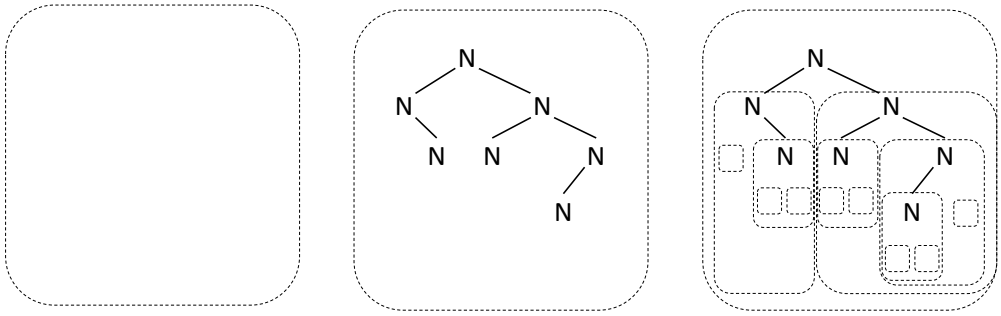
$$T = N(N(\lambda, N(\lambda,\lambda)), N(N(\lambda,\lambda), N(N(\lambda,\lambda),\lambda)))$$

Il nodo più in alto è la *radice*, ci sono 3 foglie e 4 nodi interni, compresa la radice. La parte (c) della figura mostra lo stesso albero in cui abbiamo evidenziato la struttura induttiva, che richiede che ogni nodo N abbia un sottoalbero sinistro e uno destro, eventualmente vuoti.

Questa definizione fornisce la *struttura* di un albero binario, che è sufficiente per definire in modo induttivo due semplici ma importanti funzioni.

Definizione 5.3.21 (funzioni strutturali su alberi binari)**.** *Il numero di nodi di un albero binario, chiamato anche la sua* dimensione*, è definito nel modo seguente:*

[2] Nella Definizione 4.5.2 le foglie sono definite come nodi con grado 1. Di fatto le due definizioni coincidono, tranne che per la radice. Infatti la radice può essere una foglia per la definizione appena data, ma non per quella del Capitolo 4.



(a) (b) (c)

Figura 5.2: (a) L'albero vuoto; (b) Un albero binario; (c) Lo stesso albero in cui è evidenziata la struttura induttiva.

1. $size(\lambda) = 0$.

2. $size(N(t_1, t_2)) = size(t_1) + size(t_2) + 1$

Questo definisce univocamente la funzione size : $BT \to \mathbb{N}$.

La profondità *di un albero binario è definita come*

1. $depth(\lambda) = -1$.

2. $depth(N(t_1, t_2) = max(depth(t_1), depth(t_2)) + 1$

Questo definisce univocamente la funzione depth : $BT \to \mathbb{N} \cup \{-1\}$.

La definizione di *size* è ovvia: un albero vuoto non ha nessun nodo, mentre la dimensione di un albero costituito da un nodo e due sottoalberi è uguale alla somma del numero di nodi di questi ultimi, più uno per la radice.

Invece la profondità di un albero è la lunghezza massima di un cammino dalla radice a una foglia. Se consideriamo un albero del tipo $t = N(t_1, t_2)$, allora ogni cammino dalla radice di t a una foglia sarà composto da (1) un arco dalla radice di t alla radice di t_1 (o t_2), e (2) da un cammino dalla radice di t_1 (o t_2) a una foglia. Questo spiega la clausola induttiva, che dice che profondità di t è uguale al massimo tra le profondità di t_1 e t_2, più uno, per tener conto dell'arco aggiunto all'inizio di tutti i cammini alle foglie. Questa intuizione ci aiuta a spiegare perché la profondità dell'albero vuoto è -1. Infatti consideriamo $t_0 = N(\lambda, \lambda)$, l'albero che ha un solo nodo, sia radice che foglia. Ovviamente la lunghezza del massimo cammino dalla radice a una foglia in t_0 è 0, ma quindi usando la clausola induttiva di *depth* dobbiamo avere $depth(N(\lambda, \lambda)) = max(depth(\lambda), depth(\lambda)) + 1 = depth(\lambda) + 1 = 0$, da cui otteniamo $depth(\lambda) = -1$.

Si noti che per entrambe le funzioni, nelle clausola induttiva la funzione viene invocata due volte, una volta per ogni sottoalbero. Questo fatto è del tutto naturale, vista la struttura della definizione induttiva di questi alberi. Vediamo per esempio la valutazione induttiva di queste due funzioni sull'albero $N(\lambda, N(\lambda, \lambda))$.

$$
\begin{array}{rcll}
size(N(\lambda, N(\lambda, \lambda))) & = & size(\lambda) + size(N(\lambda, \lambda)) + 1 & \text{(Def. di } size\text{, clausola induttiva)} \\
& = & 0 + size(N(\lambda, \lambda)) + 1 & \text{(Def. di } size\text{, clausola base)} \\
& = & 0 + size(\lambda) + size(\lambda) + 1 + 1 & \text{(Def. di } size\text{, clausola induttiva)} \\
& = & 0 + 0 + 2 & \text{(Def. di } size\text{, clausola base)} \\
& = & 2 &
\end{array}
$$

$depth(N(\lambda, N(\lambda, \lambda)))$

$= max(depth(\lambda), depth(N(\lambda, \lambda))) + 1$ (Def. di *depth*, clausola induttiva)

$= max(-1, depth(N(\lambda, \lambda))) + 1$ (Def. di *depth*, clausola base)

$= max(-1, max(depth(\lambda), depth(\lambda)) + 1) + 1$ (Def. di *depth*, clausola induttiva)

$= max(-1, max(-1, -1) + 1) + 1$ (Def. di *depth*, clausola base)

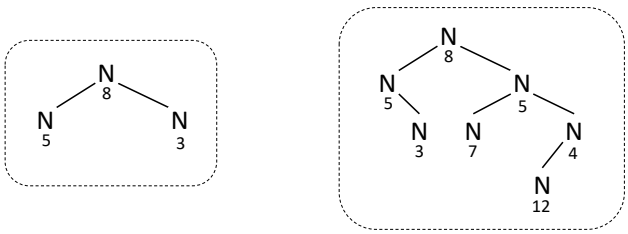
$= max(-1, -1 + 1) + 1 = max(-1, 0) + 1 = 0 + 1 = 1$

Esercizio 5.3.22. *Definire in modo induttivo la funzione* $numLeaves : BT \to \mathbb{N}$ *che applicata a un albero restituisce il numero di foglie.*

Nella pratica siamo interessati a usare gli alberi binari come strutture dati, con la possibilità di memorizzare nelle foglie e/o nei nodi interni dei valori di un certo tipo. Introduciamo quindi una piccola variazione della definizione precedente: gli alberi binari etichettati su un insieme A.

Definizione 5.3.23 (Alberi binari etichettati)**.** *L'insieme* BT_A *degli alberi binari etichettati con elementi di un insieme dato* A *è il più piccolo insieme che soddisfa:*

1. $\lambda \in BT_A$*, l'albero vuoto;*
2. *se* $t_1, t_2 \in BT_A$ *allora* $N(t_1, a, t_2) \in BT_A$ *per ogni* $a \in A$*.*



(a) (b)

Figura 5.3: Due alberi binari etichettati sui naturali.

La Figura 5.3 mostra due alberi binari etichettati sui naturali, quindi elementi di $BT_{\mathbb{N}}$. Ogni etichetta è disegnata sotto il suo nodo. Quindi gli alberi in figura sono rispettivamente $N(N(\lambda, 5, \lambda), 8, N(\lambda, 3, \lambda))$ e

$$N(N(\lambda, 5, N(\lambda, 3, \lambda)), 8, N(N(\lambda, 7, \lambda), 5, N(N(\lambda, 12, \lambda), 4, \lambda)))$$

Molti algoritmi che operano su alberi binari sono basati sul concetto di *visita*, che consiste nel percorrere un cammino nell'albero, visitando sequenzialmente tutti i suoi nodi secondo una opportuna strategia. La visita di un nodo può essere associata a una opportuna azione, come stamparne l'etichetta, applicare all'etichetta una certa funzione, o comporre l'etichetta con altri valori per estrarre dall'albero un risultato. Vediamo una funzione che visita un albero binario etichettato su un insieme A e restituisce la lista delle etichette di tutti i nodi dell'albero, in un certo ordine, utilizzando la funzione *app* della Definizione 5.3.14.

Definizione 5.3.24 (Visita di un albero binario)**.** *La funzione* $visit : BT_A \to L_A$ *è definita come:*

1. $visit(\lambda) = [\,]$.
2. $visit(N(t_1, a, t_2)) = app(visit(t_1), a : visit(t_2))$

Questa funzione visita l'albero in *ordine simmetrico, da sinistra a destra*, perché per ogni nodo prima visita il sottoalbero sinistro, poi il nodo stesso e infine il sottoalbero destro: questo si vede chiaramente dagli argomenti di *app* nella clausola ricorsiva. Altri tipi di visite si possono ottenere cambiando l'ordine di queste tre azioni.

Vediamo per esempio la valutazione della funzione *visit* sull'albero di Figura 5.3 (a).

$$
\begin{aligned}
&visit(N(N(\lambda,5,\lambda),8,N(\lambda,3,\lambda))) \\
&\quad = app(visit(N(\lambda,5,\lambda),8\colon visit(N(\lambda,3,\lambda)))) && (visit\text{, clausola induttiva}) \\
&\quad = app(app(visit(\lambda),5\colon visit(\lambda)),8\colon visit(N(\lambda,3,\lambda)))) && (visit\text{, clausola induttiva}) \\
&\quad = app(app([\,],5\colon [\,]),8\colon visit(N(\lambda,3,\lambda)))) && (visit\text{, clausola base}) \\
&\quad = app(5\colon [\,],8\colon visit(N(\lambda,3,\lambda)))) && (app\text{, clausola base}) \\
&\quad = app(5\colon [\,],8\colon app(visit(\lambda),3\colon visit(\lambda))) && (visit\text{, clausola induttiva}) \\
&\quad = app(5\colon [\,],8\colon app([\,],3\colon [\,])) && (visit\text{, clausola base}) \\
&\quad = app(5\colon [\,],8\colon 3\colon [\,]) && (app\text{, clausola base}) \\
&\quad = 5\colon app([\,],8\colon 3\colon [\,]) && (app\text{, clausola induttiva}) \\
&\quad = 5\colon 8\colon 3\colon [\,] && (app\text{, clausola base}) \\
&\quad = [5,8,3]
\end{aligned}
$$

Il lettore è invitato a trovare l'ordine in cui vengono restituite le etichette dall'applicazione di *visit* all'albero di Figura 5.3 (b), e a confrontarlo con quello nella nota a pié di pagina. Valutare *visit* usando la definizione formale induttiva è molto laborioso, come appena visto, ma il risultato si può ottenere abbastanza rapidamente sfruttando la descrizione intuitiva della visita descritta sopra.[3]

Gli alberi binari sono spesso utilizzati per memorizzare insiemi di elementi. Dato un generico elemento a in A, uno è interessato a verificare se l'elemento appartiene o meno all'insieme rappresentato dall'albero. La seguente funzione prende in input un elemento di A ed un albero su A e restituisce come output `t` se l'elemento appartiene all'albero e `f` altrimenti.

Esempio 5.3.25. *La funzione belong* : $A \times BT_{\mathbb{A}} \to Bool$ *è definita come*

1. $belong(a,\lambda) = \texttt{f}$.
2. $belong(a,N(t_1,b,t_2)) = \begin{cases} \texttt{t} & \text{if } b{=}a \\ belong(a,t_1) \vee belong(a,t_2) & otherwise \end{cases}$

Come per le liste, anche sugli alberi etichettati si possono definire delle funzioni che utilizzano le operazioni definite sulle etichette. Le due funzioni che presentiamo ora, $lenghtsBT$ e $sumBT$ sono del tutto analoghe a quelle presentate nella Definizione5.3.17.

Definizione 5.3.26 (funzioni su alberi binari)**.** *Dato un insieme di simboli A, la funzione $lengthBT$: $BT_{A^\star} \to BT_{\mathbb{N}}$ è definita per induzione strutturale come*

1. $lengthBT(\lambda) = \lambda$
2. $lengthBT(N(t_1,w,t_2)) = N(lengthBT(t_1),|w|,lengthBT(t_2))$

La somma *di un albero binario di naturali è la funzione $sumBT : BT_{\mathbb{N}} \to \mathbb{N}$ definita per induzione strutturale come*

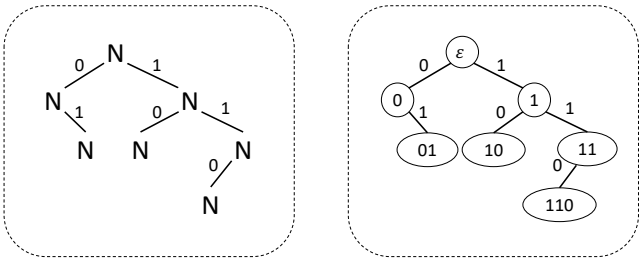
1. $sumBT(\lambda) = 0$
2. $sumBT(N(t_1,n,t_2)) = sumBT(t_1) + sumBT(t_2) + n$

[3] Il risultato è la lista $[5,3,8,7,5,12,4]$.

Ma gli alberi sono alberi?

La definizione di albero binario della Definizione 5.3.20 sembra molto diversa dalle varie definizioni di albero viste nella Sezione 4.5. La rappresentazione grafica di Figura 5.2 (b) ci convince (abbastanza) che un albero binario come definito sopra corrisponde a un albero orientato, radicato, cardinale per k=2, secondo la Definizione 4.5.9. Ma come possiamo "collegare" le due definizioni?

Per rispondere a questa legittima domanda, mostriamo come si può definire usando l'induzione strutturale una funzione che associa a ogni albero binario in $T \in BT$ un grafo orientato $G_T = (V_T, E_T)$ che lo rappresenta fedelmente. Consieriamo l'albero T mostrato nella Figura 5.2 (b). Il problema principale è determinare l'insieme di nodi V_T. Usiamo la seguente tecnica: etichettiamo tutti gli archi che collegano un nodo alla radice del sottoalbero sinistro con 0, e tutti gli altri con 1, come in Figura 5.4 (a). Si vede facilmente che ogni cammino dalla radice a un nodo è individuato univocamente da una stringa in $\{0,1\}$, e poichè tra ogni coppia di nodi c'è un solo cammino, usiamo questa stringa come "identità" del nodo. Quindi scegliamo l'insieme V_T come un sottoinsieme di $\{0,1\}^\star$, come mostrato nella Figura 5.4 (b).



(a) (b)

Figura 5.4: Identificazione dell'insieme di nodi di un albero $T \in BT$

La seguente definizione per induzione strutturale definisce sia l'insieme di nodi $V_T \subseteq \{0,1\}^\star$ che l'insieme di archi $E_T \subseteq V_T \times V_T$ del grafo G_T realizzando la costruzione descritta sopra.

1. [Clausola base] $(T = \lambda)$
$$grafo(\lambda) = (\varnothing, \varnothing)$$

2. [Clausola induttiva $(T = N(t_1, t_2))$ Supponiamo per induzione strutturale che $grafo(t_1) = (V_1, E_1)$ e $grafo(t_2) = (V_2, E_2)$. Allora definiamo

$$V = \{0w \mid w \in V_1\} \cup \{1w \mid w \in V_2\} \cup \{\varepsilon\}$$
$$E = \{(0w, 0v) \mid (w,v) \in E_1\} \cup \{(1w, 1v) \mid (w,v) \in E_2\} \cup \{(\varepsilon, 0) \mid \varepsilon \in V_1\} \cup \{(\varepsilon, 1) \mid \varepsilon \in V_2\}$$
$$grafo(N(t_1, t_2)) = (V, E)$$

Espressioni

La valutazione delle espressioni è una caratteristica fondamentale dei linguaggi di programmazione, e il riconoscimento delle espressioni come tipo di dati induttivo o ricorsivo è fondamentale per capire come possono essere elaborate. Per illustrare questo approccio lavoreremo con un esempio giocattolo: espressioni aritmetiche come $3 \cdot x^2 + 2 \cdot x + 1$ che coinvolgono solo una variabile, "x". Faremo riferimento all'insieme (tipo di dati) di tali espressioni come Exp. Ecco la sua definizione:

Definizione 5.3.27 (espressioni)**.** *L'insieme Exp è definito induttivamente nel modo seguente:*

1. [Clausole base]

- `x` $\in Exp$, *dove x è una variabile*
- `n` $\in Exp$ *per ogni* $n \in \mathbb{N}$*; quindi* $\mathbb{N} \subseteq Exp$.

2. [Clausole induttive] *Se* $\mathtt{e}, \mathtt{f} \in Exp$ *allora*

- $\langle \mathtt{e} + \mathtt{f} \rangle \in Exp$. *L'espressione* $\langle \mathtt{e} + \mathtt{f} \rangle$ *è chiamata una* somma, $\mathtt{e}$ *ed* $\mathtt{f}$ *sono chiamati i* componenti *della somma o* addendi.
- $\langle \mathtt{e} * \mathtt{f} \rangle \in Exp$. *L'espressione* $\langle \mathtt{e} * \mathtt{f} \rangle$ *è chiamata un* prodotto. $\mathtt{e}$ *ed* $\mathtt{f}$ *sono* componenti *del prodotto; sono anche chiamati* moltiplicatore *e* moltiplicando.
- $-\langle \mathtt{e} \rangle \in Exp$.

Si noti che le espressioni in Exp sono completamente tra parentesi, e che gli esponenti non sono consentiti. Quindi l'espansione dell'espressione polinomiale $3 \cdot x^2 + 2 \cdot x + 1$ dovrebbe essere scritta formalmente come:

$$\langle\langle 3 * \langle \mathtt{x} * \mathtt{x} \rangle\rangle + \langle\langle 2 * \mathtt{x} \rangle + 1\rangle\rangle \tag{5.6}$$

Queste parentesi e le occorrenze di $*$ rendono gli esempi più pesanti da leggere, quindi useremo spesso espressioni più semplici come "$3 \cdot x^2 + 2 \cdot x + 1$" invece dell'espressione (5.6). Ma è importante riconoscere che $3 \cdot x^2 + 2 \cdot x + 1$ non è una Exp; è un'abbreviazione di una Exp.

Valutazione di espressioni

Poiché l'unica variabile in una Exp è x, il valore di una Exp è determinato univocamente dal valore di x. Per esempio, se il valore di x è 3, allora il valore di $3 \cdot x^2 + 2 \cdot x + 1$ è 34. In generale, data una qualsiasi espressione $e \in Exp$ e un valore intero n per la variabile x, possiamo valutare e per trovare il suo valore, $eval(e, n)$. È facile e utile specificare questo processo di valutazione con una definizione induttiva.

Definizione 5.3.28 (valutazione di espressioni)**.** *La* funzione di valutazione, $eval : Exp \times \mathbb{Z} \to \mathbb{Z}$ *è definita induttivamente sulle espressioni,* $e \in Exp$ *come segue. Sia* n *qualsiasi numero intero.*

1. [Clausole base]

$$eval(\mathtt{x}, n) = n; \qquad \textit{(il valore della variabile } x \textit{ è } n.\textit{)} \tag{5.7}$$
$$eval(\mathtt{k}, n) = k; \qquad \textit{(il valore del numero } k \textit{ è } k\textit{, indipendentemente da } x.\textit{)} \tag{5.8}$$

2. [Clausole induttive] *Se* $e, f \in Exp$ *allora*

$$eval(\langle \mathtt{e}_1 + \mathtt{e}_2 \rangle, n) = eval(\mathtt{e}_1, n) + eval(\mathtt{e}_2, n) \tag{5.9}$$
$$eval(\langle \mathtt{e}_1 * \mathtt{e}_2 \rangle, n) = eval(\mathtt{e}_1, n) \cdot eval(\mathtt{e}_2, n) \tag{5.10}$$
$$eval(-\langle \mathtt{e} \rangle, n) = -eval(\mathtt{e}, n) \tag{5.11}$$

Per esempio, ecco come la definizione induttiva di $eval$ arriverebbe al valore di $3 + x^2$ quando x vale 2:

$$\begin{aligned}
eval(\langle 3 + \langle \mathtt{x} * \mathtt{x} \rangle\rangle, n) &= eval(3, 2) + eval(\langle \mathtt{x} * \mathtt{x} \rangle, 2) && \text{(per (5.9))} \\
&= 3 + eval(\langle \mathtt{x} * \mathtt{x} \rangle, 2) && \text{(per (5.8))} \\
&= 3 + (eval(\mathtt{x}, 2) \cdot eval(\mathtt{x}, 2)) && \text{(per (5.10))} \\
&= 3 + (2 \cdot 2) && \text{(per (5.7))} \\
&= 3 + 4 \\
&= 7
\end{aligned}$$

Sostituzione di variabili con espressioni

La sostituzione di variabili con espressioni è un'operazione standard utilizzata dai compilatori e nel contesto dei sistemi algebrici. Per esempio sostituendo la variabile x con l'espressione $3 \cdot x$ nell'espressione $x \cdot (x - 1)$ si ottiene $3 \cdot x \cdot (3 \cdot x - 1)$. Useremo la notazione generale $subst(f, e)$ per

indicarte il risultato della sostituzione di ciascuna occorrenza della variabile x nell'espressione e con l'espressione f. Quindi, come abbiamo appena spiegato,

$$subst(3 \cdot x, x \cdot (x-1)) = 3 \cdot x \cdot (3 \cdot x - 1)$$

Questa funzione di sostituzione ha una semplice definizione induttiva.

Definizione 5.3.29 (sostituzione)**.** *La funzione di sostituzione* $subst : Exp \times Exp \to Exp$ *è definita induttivamente sul secondo argomento* $\mathtt{e} \in Exp$*, per qualunque* $\mathtt{f} \in Exp$*, come segue.*

1. [Clausole base]

$$subst(\mathtt{f}, \mathtt{x}) = \mathtt{f}; \qquad \textit{(sostituendo la variabole } \mathtt{x} \textit{ con } \mathtt{f} \textit{ si ottiene solo } \mathtt{f}\textit{)} \tag{5.12}$$
$$subst(\mathtt{f}, \mathtt{k}) = \mathtt{k}; \qquad \textit{(la sostituzione di un numero non fa nulla)} \tag{5.13}$$

2. [Clausole induttive] *Se* $\mathtt{e}, \mathtt{f} \in Exp$ *allora*

$$subst(\mathtt{f}, \langle \mathtt{e}_1 + \mathtt{e}_2 \rangle) = \langle subst(\mathtt{f}, \mathtt{e}_1) + subst(\mathtt{f}, \mathtt{e}_2) \rangle \tag{5.14}$$
$$subst(\mathtt{f}, \langle \mathtt{e}_1 * \mathtt{e}_2 \rangle) = \langle subst(\mathtt{f}, \mathtt{e}_1) * subst(\mathtt{f}, \mathtt{e}_2) \rangle \tag{5.15}$$
$$subst(\mathtt{f}, -\langle \mathtt{e} \rangle) = -\langle subst(\mathtt{f}, \mathtt{e}) \rangle \tag{5.16}$$

Ecco come la definizione ricorsiva della funzione di sostituzione troverebbe il risultato della sostituzione di x con $3 \cdot x$ in $x \cdot (x-1)$:

$$\begin{aligned}
&subst(3 \cdot x, x \cdot (x-1)) \\
&\quad = subst(\langle \mathtt{3} * \mathtt{x} \rangle, \langle \mathtt{x} * \langle \mathtt{x} + -\langle \mathtt{1} \rangle \rangle \rangle) && \text{(Espandiamo la sintassi semplificata)} \\
&\quad = \langle subst(\langle \mathtt{3} * \mathtt{x} \rangle, \mathtt{x}) * subst(\langle \mathtt{3} * \mathtt{x} \rangle, \langle \mathtt{x} + -\langle \mathtt{1} \rangle \rangle) \rangle && \text{(per (5.15))} \\
&\quad = \langle \langle \mathtt{3} * \mathtt{x} \rangle * subst(\langle \mathtt{3} * \mathtt{x} \rangle, \langle \mathtt{x} + -\langle \mathtt{1} \rangle \rangle) \rangle && \text{(per (5.12))} \\
&\quad = \langle \langle \mathtt{3} * \mathtt{x} \rangle * \langle subst(\langle \mathtt{3} * \mathtt{x} \rangle, \mathtt{x}) + subst(\langle \mathtt{3} * \mathtt{x} \rangle, -\langle \mathtt{1} \rangle) \rangle \rangle && \text{(per (5.14))} \\
&\quad = \langle \langle \mathtt{3} * \mathtt{x} \rangle * \langle \langle \mathtt{3} * \mathtt{x} \rangle + -\langle subst(\langle \mathtt{3} * \mathtt{x} \rangle, \mathtt{1}) \rangle \rangle \rangle && \text{(per (5.12) e (5.16))} \\
&\quad = \langle \langle \mathtt{3} * \mathtt{x} \rangle * \langle \langle \mathtt{3} * \mathtt{x} \rangle + -\langle \mathtt{1} \rangle \rangle \rangle && \text{(per (5.13))} \\
&\quad = 3 \cdot x \cdot (3 \cdot x - 1) && \text{(Sintassi semplificata)}
\end{aligned}$$

Supponiamo ora di dover trovare il valore di $subst(3 \cdot x, x \cdot (x-1))$ quando $x = 2$. Ci sono due approcci. Per prima cosa, potremmo effettivamente fare la sostituzione sopra indicata per ottenere $3 \cdot x \cdot (3 \cdot x - 1)$, e quindi valutare questa espressione per $x = 2$, cioè potremmo calcolare ricorsivamente $eval(3 \cdot x \cdot (3 \cdot x - 1), 2)$ per ottenere il valore finale 30. Questo approccio è descritto dall'espressione

$$eval(subst(3 \cdot x, x \cdot (x-1)), 2) \tag{5.17}$$

Nel gergo della programmazione, questa è chiamata una valutazione secondo il *Modello di Sostituzione*. Si noti che con questo approccio la formula $3 \cdot x$ appare due volte dopo la sostituzione, quindi la moltiplicazione $3 \cdot 2$ che ne calcola il valore viene eseguita due volte.

L'altro approccio è chiamato valutazione secondo il *Modello di Ambiente*. Vale a dire, per calcolare il valore della (5.17) valutiamo prima $3 \cdot x$ per il valore $x = 2$ ottenendo 6. Quindi valutiamo $x \cdot (x-1)$ per $x = 6$ ottenendo il risultato di 30. Questo approccio è descritto dall'espressione

$$eval(x \cdot (x-1), eval(3 \cdot x, 2)) \tag{5.18}$$

Usando il Modello di Ambiente si calcola il valore di $3 \cdot x$ una sola volta, quindi si richiede una moltiplicazione in meno rispetto al Modello di Sostituzione per calcolare la (5.18).

Esercizio 5.3.30. *Si scriva esplicitamente la valutazione della* (5.17) *sia usando il Modello di Sostituzione che quello di Ambiente, indicando le regole applicate ad ogni passaggio. Confrontare il numero di operazioni aritmetiche e di lettura di variabili nei due casi.*

Ma come sappiamo che i valori finali raggiunti da questi due approcci, cioè i valori interi finali di (5.17) e (5.18), coincidono? In effetti, possiamo dimostrare abbastanza facilmente che questi due approcci producono sempre lo stesso risultato, sfruttando l'induzione strutturale sulle definizioni dei due approcci.

Teorema 5.3.31. *Per tutte le espressioni $e, f \in Exp$ e per tutti i valori $n \in \mathbb{Z}$,*

$$eval(subst(\mathtt{f}, \mathtt{e}), n) = eval(\mathtt{e}, eval(\mathtt{f}, n)) \tag{5.19}$$

Dimostrazione. La dimostrazione è per induzione strutturale su `e`.

1. *[Casi base]*

   - Caso "`x`": Il lato sinistro dell'equazione (5.19) è uguale a $eval(\mathtt{f}, n)$ per il caso base della Definizione 5.3.29 (funzione di sostituzione), e anche il lato destro è uguale a $eval(\mathtt{f}, n)$ per il caso base della Definizione 5.3.28 (eval).
   - Caso "`k`": Il lato sinistro dell'equazione (5.19) è uguale a k per il caso base delle Definizioni 5.3.29 e 5.3.28. Ma anche il lato destro è uguale a k per due applicazioni del caso base della Definizione 5.3.28 .

2. *[Passi induttivi]*

   - Caso "$\langle \mathtt{e}_1 + \mathtt{e}_2 \rangle$": Per l'ipotesi induttiva (5.19) possiamo assumere che per qualunque $\mathtt{f} \in Exp$ e per $i \in \{1, 2\}$ valga

     $$eval(subst(\mathtt{f}, \mathtt{e}_i), n) = eval(\mathtt{e}_i, eval(\mathtt{f}, n)) \tag{5.20}$$

     Vogliamo dimostrare che

     $$eval(subst(\mathtt{f}, \langle \mathtt{e}_1 + \mathtt{e}_2 \rangle), n) = eval(\langle \mathtt{e}_1 + \mathtt{e}_2 \rangle, eval(\mathtt{f}, n)) \tag{5.21}$$

     La parte sinistra della (5.21) è uguale a

     $$eval(\langle subst(\mathtt{f}, \mathtt{e}_1) + subst(\mathtt{f}, \mathtt{e}_2) \rangle, n)$$

     per la definizione di sostituzione in una somma ((5.14) nella Definizione 5.3.29). Ma questo è uguale a

     $$eval(subst(\mathtt{f}, \mathtt{e}_1), n) + eval(subst(\mathtt{f}, \mathtt{e}_2), n)$$

     per la definizione di valutazione di una somma ((5.9) nella Definizione 5.3.28. Per ipotesi induttiva (5.20) questo a sua volta è uguale a

     $$eval(\mathtt{e}_1, eval(\mathtt{f}, n)) + eval(\mathtt{e}_2, eval(\mathtt{f}, n))$$

     Infine, quest'ultima espressione è uguale al lato destro della (5.21) per la definizione della valutazione di una somma ((5.9) nella Definizione 5.3.28. Ciò conclude la dimostrazione della (5.21) .
   - Caso "$\langle \mathtt{e1} * \mathtt{e2} \rangle$": Questo caso è del tutto simile al precedente.
   - Caso "$-\langle \mathtt{e} \rangle$": Questo caso è ancora più semplice.

■

5.4 Funzioni ricorsive

In ognuna delle funzioni introdotte nelle sezioni precedenti il valore della funzione per un certo argomento era definito o direttamente (nella clausola base), oppure in termini del valore della stessa funzione su argomenti "più piccoli". In particolare per le funzioni definite su $\mathbb{N}$ il valore della funzione su 0 era definito direttamente perchè non c'è nessun numero naturale $k < 0$.

Le funzioni definite induttivamente sono un caso particolare di FUNZIONI RICORSIVE. Una funzione è detta *ricorsiva* se il valore della funzione per un certo argomento è espresso in termini del valore della stessa funzione applicata a uno o più argomenti, non necessariamente più piccoli.

Per esempio, consideriamo la seguente funzione *due*:

$$due(n) = \begin{cases} 2 & \text{se } n > 100 \\ due(n+1) & \text{altrimenti.} \end{cases} \tag{5.22}$$

Si tratta di una definizione ricorsiva, ma non induttiva, perché nella seconda clausola il valore per n è espresso in termini del valore di *due* per un argomento maggiore, $n+1$. Per ottenere il valore della funzione per un certo argomento usiamo la solita tecnica: applichiamo la clausola che corrisponde all'argomento e otteniamo un valore finale, oppure continuiamo con la valutazione dell'espressione ottenuta. Valutiamo la funzione *due* per un paio di argomenti: 98 e 110.

$$\begin{aligned} due(110) &= 2 && \text{(Clausola base)} \\ \\ due(98) &= due(99) && \text{(Clausola induttiva)} \\ &= due(100) && \text{(Clausola induttiva)} \\ &= due(101) && \text{(Clausola induttiva)} \\ &= 2 && \text{(Clausola base)} \end{aligned}$$

Si verifica facilmente che valutando $due(k)$ per un qualunque $k \in \mathbb{N}$ il risultato finale è 2. Quindi la (5.22) definisce correttamente una funzione $due : \mathbb{N} \to \mathbb{N}$.

Vediamo un altro esempio di funzione ricorsiva ma non induttiva.

$$f(n) = \begin{cases} 1 & \text{se } n \leq 1 \\ f(n/2) & \text{se } n > 1 \text{ ed è pari} \\ f(3 \cdot n + 1) & \text{se } n > 1 \text{ ed è dispari} \end{cases} \tag{5.23}$$

Valutiamo questa funzione per i primi numeri naturali, da 0 a 8:

$$\begin{aligned} &f(0) = 1 \\ &f(1) = 1 \\ &f(2) = f(1) = 1 \\ &f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1 \\ &f(4) = f(2) = f(1) = 1 \\ &f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1 \\ &f(6) = f(3) = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1 \\ &f(7) = f(22) = f(11) = f(34) = f(17) = f(52) = f(26) = f(13) = f(40) = f(20) \\ &\quad = f(10) = f(5) = f(16) = f(8) = f(4) = f(2) = f(1) = 1 \\ &f(8) = f(4) = f(2) = f(1) = 1 \end{aligned}$$

È facile convincersi che valutando la funzione per un certo $k \in \mathbb{N}$, se la valutazione termina allora il valore finale deve essere 1. Ma il numero di passi necessario per arrivare al risultato non sembra seguire una regola precisa (si veda $f(7)$). Viene naturale chiedersi: ma è sicuro che la valutazione di $f(k)$ termina per ogni k? Torneremo a discutere questa funzione più avanti nell'Esempio 5.4.10, quando parleremo della *Congettura di Collatz*. Per il momento cerchiamo di affrontare questo problema, cioè se la valutazione di una funzione ricorsiva termina, da una prospettiva più ampia.

Nelle definizioni induttive di funzioni delle sezioni precedenti, la clausola base e la clausola induttiva corrispondevano direttamente alle clausole della definizione induttiva dell'insieme su cui la funzione era definita. Questa tecnica garantisce che la funzione sia ben definita. Anche la funzione *due* vista sopra è ben definita, ma è facile convincersi che una descrizione ricorsiva e non

induttiva potrebbe non essere accettabile, nel senso che potrebbe non definire in modo univoco una funzione, oppure potrebbe non esserci alcuna funzione che rispetti tale descrizione. Esploriamo queste problematiche con alcuni esempi di definizioni ricorsive.

$$g(n) = \begin{cases} 0 & \text{se } n = 0 \\ g(n+1) & \text{altrimenti.} \end{cases} \tag{5.24}$$

Questa definizione ha una clausola base, ma la seconda definisce g per gli altri valori in $\mathbb{N}$ in termini del suo valore per un argomento più grande. Un possibile modo di interpretare una definizione ricorsiva come questa è quello *dichiarativo*, che consiste nel caratterizzare un insieme di funzioni, cioè tutte quelle che soddisfano entrambe le clausole. Da questo punto di vista è facile convincersi che ogni funzione che valga 0 su 0 e che sia costante per tutti gli altri naturali soddisferebbe la (5.24).

Ma questo non è il modo in cui noi informatici vediamo le definizioni ricorsive: noi le interpretiamo in modo *operazionale* o *computazionale*, pensando a come un programma in un tipico linguaggio di programmazione valuterebbe g su un dato argomento. Quindi possiamo pensare che, fissati gli insiemi di partenza e di arrivo (supponiamo che siano entrambi $\mathbb{N}$, per essere concreti), una definizione ricorsiva di, per esempio, *rec*, caratterizzi in realtà una relazione R_{rec} su $\mathbb{N}$ così definita:

$$R_{rec} = \left\{ (n, m) \quad | \quad \begin{array}{l} \text{partendo da } rec(n) \text{ e applicando iterativamente (e correttamente)} \\ \text{le clausole della definizione, la valutazione termina con il risultato di } m \end{array} \right\}$$

Poiché le funzioni sono relazioni totali e univalenti, diremo che una definizione ricorsiva per *rec* è *ben data* se R_{rec} è totale e univalente. Nel caso della (5.24), fissando il dominio come $\mathbb{N}$ abbiamo per esempio che per valutare $g(1)$ applicheremmo la seconda clausola, valutando poi $g(2)$, e poi $g(3)$, e così via senza mai terminare. Quindi la relazione R_g non è totale perché non contiene alcuna coppia del tipo $(1, \_)$. Di conseguenza, per l'interpretazione operazionale che noi adottiamo la (5.24) non è ben data: essa non definisce correttamente una funzione, ma solo una funzione parziale (indefinita su tutti i naturali tranne che su 0).

$$f(n) = 1 + f(n-1) \tag{5.25}$$

Possiamo considerare questa come una definizione induttiva in cui manca la clausola base. Se consideriamo $\mathbb{N}$ come dominio di f, vediamo subito che la definizione non è ben data, perchè $f(0) = f(-1)$ e $-1 \notin \mathbb{N}$. Se invece consideriamo $\mathbb{Z}$ come dominio e proviamo a valutarla per un qualunque argomento $k \in \mathbb{Z}$ abbiamo

$$f(k) = 1 + f(k-1) = 2 + f(k-2) = 3 + f(k-3) = \ldots$$

Quindi la valutazione non termina mai, mostrando che la (5.25) non definisce correttamente una funzione.

$$h(n) = \begin{cases} h(n-2) & \text{se } n \geq 2 \\ h(n+2) & \text{se } n = 1 \\ 0 & \text{se } n = 0 \end{cases} \tag{5.26}$$

Questa definizione è per certi aspetti simile alla (5.24): dichiarativamente, ogni funzione sui naturali che valga 0 sui numeri pari e una costante arbitraria k sui numeri dispari soddisferebbe le tre clausole. Ma operazionalmente se si valuta h su un numero dispari $k \geq 3$ dopo un certo numero di passi si arriva a valutare $h(3)$ e quindi $h(1)$, $h(3)$, $h(1)$,... Quindi la (5.26) non definisce correttamente una funzione.

Viene quindi naturale chiedersi: come si può verificare se la definizione ricorsiva di una funzione è ben data? Un importante risultato della Teoria della Calcolabilità, il Teorema di Rice, ci dice che non esiste un procedimento effettivo che, data una qualunque definizione ricorsiva, ci consenta di concludere se essa caratterizza un funzione totale oppure no. Però si possono identificare delle condizioni sufficienti, nel senso che se una definizione ricorsiva le soddisfa allora è ben data. Per

esempio, tutte le definizioni induttive viste nelle sezioni precedenti sono ben date. Inoltre, per quanto riguarda l'univalenza, è ovvio che se per ogni valore del dominio solo una delle clausole della definizione è applicabile, allora la relazione indotta dalla definizione ricorsiva è univalente: questa è una condizione sufficiente facilmente verificabile con una semplice analisi della definizione.[4]

Approfondiamo invece il problema di verificare se la relazione indotta da una definizione ricorsiva è totale. Supponiamo di avere una definizione ricorsiva di rec su un dominio A e assumiamo che per ogni elemento $a \in A$ ci sia almeno una clausola applicabile (altrimenti possiamo già concludere che R_{rec} non è totale). In questa situazione se R_{rec} non è totale (cioè non c'è nessuna coppia del tipo $(a, \_)$ in R_{rec} per un certo $a \in A$) allora vuol dire che valutando $rec(a)$ incorriamo in una computazione infinita. Vediamo come formalizzare delle condizioni che garantiscono che questo non possa mai succedere, e che quindi la definzione di rec sia ben data.

Definizione 5.4.1 (Relazione di precedenza indotta da una definizione ricorsiva)**.** *Data una definizione ricorsiva di* $rec : A \to B$*, la* relazione di precedenza indotta *è la relazione* $\prec_{rec} \subseteq A \times A$ *definita come segue:*

$$\textit{per ogni } x, y \in A,\ x \prec_{rec} y \textit{ se e solo se } f(y) \textit{ è definita direttamente in termini di } f(x)$$

Si noti che la relazione $\prec$ così definita non è necessariamente aciclica, anche se il simbolo utilizzato potrebbe suggerirlo.

Esempio 5.4.2. *Per le definizioni ricorsive presentate sopra, le relazioni di precedenza da esse indotte sono le seguenti:*

(5.22) $\prec_{due} \subseteq \mathbb{N} \times \mathbb{N}$*, definita come* $n + 1 \prec_{due} n$ *se* $n \leq 100$

(5.24) $\prec_g \subseteq \mathbb{N} \times \mathbb{N}$*, definita come* $n + 1 \prec_g n$ *se* $n > 0$

(5.25) $\prec_f \subseteq \mathbb{Z} \times \mathbb{Z}$*, definita come* $n - 1 \prec_f n$ *se* $n \in \mathbb{Z}$

(5.26) $\prec_h \subseteq \mathbb{N} \times \mathbb{N}$*, definita come* $\prec_h = \{(n-2, n) \mid n \geq 2\} \cup \{(3, 1)\}$

Quindi se $b \prec_{rec} a$, per valutare $rec(a)$ devo necessariamente valutare $rec(b)$. Ma allora se c'è una sequenza infinita di elementi di A tali che[5]

$$a_1 \succ a_2 \succ a_3 \succ \dots$$

allora la valutazione di $rec(a_1)$ non termina mai. Questo vale per esempio per le relazioni $\prec_g$ e $\prec_f$ dell'Esempio 5.4.2, per le quali abbiamo, per esempio

$$1 \succ_f 0 \succ_f -1 \succ_f -2 \dots$$

$$3 \succ_g 4 \succ_g 5 \succ_g 6 \dots$$

ma anche per $\prec_h$, per la quale abbiamo

$$7 \succ_h 5 \succ_h 3 \succ_h 1 \succ_h 3 \succ_h 1 \dots$$

Per concludere formalmente questa discussione con una condizione che garantisce che una definizione ricorsiva sia ben data, introduciamo la seguente definizione.

Definizione 5.4.3 (Relazione ben fondata)**.** *Una relazione* $\sqsubset$ *su un insieme* A*,* $\sqsubset \subseteq A \times A$*, è* BEN FONDATA *se non esiste una catena infinita decrescente*

$$a_1 \sqsupset a_2 \sqsupset a_3 \sqsupset \dots$$

di elementi $a_i \in A$*.*

Dalle considerazioni precedenti segue il seguente risultato, che fornisce una condizione sufficiente affinché la definizione ricorsiva di una funzione sia ben data.

[4] Esercizio: perchè questa condizione non è necessaria, ma solo sufficiente?
[5] Abbastanza ovviamente, scriviamo $\succ$ per $\prec^{op}$

Proposizione 5.4.4. *Data una definizione ricorsiva di* $rec : A \to B$*, essa è ben data (e quindi* rec *è totale) se (1) per ogni elemento* $a \in A$ *c'è esattamente una clausola della definizione applicabile per valutare* $rec(a)$ *e (2) la relazione* $\prec_{rec}$ *è ben fondata.*

Esempio 5.4.5. *La* (5.22) *induce un ordinamento ben fondato: esso ha una sola catena discendente finita,* $0 \succ_{due} 1 \succ_{due} \ldots \succ_{due} 99 \succ_{due} 100 \succ_{due} 101$*, mentre tutti gli altri elmenti elementi di* $\mathbb{N}$ *non sono in relazione due tra loro.*

Abbiamo visto che le definizioni induttive sono un caso particolare di ricorsione, e sono sempre ben date. Vediamo perché, utilizzando il concetto di relazione ben fondata.

Definizione 5.4.6 (Relazione di precedenza indotta da definizione induttiva di un insieme)**.** *Data una definizione induttiva di un insieme* A*, la* relazione di precedenza indotta *è la relazione* $\prec_A \subseteq A \times A$ *definita come segue:*

> *se per la clausola induttiva l'elemento* a *appartiene all'insieme perché ci appartengono* $a_1, \ldots, a_k$*, allora* $a_1 \prec_A a, a_2 \prec_A a, \ldots, a_k \prec_A a$*.*

Proposizione 5.4.7 (Relazione indotta da definizione induttiva è ben fondata)**.** *La relazione* $\prec_A \subseteq A \times A$ *della Definizione 5.4.6 è ben fondata.*

Infatti poichè l'insieme caratterizzato è il più piccolo che soddisfa le clausole base e induttiva, abbiamo che ogni elemento dell'insieme ci appartiene per una sequenza finita di applicazioni della clausola induttiva, e questo garantisce che la relazione $\prec_A$ *sia ben fondata.*

Esempio 5.4.8 (Relazioni di precedenza)**.** *La Definizione 5.1.1 induce sui naturali la relazione* $Succ \subseteq \mathbb{N} \times \mathbb{N} = \{(x, y) \in \mathbb{N} \times \mathbb{N} \mid y = x + 1\}$ *presentata nell'Esempio 2.1.8. Si osservi che la sua chiusura* transitiva *(non riflessiva),* $Succ^+$ *coincide con la relazione* $<$ *(minore stretto) sui naturali.*

La Definizione 5.3.1 induce sulle stringhe in $A^\star$ *relazione* $\prec_{A^\star} = \{(w, wa) \mid w \in A^\star, a \in A\}$*.*

La Definizione 5.3.13 induce sulle liste in L_A *relazione* $\prec_{L_A} = \{(lst, a\colon lst) \mid lst \in L_A, a \in A\}$*.*

La Definizione 5.3.20 induce sugli alberi binari in BT *la relazione* $\prec_{BT} = \{(t', t) \mid t'$ *è sottoalbero destro o sinistro di* $t\}$*.*

Spesso la definizione induttiva di una funzione su un insieme A definito induttivamente induce la stessa relazione di precedenza, cioè $\prec_A$. Questo vale per esempio per la la successione dei numeri triangolari (5.1.2) e per il fattoriale (5.2.4) definiti induttivamente sui naturali, per lunghezza e concatenazione di stringhe (5.3.4), per le operazioni strutturali su liste (Definizione 5.3.14), ed altre. Questo garantisce automaticamente che tali definizioni ricorsive siano ben date, e quindi le funzioni da esse caratterizzate sono totali.

Tuttavia non sempre è così. Per esempio la relazione di precedenza indotta dalla successione di Fibonacci (Esempio 5.2.13) è diversa da *Succ*. Infatti

$$\prec_{fib} = \{(n-1, n), (n-2, n) \mid n > 2\}$$

Non è difficile verificare direttamente che $\prec_{fib}$ è ben fondata, ma questo segue anche immediatamente dai seguenti fatti, di cui lasciamo la facile dimostrazione al lettore.

Proposizione 5.4.9 (Proprietà di relazioni ben fondate)**.**

1. *Se* $\sqsubset$ *è una relazione ben fondata e* $\sqsubset_1 \subseteq \sqsubset$*, allora anche* $\sqsubset_1$ *è ben fondata.*

2. *Una relazione* $\sqsubset$ *è ben fondata se e solo se lo è la sua chiusura transitiva* $\sqsubset^+$*.*

Poichè *Succ* è ben fondata, anche $Succ^+$ lo è per il punto 2. Nell'Esempio 5.4.8 abbiamo osservato che $Succ^+$ è la relazione $<$ su $\mathbb{N}$, e chiaramente $\prec_{fib} \subseteq <$, quindi $\prec_{fib}$ è ben fondata per il punto 1. Più in generale, qualunque definizione ricorsiva sui naturali in cui il valore per un $n \in N$ è espresso in termini del valore su argomenti *strettamente* minori di n è ben data.

Per concludere questa sezione torniamo alla Definizione (5.23).

Esempio 5.4.10 (La congettura di Collatz)**.** *Consideriamo di nuovo la funzione* (5.23)*, che ricopiamo qui per convenienza:*

$$f(n) = \begin{cases} 1 & \text{se } n \leq 1 \\ f(n/2) & \text{se } n > 1 \text{ ed è pari} \\ f(3 \cdot n + 1) & \text{se } n > 1 \text{ ed è dispari} \end{cases} \tag{5.27}$$

È facile convincersi che come relazione è univalente, e che se $f(k)$ è definito per un $k \in \mathbb{N}$ allora deve valere $f(k) = 1$. Quindi la funzione costante $f(n) = 1$ per ogni $n \in \mathbb{N}$ soddisfa le tre clausole della definizione, guardandola "dichiarativamente". Tuttavia è un problema aperto stabilire se la valutazione di $f(k)$ termini per qualunque $k \in \mathbb{N}$ oppure no. La Congettura di Collatz *afferma che questa funzione è totale: essa rimane al momento solo una congettura, anche se si è verificato che la valutazione termina per tutti i numeri fino a $5,764 \cdot 10^{18}$. Questa congettura rimane uno dei più interessanti problemi aperti della matematica per la semplicità della sua formulazione.*

5.5 Tipologie di Ricorsione

Le funzioni ricorsive (e induttive) delle sezioni precedenti presentavano tutte una forma di ricorsione *diretta*: in alcune clausole la valutazione della funzione poteva richiedere la valutazione della stessa su argomenti diversi. Esistono altre tipologie di definizioni ricorsive, di cui presentiamo di seguito una breve rassegna. Ne mostreremo qualche esempio, ma senza approfondirne la discussione: le problematiche di terminazione descritte nella sezione precedente si presentano naturalmente anche in questi casi.

Ricorsione annidata

Si ha *ricorsione annidata* quando una funzione compare come parametro in una sua chiamata ricorsiva.

Esempio 5.5.1. *La* funzione 91 *di MaCarthy è definita nel seguente modo:*

$$f(n) = \begin{cases} n - 10 & \text{se } n > 100 \\ f(f(n + 11)) & \text{se } n \leq 100 \end{cases} \tag{5.28}$$

Questa funzione è definita per ricorsione annidata. Si vede subito che come relazione è univalente, ma non è immediato vedere che sia totale, perché nella clausola ricorsiva la funzione è applicata due volte. Con un po' di sforzo si può verificare che $f(n) = 91$ per ogni $n \leq 100$ e $f(n) = n - 10$ per ogni $n > 100$.

Esempio 5.5.2. *La* funzione di Ackermann *è definita per ricorsione annidata come segue:*

$$A(m, n) = \begin{cases} n + 1 & \text{se } m = 0 \\ A(m - 1, 1) & \text{se } m > 0 \text{ e } n = 0 \\ A(m - 1, A(m, n - 1)) & \text{se } m > 0 \text{ e } n > 0 \end{cases} \tag{5.29}$$

Si può dimostare che la definizione caratterizza una funzione totale. Si tratta di una funzione che cresce molto rapidamente, anche per valori piccoli del primo argomento.

Ricorsione mutua

Quando la chiamata ricorsiva avviene indirettamente, cioè da parte di una seconda funzione chiamata a sua volta, direttamente o indirettamente, dalla prima, si parla di *ricorsione mutua* o *indiretta*.

Un esempio artificiale, dove le due funzioni sono ovviamente totali su $\mathbb{N}$, è il seguente:

$$ping(n) = \begin{cases} 0 & \text{se } n = 0 \\ pong(n - 1) & \text{altrimenti} \end{cases} \qquad pong(n) = \begin{cases} 0 & \text{se } n = 0 \\ ping(n - 1) & \text{altrimenti} \end{cases}$$

Ricorsione procedurale

In questo capitolo abbiamo discusso della definizione di *funzioni ricorsive*, intendendo *funzioni* nel senso matematico del termine, come introdotte nella Definizione 2.5.1. Ma quando si deve risolvere un problema con un programma, nella maggior parte dei linguaggi di programmazione si possono usare anche effetti collaterali (come input e output, o modifica di variabili/array con assegnamento) che non sono rappresentabili immediatamente come funzioni matematiche. In molti linguaggi una *procedura* può essere invocata passando degli argomenti, ma non ci si aspetta che restituisca un risultato. Un tipico esempio sono le funzioni in C/C++ o i metodi in Java che sono dichiarati di tipo `void`.

Anche le procedure naturalmente possono essere ricorsive, terminando quando l'input è "abbastanza piccolo" (tipicamente senza fare niente, o altrimenti invocando se stesse su argomenti "più piccoli". Come semplice esempio illustrativo vediamo una procedura C che stampa ricorsivamente tutti gli elementi di un array.

```
void stampa_array(int a[], int i, int n) {
   if (i < n)   // clausola ricorsiva: c'e' ancora qualcosa da stampare
       {     printf("%d ", a[i]);
             stampa_array(a, i+1, n);
   }    // clausola base: i == n; l'array a[i..n-1] e' vuoto, la procedura termina
}
```

5.6 Esercizi

A complemento degli esercizi su induzione sui naturali compresi nella Sezione 5.2, proponiamo al lettore alcuni altri esercizi di quel tipo.

Esercizio 5.6.1. *Dimostrare per induzione che per ogni* $n \in \mathbb{N}$ *il valore* $n^3 - n$ *è divisibile per* 3.

Esercizio 5.6.2. *Dimostrare per induzione che per ogni* $n \in \mathbb{N}$ *il valore* $2^n \cdot 2^n - 1$ *è divisibile per* 3.

Esercizio 5.6.3. *Dimostrare per induzione che per ogni* $n \in \mathbb{N}$ *il valore* $n^2 + n$ *è divisibile per* 2.

Esercizio 5.6.4. *Dimostrare per induzione che la somma dei primi* n *numeri dispari è uguale a* n^2. *In formule:*

$$\forall n \in \mathbb{N}^+ \left(\sum_{i=1}^{n} (2 \cdot i - 1) = n^2 \right)$$

Esercizio 5.6.5. *Consideriamo la generalizzazione della proprietà distributiva:*

$$D(n) = \forall A, A_1, ..., A_n \left(A \cap \left(\bigcup_{i=1}^{n} A_i \right) = \bigcup_{i=1}^{n} (A \cap A_i) \right)$$

Dimostrare per induzione che:

$$\forall n \, (\, n \geq 2 \, \rightarrow \, D(n) \,)$$

CAPITOLO 6

Calcolo Combinatorio

6.1 Insiemi

Molti dei problemi che vedrete durante il corso dei vostri studi (e che dovrete affrontare nella vostra vita di Informatici) possono, in linea del tutto generale, essere ridotti alla seguente domanda:

quanti elementi ha un dato insieme?

Questa quantità è chiamata "cardinalità".

Definizione 6.1.1. *Sia $A = \{a_1, a_2, \ldots, a_n\}$ un insieme contenente n elementi (con $n \geq 1$ se l'insieme non è vuoto, altrimenti $n = 0$) presi da un universo U. In questo caso diciamo che A è un insieme* FINITO *e che A ha* CARDINALITÀ *n, indicata anche con $|A|$.*

Notazione 6.1.2. $|A| = n$.

Esempio 6.1.3.

- *l'insieme delle vocali $V = \{a, e, i, o, u\}$ ha cardinalità 5, cioè $|V| = 5$.*
- $|AN| = 62$ *(solo 52 caratteri alfanumerici se consideriamo l'alfabeto italiano invece di quello inglese).*
- $|\emptyset| = 0$.
- *Ricordiamoci che ogni numero naturale $n \in \mathbb{N}$, può essere visto come l'insieme $\{0, 1, \ldots n-1\}$. È facile da vedere che, per tutti gli $n \in N$, $|n| = n$.*

E $|\mathbb{N}|$? Oppure $|\mathbb{R}|$? Sono esempi di insiemi *infiniti* (ovvero contenenti un numero di elementi non finito). Ci sono molti tipi diversi di infiniti (ad esempio $|\mathbb{N}| < |\mathbb{R}|$) che sono studiati nella teoria dei numeri ordinali e cardinali. Questa teoria, per quanto affascinante, sarà utile (e comprensibile) solo a pochi di voi. Per questa ragione, durante il corso studieremo solo insiemi di cardinalità finita.

Osservazione 6.1.4. *Nel resto di questo capitolo, ogni volta che ci riferiremo alla cardinalità di un insieme, staremo assumendo implicitamente che l'insieme è finito. Pertanto, eviteremo di specificare ogni volta insieme* finito.

Esercizio 6.1.5. *Determinare la cardinalità di ciascuno dei seguenti insiemi:* $\{\, a \,\}$, $\{\, \{a\} \,\}$, $\{\, a, \{a\} \,\}$, $\{\, a, \{a\}, \{a, \{a\}\} \,\}$, $\emptyset$, $\{\, \emptyset \,\}$, $\{\, \emptyset, \{\emptyset\} \,\}$, $\{\, \emptyset, \{\emptyset\}, \{\emptyset, \{\emptyset\}\} \,\}$

In questo capitolo useremo le nozioni sugli insiemi viste precedenti, in particolare, le definizioni di complemento $\bar{A}$, differenza $A \setminus B$, intersezione $A \cap B$ e unione $A \cup B$. La cardinalità di tali insiemi dovrebbe essere quindi chiara dalla loro definizione. Occorre porre un po' di attenzione al complemento: anche se A è finito, il suo complemento $\bar{A} = U \setminus A$ potrebbe non esserlo a causa dell'universo U. Per esempio, se $A = \{1, 2, 3, \ldots, n\}$ è l'insieme dei primi n naturali, il suo complemento è infinito essendo $U = \mathbb{N}$. Può essere utile visualizzare graficamente la dimensione di tali insiemi usando i diagrammi di Venn dove A e B si intersecano parzialmente, evidenziandone la parte relativa all'operazione insiemistica corrispondente.

Vediamo alcune proprietà utili.

Proposizione 6.1.6. *Per tutti gli insiemi A vale che se $mathcalF = \{A_i \mid i \in I\}$ è una partizione di A, allora $|A| = \sum_{i \in I} |A_i|$.*

Dimostrazione. Tutti e soli gli elementi di A appaiono esattamente una volta in uno degli A_i. Contare quanti elementi sono contenuti in A è quindi lo stesso che contare quanti elementi sono in tutti gli A_i, poiché non ci possono essere ripetizioni per definizione di partizione. ■

Proposizione 6.1.7. *Per tutti gli insiemi A, B vale che $A = (A \setminus B) \cup (A \cap B)$ e, inoltre, $\{A \setminus B, A \cap B\}$ è una partizione di A quando $A \neq B$ e $A \cap B \neq \varnothing$.*

Dimostrazione. Se $B = \varnothing$ oppure $A \cap B = \varnothing$ (cioè sono disgiunti), è facile verificare che la proposizione vale mediante sostituzione e ispezione diretta: infatti $A \setminus B = A$ perchè non ci possono essere elementi comuni tra A e B, ed è sempre $A \setminus B = \varnothing$.

Nel resto del capitolo, consideremo quindi sempre il caso interessante che rimane: negando la condizione $B = \varnothing$ oppure $A \cap B = \varnothing$, otteniamo $B \neq \varnothing$ e $A \cap B \neq \varnothing$. In altre parole, è il caso in cui A e B sono (parzialmente) intersecati. Prendiamo un qualunque elemento $a \in A$ e consideriamo i due possibili casi:

- $a \in B$: quindi $a \in A \cap B$ per definizione;
- $a \notin B$: quindi $a \in A \setminus B$ per definizione.

Quando $A \neq B$ vale $A \setminus B \neq \varnothing$ e, insieme alla condizione $A \cap B \neq \varnothing$, implicano che $\{A \setminus B, A \cap B\}$ è una partizione di A perché $A \setminus B$ e $A \cap B$ sono non vuoti e disgiunti, dimostrando l'enunciato della proposizione. ■

La seguente proprietà sarà utilizzata varie volte in seguito, perchè nell'unione $\mathbb{A} \cup B$ occorre prestare attenzione a non contare *due* volte gli elementi comuni ad A e B.

Proposizione 6.1.8. *Per tutti gli insiemi A, B vale che $|A \cup B| = |A| + |B| - |A \cap B|$.*

Dimostrazione. Lasciamo da verificare per ispezione diretta il caso semplice $B = \varnothing$ oppure $A \cap B = \varnothing$ al lettore. Ci concentriamo sul caso interessante $B \neq \varnothing$ e $A \cap B \neq \varnothing$. Possiamo partizionare $A \cup B$ in tre insiemi: $A \setminus B$, $A \cap B$, $B \setminus A$. La Proposizione 6.1.7 ci dice che $A = (A \setminus B) \cup (A \cap B)$ e $B = (B \setminus A) \cup (A \cap B)$.

Ora prendiamo in considerazione le cardinalità e osserviamo che $|A \cup B| = |A \setminus B| + |A \cap B| + |B \setminus A|$ per la Proposizione 6.1.6, avendo preso una partizione dell'unione.

Osserviamo che data un'uguaglianza $X = Y$, possiamo sempre mantenerla aggiungendo togliendo lo stesso termine: $X = Y$ se e solo se $X = Y + 0$ se e solo se $X = Y + T - T$ per qualunque valore T dello stesso tipo di X e Y. Applichiamo questa osservazione alla nostra disuguaglianza dove $X = |A \cup B|$, $Y = |A \setminus B| + |A \cap B| + |B \setminus A|$ e $T = |A \cap B|$. Otteniamo $|A \cup B| = |A \setminus B| + |A \cap B| + |B \setminus A| + |A \cap B| - |A \cap B|$.

Applicando il fatto che $A = (A \setminus B) \cup (A \cap B)$ e $B = (B \setminus A) \cup (A \cap B)$, riscriviamo l'uguaglianza come $|A \cup B| = |A| + |B| - |A \cap B|$. ■

Ci sono due rilevanti implicazioni della proprietà appena vista.

Corollario 6.1.9.

- $|A \cup B| \leq |A| + |B|$, *perché* $|A \cap B| \geq 0$ *(cioè* $-|A \cap B| \leq 0$*) e quindi* $|A| + |B| - |A \cap B| \leq |A| + |B|$.
- *A e B disgiunti implica che* $|A \cup B| \leq |A| + |B|$ *perché* $|A \cap B| = 0$.

Proposizione 6.1.10. *Per tutti gli insiemi A, B vale che se $B \subseteq A$, allora $|B| \leq |A|$.*

Dimostrazione. Sappiamo che $A = (A \setminus B) \cup (A \cap B)$ per la Proposizione 6.1.7. Poiché $B \subseteq A$, vale che $(A \cap B) = B$. Possiamo quindi partizionare A come $A \setminus B$ e B. Quindi $|A| = |A \setminus B| + |B|$ per la Proposizione 6.1.6. Essendo $|A \setminus B| \geq 0$ per definizione, $|A| \geq 0 + |B| = |B|$. ■

Riprendiamo ora quanto enunciato nella Proposizione 6.1.8, cioè $|A \cup B| = |A| + |B| - |A \cap B|$, e chiediamoci cosa succede all'unione se prendiamo tre insiemi $A, B.C$. La formula diventa $|A \cup B \cup C| = |A| + |B| + |C| - |A \cap b| - |A \cap C| - |B \cap C| + |A \cap B \cap C|$. Questi sono casi speciali del *principio di inclusione-esclusione.*

Proposizione 6.1.11. *Presi r insiemi $S_1, S_2, \ldots, S_r$, abbiamo la seguente uguaglianza, dove $(-1)^i$ vale 1 se i è un numero pari e vale -1 se i è dispari:*

$$\left|\bigcup_{j=1}^{r} S_j\right| = \sum_{I \subseteq \{1,2,\ldots,r\}, I \neq \varnothing} (-1)^{|I|+1} \left|\bigcap_{i \in I} S_i\right|$$

Dimostrazione. La dimostrazione non viene mostrata, ma è importante ricordarsi l'enunciato. ■

Osservazione 6.1.12. *Diverse applicazioni nell'information retrieval (e non solo) richiedono di determinare la similarità di documenti testuali, per esempio, le pagine di Wikipedia. Viene adottato il modello "bags of words", dove l'universo U è l'insieme di tutte le parole usate nel mondo, dove le parole sono lemmatizzate: in parole semplici, i verbi sono portati tutti all'infinito e le parole al singolare maschile. Ogni documento D viene rappresentato come una "bag of words", cioè un sottoinsieme $S_D \subseteq U$: semplicemente si prendono le parole distinte in D, ignorando il loro ordine di apparizione dentro D. La similarità tra due documenti viene quindi stabilita mediante i loro corrispettivi insiemi $A, B \subseteq U$. Tra gli indici adottati, c'è* l'indice di Jaccard*, definito come*

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Se $A = B$, vale che $J(A, B) = 1$. Se $A \cap B = \varnothing$, vale che $J(A, B) = 0$. In generale, vale che $0 \leq J(A, B) \leq 1$, e quanto più alto è $J(A, B)$, tanto più simili sono gli insiemi A e B (e quindi i documenti che rappresentano).

Consieriamo ora il prodotto cartesiano $A \times B = \{(a, b) \mid a \in A, b \in B\}$. Ci aspettiamo che valga la seguente proprietà, la cui dimostrazione richiede un po' di attenzione.

Proposizione 6.1.13. *Per tutti gli insiemi A, B vale che $|A \times B| = |A| \cdot |B|$.*

Dimostrazione. Definiamo $S_a = \{(a, b) \mid b \in B\}$ per un elemento fissato $a \in A$. Valgono le seguenti osservazioni:

- $|S_a| = |B|$ per costruzione perché ci sono tante coppie in S_a quanti sono gli elementi b in B.
- $a \neq a'$ implica che $S_a \cap S_{a'} = \varnothing$: presa una qualuqnue coppia $(a, b) \in S_a$ e una qualunque coppia $(a', b') \in S_{a'}$, poiché $a \neq a'$ per ipotesi, abbiamo che vale sempre $(a, b) \neq (a', b')$ indipendentemente da b e b'.
- $\cap_{a \in I} S_a = \varnothing$ per qualunque sottoinsieme I di A tale che $|I| \geq 2$: se intersechiamo due o più insiemi S_a, $a \in I$, otteniamo sempre l'insieme vuoto per l'osservazione precedente (ne bastano due).
- $\cup_{a \in A} S_a = A \times B$ perché tutte le possibili coppie (a, b), con $a \in A, b \in B$, sono ottenute in tal modo.

Utilizziamo le osservazioni per il nostro enunciato: $|A \times B| = |\cup_{a \in A} S_a|$ per l'osservazione 4. Usiamo le osservazioni 2 e 3 nel principio di inclusione-esclusione, dove tutte le intersezini di due o più sottoinsiemi sono vuote, rimanendo quindi soltanto la somma delle cardinalità degli insiemi: $|\cup_{a \in A} S_a| = \sum_{a \in A} |S_a|$. Infine usiamo l'osservazione 1 per dire che tutte queste cardinalità sono uguali a $|B|$: $\sum_{a \in A} |S_a| = \sum_{a \in A} |B|$. Ora, se sommiamo t volte la stessa quantità $|B|$, equivale a moltiplicare t per $|B|$, dove $t = |A|$ nel nostro caso: $\sum_{a \in A} |B| = |A| \cdot |B|$, dimostrando l'enunciato. ■

Esercizio 6.1.14. *Modificare la dimostrazione, evitando di usare il principio di inclusione-esclusione e osservando che $\{S_a \mid a \in A\}$ è una partizione di $A \times B$.*

Se passiamo al prodotto cartesiano di più insiemi, caratterizziamo le n-ple (anche note come tuple). Ci aspettiamo che valgano relazioni come $|A\times B\times C| = |A|\cdot|B|\cdot|C|$ e $|A\times B\times C\times D| = |A|\cdot|B|\cdot|C|\cdot|D|$ per esempio. Proviamo che uguaglianze di questo tipo valgono usando l'induzione.

Proposizione 6.1.15. *Per tutte le scelte di insiemi $A_1, A_2, \ldots, A_{r-1}, A_r$, con $r \geq 2$, vale che*

$$|A_1 \times A_2 \times \cdots \times A_{r-1} \times A_r| = |A_1| \cdot |A_2| \cdots |A_{r-1}| \cdot |A_r|$$

Dimostrazione. Procediamo per induzione su $r \geq 2$. Caso base $r = 2$. Vale $|A_1 \times A_2| = |A_1| \cdot |A_2|$ per la Proposizione 6.1.13, ponendo $A = A_1$ e $B = A_2$.

Passo induttivo $r > 2$. Usando l'associatività possiamo scrivere $A_1 \times A_2 \times \cdots \times A_{r-1} \times A_r = (A_1 \times A_2 \times \cdots \times A_{r-1}) \times A_r$. Applichiamo la Proposizione 6.1.13, ponendo $A = (A_1 \times A_2 \times \cdots \times A_{r-1})$ e $B = A_r$, ottenendo così

$$|A_1 \times A_2 \times \cdots \times A_{r-1} \times A_r| = |A_1 \times A_2 \times \cdots \times A_{r-1}| \cdot |A_r|$$

Per ipotesi induttiva, sappiamo che

$$|A_1 \times A_2 \times \cdots \times A_{r-1}| = |A_1| \cdot |A_2| \cdots |A_{r-1}|$$

e quindi, sostituendo, otteniamo

$$|A_1 \times A_2 \times \cdots \times A_{r-1} \times A_r| = |A_1 \times A_2 \times \cdots \times A_{r-1}| \cdot |A_r| = |A_1| \cdot |A_2| \cdots |A_{r-1}| \cdot |A_r|$$

■

Esempio 6.1.16. *Le targhe automobilistiche italiane hanno il formato* `AA000AA` *con 22 lettere utilizzate (prese dalle 26 lettere dell'alfabeto inglese, tranne I, O, Q, U perché potrebbero generare confusione nella lettura). Ogni targa può essere quindi vista come una tupla di $r = 7$ insiemi, $A_1 \times A_2 \times A_3 \times A_4 \times A_5 \times A_6 \times A_7$, dove $|A_1| = |A_2| = |A_6| = |A_7| = 22$ e $|A_3| = |A_4| = |A_5| = 10$. È quindi possibile avere $22^4 \cdot 10^3 = 234\,256\,000$ targhe distinte in questo modo.*

Le implicazioni della Proposizione 6.1.15 sono importanti. Per esempio una sequenza di lunghezza n su A può essere vista come una n-pla di $A \times A \times \cdots \times A = A^n$, dove $A = A_1 = A_2 = \cdots = A_n$.

Corollario 6.1.17. *Sia A^n l'insieme delle sequenze di lunghezza n su un insieme A. La sua cardinalità è $|A^n| = |A|^n$.*

Esempio 6.1.18. *Per esempio, nelle sequenze binarie, vale che $A = \{0, 1\} = 2$ e quindi ce ne sono $|2^n| = |2|^n = 2^n$. Un altro esempio sono le stringhe in formato ASCII esteso, dove ogni carattere occupa un byte (8 bit). In questo caso $|A| = 256 = 2^8$ e ci sono 256^n sequenze di lunghezza n in ASCII esteso.*

Vediamo adesso un modo alternativo per mostrare che ci sono 2^n stringhe binarie di lunghezza n utilizzando l'induzione. Questo ci tornerà utile in seguito per contare i sottoinsiemi di cardinalità prefissata.

Indichiano con $B(n)$ il numero di sequenze binarie di lunghezza n (ovviamente sappiamo già che $B(n) = 2^n$, ma qui vogliamo invece dedurlo).

Se $n = 0$ otteniamo la sequenza vuota e quindi $B(0) = 1$, che rappresenta la clausola base.

Per $n \geq 1$, osserviamo che vale uno schema ricorsivo: prendiamo tutte le $B(n-1)$ sequenze binarie di lunghezza $n-1$ e ci appendiamo un bit $b = 0$ in fondo; poi prendiamo di nuovo tutte le $B(n-1)$ sequenze binarie di lunghezza $n-1$ e questa volta ci appendiamo un bit $b = 1$ in fondo. I due gruppi di sequenze così ottenute sono disgiunti e la loro unione fornisce tutte le $B(n)$ sequenze binarie di lunghezza n: abbiamo pertanto ottenuno una partizione in due di queste ultime.

Per $n \geq 1$, la clausola induttiva è $B(n) = B(n-1) + B(n-1)$, dove il primo $B(n-1)$ corrisponde al caso $b = 0$ e il secondo al caso $b = 1$. Quindi $B(n) = 2B(n-1)$.

Mostriamo per induzione che $B(n) = 2^n$.

Caso base: $B(0) = 2^0 = 1$.

Passo induttivo: $B(n) = 2B(n-1) = 2 \cdot 2^{n-1} = 2^n$, applicando l'ipotesi induttiva che $B(n-1) = 2^{n-1}$.

6.2 Relazioni

Dal momento che ogni relazione $R \subseteq A \times B$ è essa stessa un insieme, possiamo parlare di cardinalità di una relazione. Da quanto visto finora sappiamo che $|R| \leq |A \times B|$. Possiamo dire qualcosa in più? Quando R gode delle proprietà viste nella Sezione 2.4 possiamo stabilire dei limiti più precisi.

Proposizione 6.2.1. *Per tutti gli insiemi A, B e per tutte le relazioni $R \subseteq A \times B$ vale che:*

6.2.1.1. *Se R è totale, allora $|A| \leq |R|$;*

6.2.1.2. *Se R è univalente, allora $|R| \leq |A|$;*

6.2.1.3. *Se R è surgettiva, allora $|B| \leq |R|$;*

6.2.1.4. *Se R è iniettiva, allora $|R| \leq |B|$;*

Dimostrazione. Dimostriamo i primi due punti. Il terzo e quarto sono lasciati come esercizio.

1. Se R è totale, allora per tutti gli $a \in A$ c'è almeno una coppia $(a, b) \in R$ per un qualche $b \in B$. Pertanto $|A| \leq |R|$.
2. Se R è univalente, allora per tutti gli $a \in A$ c'è al più una coppia $(a, b) \in R$ per un qualche $b \in B$. Essendo $R \subseteq A \times B$, ogni elemento di R è una coppia (x, y) con $x \in A$ e $y \in B$. Visto che non ci possono essere in R due coppie (a, y_1) e (a, y_2) con lo stesso a ma con $y_1 \neq y_2$, allora il numero di coppie in R è necessariamente minore o uguale del numero di elementi di A. Pertanto $|R| \leq |A|$.

■

Esercizio 6.2.2. *È vero che per tutti gli insiemi A, B e tutte le relazioni $R \subseteq A \times B$ vale che $|R^{op}| = |R|$? In caso affermativo, mostrare una dimostrazione. In caso negativo, fornire un controesempio.*

Esercizio 6.2.3. *Dimostrare le Proposizioni 6.2.1.3 e 6.2.1.4.*

È importante notare che le quattro proprietà fondamentali delle relazioni giocano un ruolo chiave anche nelle cardinalità. Per evidenziare la dualità che emerge ancora una volta, abbiamo riassunto la Proposizione 6.2.1 nella Tabella 6.1.

totale	$\Rightarrow$	$\lvert A\rvert \leq \lvert R\rvert$	surgettiva	$\Rightarrow$	$\lvert B\rvert \leq \lvert R\rvert$
univalente	$\Rightarrow$	$\lvert R\rvert \leq \lvert A\rvert$	iniettiva	$\Rightarrow$	$\lvert R\rvert \leq \lvert B\rvert$

Tabella 6.1: Proprietà di relazioni e cardinalità

La Proposizione 6.2.1 ha una conseguenza immediata sulle funzioni.

Corollario 6.2.4. *Per tutti gli insiemi A, B vale che se $R \subseteq A \times B$ è una funzione, allora $|A| = |R|$.*

Dimostrazione. Essendo R una funzione, è sia totale che univalente. Quindi entrambe le condizioni $|A| \leq |R|$ e $|R| \leq |A|$ sono verificate. Unite al fatto che $\leq$ è antisimmetrica, ci permettono di concludere che $|A| = |R|$. ■

Ai fini del conteggio, quando R è una funzione, può essere vista come una n-pla con $n = |A|$ su B^n. Infatti, dato $A = \{a_1, a_2, \ldots, a_n\}$, abbiamo $|B|$ scelte da cui pescare l'immagine $R(a_i)$ per $1 \leq i \leq n$. Ne deriva che il numero di funzioni da A a B è uguale al numero di n-ple in B^n, ossia $|B^n| = |B|^n = |B|^{|A|}$.

Esempio 6.2.5. *Supponendo di dover fornire le previsioni metereologiche per la prossima settimana, possiamo denotare con A i sette giorni della settimana e con B le possibili previsioni: per esemio, $B = \{sole, nebbia, pioggia, neve\}$. Una previsione è una funzione che assegna una delle previsioni in B a* ciascun *giorno in A. Esistono quindi $|B|^{|A|} = 4^7$ possibili previsioni.*

Osservazione 6.2.6. *Un caso particolare della Proposizione 6.2.1 è il* principio delle buche dei piccioni*: per tutti gli insiemi A, B vale che se esiste una relazione $R \subseteq A \times B$ totale e iniettiva, allora $|A| \leq |B|$. Infatti, valgono $|A| \leq |R|$ e $|R| \leq |B|$ e quindi si applica la transitività.*

Questo principio è solitamente formulato in termini di funzioni iniettive (cioè relazioni totali e iniettive che sono anche univalenti): Per tutti gli insiemi A, B, vale che se esiste una funzione iniettiva $f\colon A \to B$, allora $|A| \leq |B|$ (chiaramente il risultato vale ancora indipendentemente dal fatto che sia univalente). Inoltre il principio viene formulato in maniera contronominale*: Per tutti gli insiemi A, B vale che se $|A| > |B|$, allora non esiste alcuna funzione iniettiva $f\colon A \to B$. (Se infatti esistesse una tale f, avremmo una contraddizione in quanto sarebbe $|A| \leq |B|$.)*

Biiezioni

La Proposizione 6.2.1 ha una conseguenza immediata di fondamentale importanza: prendendo relazioni totali, univalente, surgettive ed iniettive, cioè delle biiezioni, si ottiene la famosa *regola di biiezione*, che mostreremo come utilizzare in seguito.

Corollario 6.2.7 (Regola di biiezione)**.** *Per tutti gli insiemi A, B vale che se esiste una biiiezione $R \subseteq A \times B$, allora $|A| = |B|$.*

Dimostrazione. Assumiamo che esista $R \subseteq A \times B$ totale, univalente, surgettiva ed iniettiva. Allora dalla Proposizione 6.2.1 si ha che

$$\begin{array}{ll} |A| \leq |R| & |B| \leq |R| \\ |R| \leq |A| & |R| \leq |B| \end{array}$$

Dall'antisimmetria di $\leq$, ne segue che

$$|A| = |R| \quad |R| = |B|$$

e quindi $|A| = |B|$. ■

La regola è molto utile nel seguente scenario per due insieme A e B:

- dobbiamo contare A, cioè valutare la sua cardinalità $|A|$;
- abbiamo già contato B, cioè valutato la sua cardinalità $|B|$;
- se troviamo una biiezione R tra A e B (quindi $A \cong B$) allora possiamo dedurre il valore della cardinalità $|A|$, perché $|A| = |B|$.

Illustriamo subito tale procedimento per l'insieme delle parti $\wp(A) = \{\, B \mid B \subseteq A \,\}$. Abbiamo visto nella Proposizione 2.6.21 che vale che $\wp(A) \cong 2^A$, quindi possiamo concludere adesso che $|\wp(A)| = |2^A| = |2|^{|A|} = 2^{|A|}$. Utilizzeremo questa biiezione in seguito e per questo ricordiamo che è utile considerare la funzione caratteristica $\chi_B\colon A \to 2$ definita come

$$\chi_B(a) = \begin{cases} 1 & \text{se } a \in B \\ 0 & \text{se } a \notin B \end{cases}$$

perché trasforma un sottoinsieme B di A in una sequenza binaria di lunghezza $n = |A|$, dove ci sono esattamente $k = |B|$ bit a 1.

A questo punto è naturale la seguente questione: la regola di biiezione ci dice che se due insiemi sono in biiezione allora hanno la stessa cardinalità; è vero il contrario, cioè, se due insiemi hanno la stessa cardinalità, allora sono in biiezione? La risposta è positiva.

Per dimostrarlo è necessario ricordare ogni numero naturale $n \in \mathbb{N}$, rappresenta anche l'insieme $n = \{0, 1, \ldots, n-1\}$ ed osservare il seguente semplice fatto.

Proposizione 6.2.8. *Per tutti gli insiemi A e per tutti i numeri naturali $n \in \mathbb{N}$ vale che, se $|A| = n$, allora $A \cong n$.*

Dimostrazione. Per definizione $|A| = n$ significa che A contiene esattamente n elementi: possiamo quindi scrivere $A = \{a_1, a_2, \dots a_n\}$ per qualche elemento a_i. Definiamo adesso la funzione $f\colon A \to n$ (per $n = \{0, 1, \dots, n-1\}$) come

$$a_i \mapsto i - 1$$

e la funzione $g\colon n \to A$ come

$$i \mapsto a_{i+1}.$$

È immediato verificare che g è la funzione inversa di f e che quindi f è una biiezione. Pertanto $A \cong n$. ■

Questo è sufficiente per fornire una risposta positiva alla domanda.

Teorema 6.2.9. *Per tutti gli insiemi A e per tutti gli insiemi B, vale che*

$$A \cong B \text{ se e solo se } |A| = |B|.$$

Dimostrazione. Trattandosi di un se e solo se possiamo spezzare la dimostrazione in due.

- Se $A \cong B$, allora $|A| = |B|$. Immediato dalla legge di biiezione (Corollario 6.2.7).
- Se $|A| = |B|$, allora $A \cong B$. Assumiamo che $|A| = |B| = n$ per un qualche numero naturale n. Dalla Proposizione 6.2.8 si ha quindi che $A \cong n$ e $B \cong n$. Per simmetria e transitività di $\cong$ (Proposizione 2.6.25) vale che $A \cong B$.

■

Utilizzando la biiezione abbiamo visto come contare i sottoinsieme di un insieme A, cioè stabilire la cardinalità del suo insieme delle parti $\wp(A)$. Cosa si può dire se ci restringiamo ai sottoinsiemi di cardinalità k per un dato k?

Definizione 6.2.10. *Dati due insiemi A e B e un intero k, dove $0 \leq k \leq |A|$, diciamo che B è un k-insieme di A se $B \subseteq A$ e $|B| = k$.*

Vogliamo quindi stabilire quanti k-insiemi ci sono in A. La cosa interessante è che la risposta non dipende da quale tipo di elementi sono contenuti in A, ma dal loro numero $n = |A|$ e da k. Indichiamo con $\binom{n}{k}$ tale numero, noto come *coefficiente binomiale*. Per poter stabilire ciò usiamo la seguente proposizione.

Proposizione 6.2.11. *Esiste una biiezione tra i k-insiemi di A e le sequenze binarie di lunghezza $n = |A|$ aventi* esattamente *k bit a 1.*

Dimostrazione. La biiezione è esattamente quella mostrata nella dimostrazione della proposizione 2.6.21, dove si utilzza la funzione caratteristica χ_B per provare che $\wp(A) \cong 2^A$. ■

Vediamo come definire ricorsivamente il coefficiente binomiale $\binom{n}{k}$ alla luce di ciò. L'idea è la seguente: prendiamo tutte le sequenze binarie di lunghezza n con k bit a 1 e guardiamo il loro *ultimo* bit b. In altre parole, ciascuna di tale sequenze avrà $n-1$ bit, che chiamiamo α, seguiti dal bit b. Ovviamente ogni sequenza avrà il suo α e il suo b, e il loro numero totale è $\binom{n}{k}$. Possiamo partizionare tali sequenze in due gruppi:

- le sequenze che hanno $b = 0$: in tali sequenze α deve necessariamente avere k bit a 1, altrimenti non farebbero parte della nostra argormentazione;
- le sequenze che hanno $b = 1$: qui α has $k-1$ bit a 1, poiché sappiamo che il k-esimo 1 è appnto b.

I due gruppi suddetti partizionano tutte le sequenze di nostro interesse e possiamo indicare il loro numero usando il coefficiente binomiale, osservando che α ha $n-1$ bit:

- le sequenze che hanno $b = 0$: ci sono $\binom{n-1}{k}$ tali sequenze perché, fissato $b = 0$, solo gli $n-1$ bit in α possono variare con la condizione che esattamente k siano a 1;

- le sequenze che hanno $b = 1$: ci sono $\binom{n-1}{k-1}$ tali sequenze perché, fissato $b = 1$, solo gli $n - 1$ bit in α possono variare con la condizione che esattamente $k - 1$ siano a 1.

Mettendo insieme le osservazioni sopra, possiamo scrivere la nostra clausola ricorsiva come

$$\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1}$$

La clausola base è un po' più impegnativa del solito, perché deve catturare le sequenti situazioni dove le scelte di b diventano obbligate:

- $n = 0$: c'è una sola sequenza ed è quella vuota;
- $k = 0$: c'è una sola sequenza ed è quella composta da n bit tutti a 0 (nota: poiché $n = 0$ implica che $k = 0$, essendo $0 \leq k \leq n$, il caso precedente viene assorbito da questo;
- $k = n$: c'è una sola sequenza ed è quella composta da n bit tutti a 1.

Possiamo quindi scrivere che

$$\binom{n}{0} = \binom{n}{n} = 1$$

Utilizziamo ora la nozione di numero fattoriale vista in precedenza ($n! = n \cdot (n-1) \cdot (n-2) \cdots 3 \cdot 2 \cdot 1$, dove si pone $0! = 1$) per provare per induzione che la formula chiusa del coefficiente binomiale è la legge dei tre fattoriali.

Proposizione 6.2.12.

$$\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$$

Dimostrazione. Caso base: $\binom{n}{0} = \frac{n!}{0!n!} = 1$ e $\binom{n}{n} = \frac{n!}{n!0!} = 1$.

Passo induttivo: $\binom{n}{k} = \binom{n-1}{k} + \binom{n-1}{k-1} = \frac{(n-1)!}{k!\,(n-1-k)!} + \frac{(n-1)!}{(k-1)!\,(n-k)!} = \frac{(n-1)!}{(k-1)!\,(n-1-k)!} \cdot [1/k + 1/(n-k)] = \frac{(n-1)!}{(k-1)!\,(n-1-k)!} \cdot \frac{n}{k\,(n-k)} = \frac{n!}{k!\,(n-k)!}$, ricordando che $x! = x \cdot (x-1)!$. ■

Combinazioni

La scelta dei k-insiemi di A è noto anche come problema delle *combinazioni* di k elementi di A, ovvero tutti i possibili modi di scegliere esattamente k elementi di A, ignorando il loro ordine, che sappiamo ora essere $\binom{n}{k} = \frac{n!}{k!\,(n-k)!}$. Questa formula viene spesso incontrata nel calcolo combinatorio.

Esempio 6.2.13. *Sia il nostro gruppo composto da Anna, Bruno, Ciro, Daniela, ed Enrico, rappresentati dall'insieme $A = \{a, b, c, d, e\}$, ed ammettiamo di avere un tavolo da 3 posti: le posibili combinazioni di 3 elementi di A sono $\binom{5}{3} = \frac{5!}{3!(5-3)!} = \frac{5 \cdot 4 \cdot 3 \cdot 2 \cdot 1}{(3 \cdot 2 \cdot 1)(2 \cdot 1)} = \frac{5 \cdot 4}{2} = 10$. Anche qui possiamo verificare il calcolo a mano:*

- *abc*
- *abd*
- *abe*
- *acd*
- *ace*
- *ade*
- *bcd*
- *bce*
- *bde*
- *cde*

Osserviamo che "scegliere k elementi da n" equivale esattamente a "lasciare fuori $n - k$ elementi" (se di 5 ne sediamo 3 al tavolo, 2 rimangono fuori). Tuttavia, "lasciare fuori $n - k$ elementi da n" è solo un altro modo di dire "scegliere $n - k$ elementi da n" (ovvero, tutti i modi di scegliere chi *non* si siede). In altre parole abbiamo la seguente proprietà.

Proposizione 6.2.14.

$$\binom{n}{k} = \binom{n}{n-k}$$

Dimostrazione. La proprietà si può dimostrare semplicemente andando a sviluppare $\binom{n}{n-k}$: $\binom{n}{n-k} = \frac{n!}{(n-k)!(n-(n-k))!} = \frac{n!}{(n-k)!(k)!} = \binom{n}{k}$. ■

È interessante collegare quanto visto finora per il coefficiente binomiale con la definizione di *insieme delle parti* di A, usando le combinazioni: notiamo infatti che il numero di tutti i sottoinsiemi corrisponde a

- L'insieme vuoto
- Tutti i sottoinsiemi di cardinaltà 1
- Tutti i sottoinsiemi di cardinaltà 2
- ...
- Tutti i sottoinsiemi di cardinaltà $n-1$
- L'insieme stesso A

Sappiamo ora calcolare questo valore, in quanto il numero di k-insiemi é dato proprio dal numero di combinazioni di k elementi di A. Possiamo quindi dire che il numero di sottoinsiemi di A ṗari a:

$$\sum_{k=0}^{n} \binom{n}{k}$$

Questo ci consente di calcolare il numero, ma non è pienamente soddisfacente, in quanto la formula non è di immediata valutazione. Avendo visto la biiezione dell'insieme delle parti di A con il numero di sequenze possibili con $|A|$ bit, possiamo concludere il seguente risultato.

Proposizione 6.2.15.

$$\sum_{k=0}^{n} \binom{n}{k} = 2^n$$

Tornando alla regola di biiezione, essa agisce come una lente d'ingrandimento sulla capacità di contare: se uno determina la dimensione di un insieme, allora può determinare immediatamente le dimensioni di molti altri insiemi attraverso biiezioni.

Esempio 6.2.16. *Consideriamo i due seguenti insiemi:*

(A) *tutti i modi di selezionare una dozzina di ciambelle tra 5 varietà disponibili;*

(B) *tutte le sequenze di 16 bit con esattamente 4 uni.*

Un esempio di elemento di A è

$$\underbrace{00}_{cioccolato}\ \underbrace{\ }_{limone}\ \underbrace{000000}_{zucchero}\ \underbrace{00}_{fragola}\ \underbrace{00}_{semplice}$$

Di sopra abbiamo disegnato ogni ciambella con uno 0 *ed abbiamo lasciato uno spazio vuoto tra ogni diversa varietà. Quindi la selezione di sopra consiste di due ciambella al cioccolato, nessuna ciambella al limone, 6 ciambelle con lo zucchero, due alla fragola e due semplici. Adesso inseriamo un* 1 *in ognuno dei quattro spazi*

$$\underbrace{00}_{cioccolato}\ 1\ \underbrace{\ }_{limone}\ 1\ \underbrace{000000}_{zucchero}\ 1\ \underbrace{00}_{fragola}\ 1\ \underbrace{00}_{semplice}$$

ed otteniamo un elemento di B

0011000000100100

Infatti questa è una sequenza di 16 bit in cui 1 *appare esattamente* 4 *volte. Questo esempio suggerisce una biiezione dall'insieme A all'insieme B: una dozzina di ciambelle che consiste di sequenza di*

c cioccolato, l limone, z zucchero, f fragola, s semplici

viene mappata nella sequenza

$$\underbrace{0\ldots0}_{c}1\underbrace{0\ldots0}_{l}1\underbrace{0\ldots0}_{z}1\underbrace{0\ldots0}_{f}1\underbrace{0\ldots0}_{s}$$

La sequenza risultante ha sempre 16 *bits e esattamente* 4 *uni, ed è quindi un elemento di* B*. Inoltre questa funzione è una biiezione: ogni sequenza di bits in* B *viene esattamente da un solo ordine di dodici ciambelle. Quindi* $A \cong B$ *e, grazie alla regola di biiezione,* $|A| = |B|$*.*

Più in generale vale che il numero di modi di selezionare z *ciambelle quando sono disponibili* g *gusti è lo stesso del numero di sequenze con esattaente* z *zeri e* $g-1$ *uni. Ponendo* $n = z+g-1$ *e* $k = g-1$*, abbiamo così che tale numero è dato da* $\binom{z+g-1}{g-1}$*.*

Osserviamo, tra l'altro, che tale formula si applica anche al problema di ripartire z *elementi in* g *insiemi: ad esempio, tutti i modi possibili di riporre* z *magliette in* g *cassetti.*

Permutazioni

Quando si va in pizzeria con gli amici, ci chiediamo sempre "chi si siede vicino a chi?", o, in altre parole, "in che ordine ci sediamo intorno al tavolo?". Dato un insieme A, una *permutazione* di A è un ordinamento dei suoi elementi. Le permutazioni ci consentono di studiare problemi dove ci chiediamo, ad esempio, quanti modi diversi ci sono di ordinare un determinato insieme, come ad esempio il gruppo di amici in pizzeria.

Definizione 6.2.17. *Dato un insieme finito* A *con* $|A| = n$*, una permutazione di* A *è una sequenza ordinata* $a_1, \ldots, a_n$ *dove tutti e soli gli elementi di* A *appaiono esattamente una volta.*

Una domanda naturale è: in quanti modi diversi ci possiamo sedere in torno al tavolo? Ovvero, quante permutazioni ha l'insieme A?

Supponiamo di fissare una sedia a capotavola come punto di partenza (e assumiamo che il numero di sedie sia uguale al numero di persone): chiaramente ci sono n possibili scelte per chi si siederà in quel posto. Una volta seduto il primo amico, chiunque esso sia, rimarranno $n-1$ scelte per la seconda sedia, poi $n-3$ per la terza, e così via, fino ad arrivare all'ultima sedia libera dove avremo 1 sola scelta, ovvero si dovra sedere l'ultimo amico rimasto in piedi.

Esempio 6.2.18. *Sia il nostro gruppo composto da Anna, Bruno, Ciro e Daniela, rappresentati dall'insieme* $A = \{a, b, c, d\}$*: le permutazioni di* A *sono* 24*. È tedioso ma facile verificare la proprietà andando ad enumerarle una ad una:*

- *abcd*
- *abdc*
- *acbd*
- *acdb*
- *adbc*
- *adcb*
- *bacd*
- *badc*
- *bcad*
- *bcda*
- *bdac*
- *bdca*
- *cabd*
- *cada*
- *cbad*
- *cbda*
- *cdab*
- *cdba*
- *dabc*
- *dacb*
- *dbac*
- *dbca*
- *dcab*
- *dcba*

Possiamo impostare una regola ricorsiva come prima. Sia $P(n)$ il numero di permutazioni di A, dove $n = |A|$. Ognuno degli n elementi di A può occupare la prima posizione, lasciando le rimanenti $n-1$ agli altri elementi di A. Ora le rimanenti posizioni ospitano anch'esse una permutazione, ma questa volta con $n-1$ elementi. La clausola ricorsiva è quindi:

$$P(n) = n \cdot P(n-1)$$

dove il caso base è $P(1) = 1$ perché c'è una sola permutazione, avendo un solo elemento.

Proposizione 6.2.19.

$$P(n) = n \cdot (n-1) \cdot (n-1) \cdots 1 = n!$$

Dimostrazione. Per induzione su n. Caso base: $P(1) = 1! = 1$. Passo induttivo: $P(n) = n \cdot P(n-1) = n \cdot (n-1)! = n!$. ■

Disposizioni

Ammettiamo ora che il ristorante non possa mettere tutti gli n amici allo stesso tavolo, ma solamente k di loro, facendo sedere i rimanenti in un'altra sala. In quanti modi diversi possiamo riempire il tavolo con k posti? Chiamiamo questo concetto le *disposizioni* di n elementi in k posti, note anche come k-permutazioni di n.

Definizione 6.2.20. *Dato un insieme finito A con $|A| = n$ e un intero $k \leq n$, una disposizione degli elementi di A in k posti è una sequenza ordinata $a_1, \ldots, a_k$ dove esattamente k elementi di A appaiono esattamente una volta.*

Esempio 6.2.21. *Sia il nostro gruppo composto da Anna, Bruno, Ciro, Daniela, ed Enrico, rappresentati dall'insieme $A = \{a, b, c, d, e\}$, ed ammettiamo di avere un tavolo da 2 posti: ci sono 20 disposizioni di $n = 5$ elementi di A in $k = 2$ posti, come possiamo verificare:*

- ab
- ac
- ad
- ae
- ba
- bc
- bd
- be
- ca
- cb
- cd
- ce
- da
- db
- dc
- de
- ea
- eb
- ec
- ed

Possiamo ricondurre le disposizioni alle combinazioni. Ogni combinazione rappresenta una scelta di k elementi: le disposizioni richiedono di indicare ulteriormente le $k!$ permutazioni di quest'ultimi. Quindi il numero di disposizioni è

$$\binom{n}{k} \cdot k! = \frac{n!}{(n-k)!}$$

6.3 Sulle distanze in modo ricorsivo

Abbiamo visto la definizione di distanza nel Capitolo 4, nella Sezione 4.6. Possiamo definire la distanza $d()$ su un grafo connesso $G = (V, E)$ in modo ricorsivo:

- $d(x, y) = 1$ se $xy \in E$
- $d(x, y) = 1 + \min\{d(z, y) \mid z \in N(x)\}$

L'idea alla base è che un path minimo che parte da x verso $y \neq x$, deve necessariamente passare da un suo vicino in $N(x)$. Non sapendo quale sia questo vicino, consideriamo tutti i candidati in $z \in N(x)$. Essendo il grafo connesso, anche z avrà il suo path minimo verso y: se prendiamo il più breve tra tali cammini, otteniamo il path minimo da x a y aggiungendo l'arco xz a tale path da z a y. (In altre parole, l'idea di base è che se togliamo il primo arco di un path minimo, otteniamo ancora un path minimo.)

E' possibile calcolare la distanza di *tutti* i nodi di G da un nodo target t usando questa logica; Osserviamo che x è a distanza 1 da t *se e solo se* x e t sono vicini. Altrimenti, se x è a distanza $k > 1$ da t, x ha un vicino $y \in N(x)$ a distanza esattamente $k - 1$ da t.

Possiamo quindi calcolare le distanze applicando l'induzione "a ritroso":

- Applichiamo il caso base per conoscere *tutti* i nodi a distanza 1 da t.
- Se conosciamo tutti i nodi a distanza $\leq k$ da t, possiamo scoprire tutti i nodi a distanza $k + 1$ da t (sono tutti i nodi che non sono già a distanza $\leq k$ da t, ma hanno un vicino a distanza k da t).

6.4 Contare negli alberi

Sia T un albero con n nodi. Se $n = 1$, chiaramente ci sono zero archi. In generale, se $n \geq 2$ cosa succede?

Proposizione 6.4.1. *Un albero T con n nodi ha $n - 1$ archi.*

Dimostrazione. Dimostrazione per induzione su n.

Caso base: $n = 2$. In questo caso l'unica possibilità di collegare i due nodi è tramite $n - 1 = 1$ arco.

Passo induttivo: $n > 2$. Sia f una foglia qualunque di T: rimuovendo f, otteniamo un albero T' che ha un nodo f e un arco in meno rispetto a T. Inoltre, T' ha $n' = n = 1$ nodi: per ipotesi induttiva ha quindi $n' - 1 = n - 2$ archi. Deriviamo quindi cha T ha $(n - 2) + 1 = n - 1$ archi. ■

Prendiamo ora un albero binario T (che è quindi radicato). Abbiamo visto che è completo se ogni nodo interno ha entrambi i figli non vuoti, e che è pieno se è completo e le foglie sono tutte alla stessa distanza dalla radice. Prendiamo la sua altezza h e osserviamo la sua relazione con il numero f di foglie in T e il numero i di nodi interni (inclusa la radice): osserviamo che il numero totale di nodi è quindi $n = f + i$.

- Se l'altezza è $h = 0$, abbiamo $f = 1$ e $i = 0$.
- Se l'altezza è $h = 1$, abbiamo $f = 2$ e $i = 1$.
- Se l'altezza è $h = 2$, abbiamo $f = 4$ e $i = 3$.
- Se l'altezza è $h = 3$, abbiamo $f = 8$ e $i = 7$.
- ...

Proposizione 6.4.2. *Un albero pieno di altezza h ha 2^h foglie e $2^h - 1$ nodi interni, per un totale di $2^{h+1} - 1$ nodi.*

Dimostrazione. Per induzione sull'altezza h. Caso base $h = 0$, è immediato, come visto sopra. Per il passo induttivo, l'albero con altezza $h - 1$ ha $f' = 2^{h-1}$ foglie e $i' = 2^{h-1} - 1$ nodi interni. Passando a livello h, osserviamo che il numero di foglie raddoppia $f = 2 \cdot f' = 2^h$, e il numero di nodi interni è il numero totale di nodi dell'albero pieno di altezza $h-1$, cioè $i = f' + i' = 2^h - 1$. ■

La proposizione 6.4.2 indirettamente dice anche che vale la seguente uguaglianza, che poi non è altro che il numero naturale rappresentato in binario da h bit tutti a 1:

$$\sum_{i=0}^{h} 2^i = 2^{h+1} - 1$$

La proposizione 6.4.2 ci dice anche che $h = \log_2 f$ nell'albero pieno. Se prendiamo un albero T qualunque, non necessariamente pieno, abbiamo visto che la sua altezza è $h \leq n - 1$, e l'altezza massima $h = n - 1$ si ha quando ogni nodo interno ha un solo figlio e c'è solo una foglia. Qual è l'altezza minima rispetto al suo numero f di foglie? Vogliamo compattare al massimo i nodi di T verso la radice: anche se f non è più necessariamente una potenza del 2, esse sono disposte a distanza h e $h - 1$ (se invece sono 2^h sono tutte a distanza h perché l'albero è pieno, come abbiamo visto). Ne consegue che in generale

$$h \geq \log_2 f$$

Negli alberi k-ari, la base 2 viene sostituita dalla base k e quindi abbiamo $\log_k f$.

Come esempio di applicazione, negli algoritmi di ordinamento, verrà mostrato un albero di decisione che ha $f = n!$ foglie. L'altezza minima di tale albero è $\log_2 n!$ che cresce asintoticamente come $n \log n$ usando l'approssimazione di Stirling del fattoriale $n! \approx \sqrt{2\pi n} \left(\frac{n}{e}\right)^n$.

Da quanto visto finora, non c'è necessariamente un legame tra il numero di nodi interni e le foglie. Prendiamo un albero radicato T con f foglie. Un nodo interno u di T (inclusa la radice) si dice *unario* se u ha un solo figlio (si ricordi che le foglie hanno zero figli). Chiaramente un *non unario* ha almeno due figli. Se prendiamo una catena di nodi unari con una sola foglia, emerge che non c'è una relazione diretta con f. Invece quelli non unari sono strettamente correlati a f.

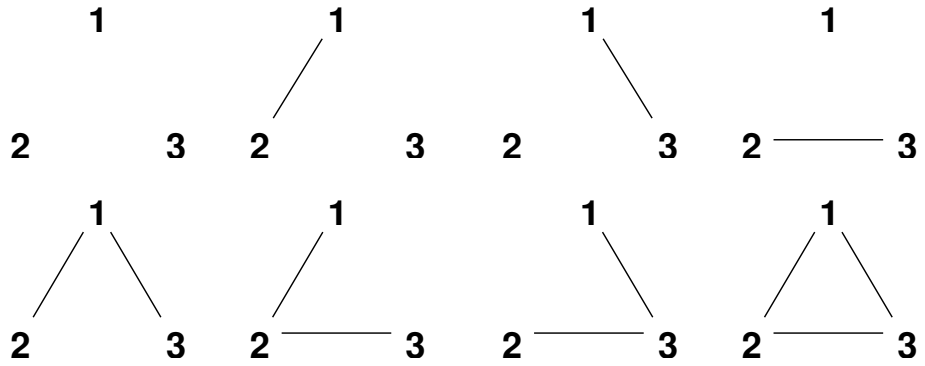


Figura 6.1: Grafi possibili su 3 nodi.

Proposizione 6.4.3. *Il numero di nodi non unari in T è al massimo $f-1$.*

Dimostrazione. Osserviamo che esiste sempre un nodo interno x in T i cui figli sono tutte foglie: per assurdo, se così non fosse, potremmo sempre scegliere un figlio che non è una foglia, rendendo di fatto T di cardinalità infinita. Abbiamo due casi su x usando l'induzione su f (lasciamo al lettore il caso base):

- Se x è unario, allora possiamo cancellare il suo unico figlio, che è una foglia, e x diventa una foglia; in questo modo, il numero f di foglie rimane inalterato ma c'è un nodo unario in meno (che mostra quindi che il numero di nodi unari può essere arbitrario rispetto a f).
- Se x è non unario, allora ha almeno $c \geq 2$ figli, che sono tutte foglie. Se le rimuoviamo, rendiamo x una foglia: nell'albero risultante T', c'è un nodo non unario in meno e $f' = f-c+1$ foglie (c sono rimosse e x diventa foglia). Poiché $c \geq 2$, abbiamo $f' \leq f-1$ foglie e quindi c'è sicuramente una foglia in meno. Possiamo applicare l'ipotesi induttiva su f' e concludere che T' ha al massimo $f'-1$ nodi non unari: di conseguenza T ne ha al massimo $f'-1+1 = f'$ (cioè ha x in più rispetto a T') e il risultato segue dal fatto che $f' \leq f-1$.

■

6.5 Contare nei grafi

I grafi ci aiutano a modellare moltissimi tipi di strutture, quindi contare grafi e strutture nei grafi ci da strumenti per poter contare queste strutture.

Una prima domanda naturale è la seguente:

Quanti grafi esistono?

Chiaramente, un numero illimitato se non fissiamo la dimensione. Quindi proviamo una formulare una domanda a cui sappiamo rispondere: quanti grafi esistono con n nodi?

Chiamiamo G_n l'insieme di grafi con n nodi, e quindi $|G_n|$ la sua cardinalità.

Se ad esempio prendiamo $n = 3$, possiamo vedere che esistono 8 grafi diversi:

L'intuizione è che ogni grafo consiste nel prendere un sottoinsieme degli archi possibili, ovvero un insieme $E \subseteq V \times V$. Nei grafi *non* orientati, abbiamo alcune restrizioni: non possiamo scegliere il cappio, e per ogni coppia di nodi x, y gli archi xy e yx sono in realtà lo stesso arco. Il numero di archi possibili è quindi $m_{max} = \frac{n(n-1)}{2}$.

Per ogni $i \in \{0, \ldots, m_{max}\}$, sappiamo che il numero di grafi su n nodi con i archi corrisponde al numero di sottoinsiemi di archi grandi i, ovvero $\binom{m_{max}}{i}$. Il numero totale di grafi che posso costruire è quindi:

$$|G_n| = \sum_{i \in \{0,\ldots,m_{max}\}} \binom{m_{max}}{i}$$

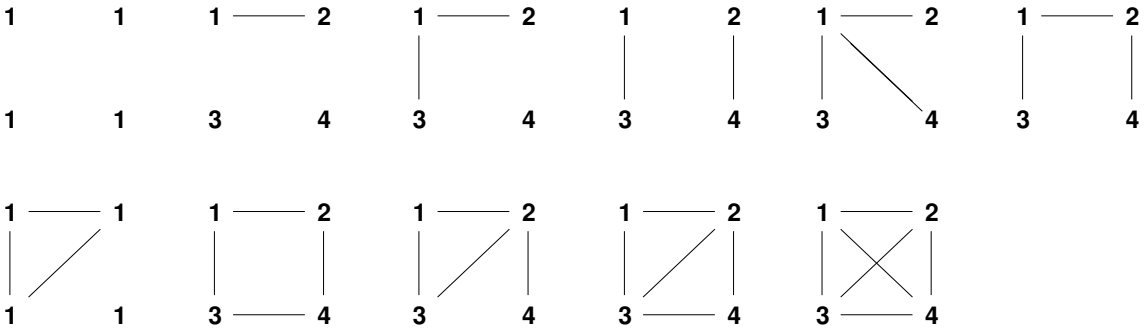


Figura 6.2: 11 tipi di grafi possibili su 4 nodi. Esistono $2^{\frac{n(n-1)}{2}} = 64$ grafi, ma ognuno è isomorfo a uno di questi 11.

Tuttavia esiste un metodo alternativo che ci consente di arrivare ad una soluzione più semplice:

Ricordiamo che l'insieme di tutti i sottoinsiemi di un insieme di cardinalità k è noto come l'*insieme delle parti* $\mathcal{P}(k)$. In questo caso ci stiamo chiedendo esattamente la cardinalità dell'insieme delle parti di m_{max}.

La cardinalità di $\mathcal{P}(k)$, ovvero il numero di sottoinsiemi, corrisponde a tutti i modi di scegliere quali elementi includere: abbiamo 2 scelte per il primo elemento (incluso/non incluso), 2 scelte per il secondo, 2 scelte per il terzo... Dato che queste scelte sono indipendenti, abbiamo 2^k possibili combinazioni.

Otteniamo quindi

$$\mathcal{P}(m_{max}) = \sum_{i \in \{0,\dots,m_{max}\}} \binom{m_{max}}{i} = 2^{m_m ax} = 2^{\frac{n(n-1)}{2}}$$

Come conseguenza, possiamo osservare anche le due seguenti proprietà del coefficiente binomiale:

$$\sum_{i \in \{0,\dots,k\}} \binom{k}{i} = 2^k, \text{ e quindi } \forall_{k>0, i \geq 0} \binom{k}{i} < 2^k$$

Tuttavia, è bene osservare che alcuni dei grafi che generiamo in questo modo sono *isomorfi*, ovvero hanno "la stessa forma": se non ci interessa la mappatura dei nodi, osserviamo in Figura 6.1 che esistono solo 4 "tipi" di grafi diversi con 3 nodi (il grafo vuoto, il grafo con 1 arco, il grafo con 2 archi adiacenti, e il triangolo). Se saliamo a $n = 4$, sappiamo immediatamente dire che esistono $2^{\frac{n(n-1)}{2}} = 64$ grafi, tuttavia disegnandoli possiamo scoprire che esistono solo 11 tipi diversi.

Quanti "tipi" di grafo esistono su n nodi è una domanda più complessa, alla quale non sappiamo rispondere con una formula chiusa. Tuttavia è possibile scrivere algoritmi per calcolare questi numeri, se siamo abbastanza pazienti da aspettare il risultato (e possiamo trovare i primi valori qui nella *Online Encyclopedia of Integer Sequences* https://oeis.org/A000088)

Quanti grafi orientati esistono?

Il problema diventa leggermente più semplice sui grafi orientati: qui posso tracciare un arco distinto da ogni nodo verso ogni nodo (incluso se stesso). Abbiamo quindi esattamente $|V \times V| = n^2$ possibili archi. Ne consegue che avremo esattamente 2^{n^2} possibili grafi orientati su n nodi.

Teorema 6.5.1. *Esistono $2^{\frac{n(n-1)}{2}}$ grafi non orientati su n nodi, e 2^{n^2} grafi orientati su n nodi.*

Osserviamo anche come lo stesso ragionamento si può ricavare dalle matrici di adiacenza:

Observation 6.5.2. *Nella matrice di adiacenza di un grafo orientato ognuna delle n^2 celle può valere 0 o 1, abbiamo quindi 2^{n^2} possibili celle con cui generare 2^{n^2} possibili matrici di adiacenza.*

In un grafo non orientato, invece, le n celle della diagonale valgono 0 in quanto non sono ammessi cappi, e delle rimanenti celle possiamo usarne solo metà, in quanto la matrice deve essere

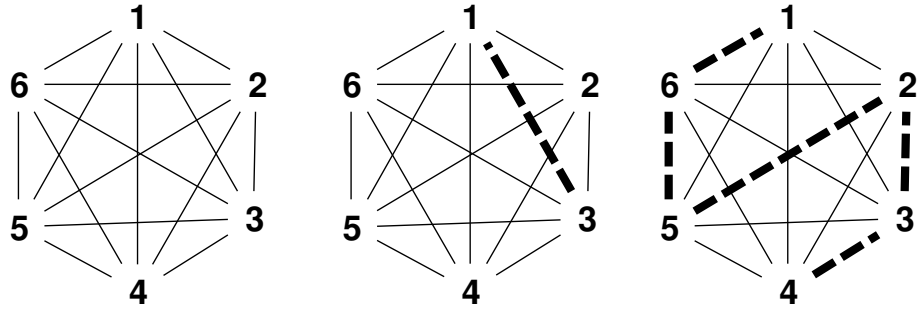


Figura 6.3: K_6, ovvero una cricca di 6 nodi (sinistra), lo shortest path tra 1 e 3 (centro), un path hamiltoniano (destra)

simmetrica rispetto alla diagonale principale. Abbiamo quindi $\frac{n^2-n}{2} = \frac{n(n-1)}{2}$ *celle utili, con cui possiamo generare* $2^{\frac{n(n-1)}{2}}$ *possibili matrici di adiacenza.*

Complemento di un grafo

Dato un grafo $G = (V, E)$, possiamo definire il suo *complemento* come il grafo $H = (V, E')$ sugli stessi nodi, che ha tutti e soli gli archi che *mancano* in G.

Formalmente, $H = (V, E')$ è il complemento di $G = (V, E)$ se $E' = \{xy \in V \times V \mid xy \notin E\}$

Guardando la Figura 6.1, ad esempio, possiamo osservare come il primo grafo (con 0 archi) sia il complemento dell'ultimo (con 3 archi), mentre ognuno dei grafi con 1 arco è il complemento di uno dei grafi con 2 archi.

Osserviamo anche che la relazione è simmetrica: se H è il complemento di G, allora G è il complemento di H.

Questa osservazione ci porta a una regola più generale:

Observation 6.5.3. *Ogni grafo* $G \in G_n$ *ha esattamente un complemento* $H \in G_n$*, e la relazione complemento* $C \subseteq G_n \times G_n$ *è una biiezione. Osserviamo inoltre che* $C^{-1} = C$*.*

Infine, se abbiamo $\frac{n(n-1)}{2}$ possibili archi, tutti i grafi complemento di un grafo G con m archi avranno esattamente $\frac{n(n-1)}{2} - m$ archi (tutti gli archi possibili, meno gli m di G). Per la stessa ragione, tutti i grafi con $\frac{n(n-1)}{2} - m$ archi hanno come complemento un grafo di m archi. Possiamo quindi concludere che il numero di grafi con n nodi e m archi è sempre uguale al numero di grafi con n nodi e $\frac{n(n-1)}{2} - m$ archi.

6.6 Contare cammini in grafi notevoli

Vediamo in questa sezione domande del tipo "quanti cammini di un certo tipo ci sono in un grafo?", e come possiamo utilizzare le tecniche che abbiamo visto in precedenza per rispondervi.

Cammini in una cricca

Definizione 6.6.1. *Una* ***cricca*** *di* n *nodi è un grafo dove ogni coppia di nodi è connessa. La cricca viene anche denotata come* K_n*, e detta grafo completo,*

Una prima domanda che possiamo chiederci è:

Esempio 6.6.2. *Quanti shortest paths ci sono tra due nodi distinti* x *e* y *in una cricca?*

La risposta è 1*, dato che se esistono tutti gli archi esiste sempre il cammino* xy *di lunghezza* 1*. Qualsiasi altro cammino avrà lunghezza maggiore.*

Ricordiamo adesso la definizione di path hamiltoniano: questo è un path che tocca *tutti* i nodi del grafo esattamente una volta. Sicuramente esiste almeno un path hamiltoniano in una cricca (se ne vede uno in figura), ma quanti ne esistono?

Esempio 6.6.3. *In una cricca, qualsiasi sequenza di nodi $v_1, v_2, \ldots$ senza ripetizioni corrisponde a un path, in quanto per ogni coppia v_i, v_j esiste sempre l'arco v_i, v_j.*

Ne consegue che qualsiasi permutazione dei nodi $1, 2, \ldots, n$ corrisponde ad un path hamiltoniano, dato che ogni nodo appare esattamente una volta nella sequenza. Possiamo convincerci di questo scrivendo una permutazione qualsiasi, e provando a seguire il corrispondente cammino nel grafo in figura.

Quindi K_n ha esattamente $n!$ path hamiltoniani.

Un concetto simile al path hamiltoniano è quello di *ciclo* hamiltoniano: questo è un ciclo che tocca ogni nodo del grafo esattamente una volta.

Quanti cicli hamiltoniani esistono?

Esempio 6.6.4. *Se guardiamo un path hamiltoniano che va da i a j, notiamo che possiamo sempre aggiungere l'arco ij e trasformare il path in un ciclo hamiltoniano.*

Ne consegue che K_n ha $n!$ cicli hamiltoniani.

Tuttavia, osserviamo che alcuni cicli sono sostanzialmente equivalenti: il ciclo $1, 6, 5, 2, 3, 4, 1$ è composto dagli stessi archi del ciclo "opposto" $1, 4, 3, 2, 5, 6, 1$.

Poniamoci allora la domanda: quanti cicli hamiltoniani *non equivalenti* esistono in K_n?

Esempio 6.6.5. *Abbiamo osservato come un ciclo e il suo simmetrico ottenuto percorrendo gli archi in senso opposto sono composti dagli stessi archi, quindi equivalenti.*

Proviamo ora a percorrere un ciclo due volte: $1, 6, 5, 2, 3, 4, 1, 6, 5, 2, 3, 4, 1$ notiamo che se iniziamo da 1 otteniamo il ciclo $1, 6, 5, 2, 3, 4, 1$, ma iniziando da un qualsiasi altro dei primi n nodi otteniamo una nuovo ciclo hamiltoniano, che usa tuttavia gli stessi archi: ad esempio $2, 3, 4, 1, 6, 5, 2$ partendo da 2 o $4, 1, 6, 5, 2, 3, 4, 1$ partendo da 4. Possiamo chiamare questi cicli rotazioni del ciclo originale.

Per ogni ciclo, quindi, sappiamo che esistono un totale di $2n$ cicli hamiltoniani equivalenti ad esso: n rotazioni e per ognuna il suo simmetrico.

Ne concludiamo che K_n ha $\frac{n!}{2n}$ cicli hamiltoniani non equivalenti.

Quanti path di k nodi esistono in K_n?

Esempio 6.6.6. *Proviamo a costruire un cammino di k nodi: abbiamo n scelte per il primo nodo, poi $n-1$ per chi sarà il secondo, $n-2$ per il terzo, e così via fino al k-esimo per cui avremo $n-(k-1)$ scelte.*

Il numero di cammini di k nodi in K_n è quindi:

$$(n)(n-1)\cdots(n-k+1) = \frac{(n)(n-1)\cdots(1)}{(n-k)\cdots(1)} = \frac{n!}{(n-k)!}$$

Notiamo anche che qualsiasi sequenza di k nodi distinti corrisponde a un valido cammino, e che quindi il numero di cammini di k nodi in K_n corrisponde al numero di possibili sequenze ordinate di lunghezza k (ovvero k-permutazioni) di n elementi.

Possiamo osservare che la formula qui sopra ci porta a una dimostrazione costruttiva della cosiddetta formula dei tre fattoriali: Se ci chiediamo quanti sono i *sottoinsiemi* di dimensione k di n elementi (denotato dal coefficiente binomiale $\binom{n}{k}$), notiamo che ogni sequenza di lunghezza k corrisponde ad un sottoinsieme, tuttavia, esistono esattamente $k!$ permutazioni di ogni sottoinsieme, corrispondenti a $k!$ sequenze che corrispondono allo stesso sottoinsieme.

Ne consegue che il numero di sequenze di lunghezza k è pari a $k!$ volte il numero di sottoinsiemi di dimensione k. Quindi che $\binom{n}{k} = \frac{n!}{k!(n-k)!}$.

Cammini nel grafo bipartito completo

Il grafo $K_{n,n}$ denota il grafo *bipartito completo* con n nodi su ogni partizione.

Definizione 6.6.7. *Un grafo $G = (V, E)$ è bipartito se sono vere le seguenti condizioni (tutte equivalenti tra loro):*

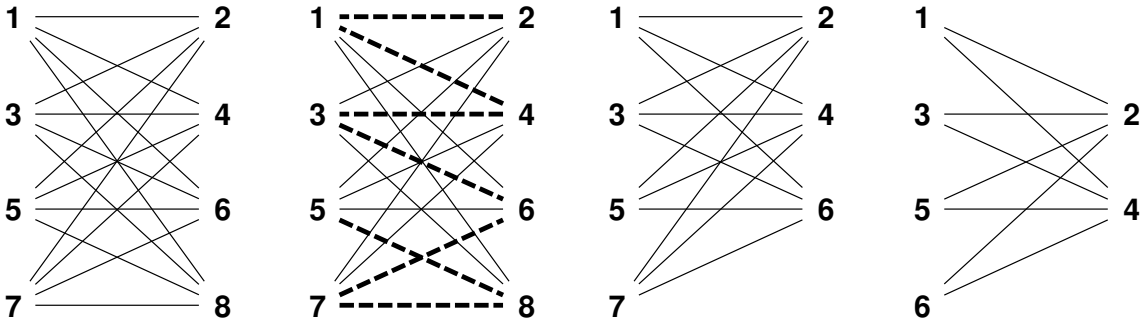


Figura 6.4: Esempi di grafi bipartiti completi. In ordine: Il grafo $K_{4,4}$, un suo path hamiltoniano, il grafo $K_{4,3}$, il grafo $K_{4,2}$.

- *Esiste una partizione V_1, V_2 di V, i.e., con $V_1 \cap V_2 = \emptyset$ e $v_1 \cup v_2 = V$, tale che non esistono archi tra nodi della stessa partizione. Ovvero, per ogni arco $xy \in E$ vale $x \in V_1 \implies y \in V_2$.*
- *Dati due colori, è possibile colorare ogni nodo di G di un colore in modo che i due estremi di ogni arco abbiano sempre colore diverso (chiamata 2-colorazione).*
- *G non contiene cicli di lunghezza dispari.*

Observation 6.6.8. *Gli alberi sono grafi che non hanno cicli. Ne consegue che un albero non ha cicli dispari, e quindi tutti gli alberi sono grafi bipartiti. E' in effetti facile trovare una 2-colorazione di un albero se coloriamo la radice del primo colore, poi i suoi figli del secondo, e procediamo a colorare ogni livello alternando tra i due colori.*

E' anche possibile dimostrare che un grafo bipartito, se connesso, può essere partizionato in un solo modo.

Esercizio 6.6.9. *Dato un grafo bipartito connesso, dimostrare che esiste una sola bipartizione v_1, V_2.*[1]

Un grafo bipartito *completo* è un grafo dove, data la partizione V_1, V_2, ogni nodo di V_1 è adiacente a *tutti* i nodi di V_2 (e chiaramente viceversa).

Chiediamoci anche qui, quanti shortest path esistono tra due nodi?

Esempio 6.6.10. *Dati $x, y \in K_{n,n}$, esiste un solo shortest path tra x e y se $x \in V_1$ e $y \in V_2$, dato che esiste l'arco xy. Se invece $x, y \in V_1$, allora esistono n shortest path tra x e y: per ogni $z \in V_2$, xz, zy è uno shortest path tra x e y.*

Ragioniamo ora sui path hamiltoniani:

Esempio 6.6.11. *In qualsiasi cammino su $K_{n,n}$ a ogni passo devo alternare tra V_1 e V_2: Assumiamo di partire da V_1; il cammino sarà della forma $a_1, b_1, a_2, b_2, a_3, b_3, \ldots, a_n, b_n$, dove tutti gli a_i appartengono a V_1 e tutti i b_i appartengono a V_2. Tuttavia, osserviamo che per qualsiasi coppia a_i, b_j, l'arco $a_i b_j$ esiste e può essere percorso nel path.*

Possiamo quindi prendere qualsiasi permutazione di V_1, qualsiasi permutazione di V_2, e alternare i loro nodi uno ad uno per ottenere un valido path hamiltoniano. Visto che V_1 e v_2 anno entrambi dimensione n, ognuno ammette $n!$ permutazioni. Inoltre, abbiamo assunto di partire da V_1: otteniamo altrettanti path hamiltoniani se partiamo invece da V_2, guadagnando quindi un ulteriore fattore 2.

Il grafo $K_{n,n}$ ha quindi $2(n!)^2$ path hamiltoniani.

Possiamo ragionare similarmente al caso delle cricche per trovare invece i *cicli* hamiltoniani:

Esempio 6.6.12. *Ogni path hamiltoniano in $K_{n,n}$ che inizia da $a_1 \in V_1$ finisce in $b_n \in V_2$, o viceversa. E' quindi sempre disponibile l'arco $a_1 b_n$ (o $b_1 a_n$) che collega il primo e ultimo nodo, trasformando il path hamiltoniano in un ciclo hamiltoniano.*

[1] Suggerimento: osserviamo la lunghezza dei cammini tra due nodi.

Il grafo $K_{n,n}$ ha quindi $2(n!)^2$ cicli hamiltoniani.

Anche in questo caso, alcuni cicli sono equivalenti, ovvero composti dagli stessi archi: ogni ciclo ha lunghezza $2n$ e quindi lo troveremo in $2n$ rotazioni (vedi Esempio 6.6.5), e ognuna di queste avrà un suo ciclo simmetrico.

Il grafo $K_{n,n}$ ha quindi $\frac{2(n!)^2}{2\cdot 2n} = \frac{n!(n-1)!}{2}$ cicli hamiltoniani non equivalenti.

Guardiamo ora grafi con un numero di nodi diverso tra le due partizioni.

Esempio 6.6.13. *In Figura 6.4 possiamo vedere il grafo $K_{4,3}$. E' possibile costruire un path hamiltoniano procedendo come per $K_{n,n}$. Tuttavia, notiamo che possiamo farlo solo partendo dal lato destro: possiamo intercalare 4 elementi $a_i \in V_1$ con 3 elementi $b_i \in V_2$ solo se iniziamo e terminiamo con a: $a_1, b_1, a_2, b_2, a_3, b_3, a_4$.*

Otteniamo quindi tutte le possibili permutazioni di 4, ovvero 4!, intercalate con le permutazioni di 3, ovvero 3!, ma perdiamo il fattore 2 rispetto a $K_{n,n}$ in quanto non abbiamo la scelta del lato iniziale.

Il grafo $K_{n,n-1}$ ha quindi $n!(n-1)!$ path hamiltoniani.

Riusciamo, utilizzando la stessa logica dell'Esempio 6.6.12, a costruire cicli hamiltoniani in $K_{4,3}$, e in generale, del grafo $K_{n,(n-1)}$?

Esercizio 6.6.14. *Determinare il numero di cicli hamiltoniani nel grafo $K_{4,3}$ e $K_{n,n-1}$.*

Riusciamo a costruire path hamiltoniani nel grafo $K_{4,2}$ (in Figura 6.4), e in generale, $K_{n,(n-2)}$?

Esercizio 6.6.15. *Determinare il numero di path hamiltoniani nel grafo $K_{n,(n-2)}$.*

CAPITOLO 7

Linguaggi Formali

Ogni linguaggio scritto, sia questo un linguaggio naturale come l'italiano o l'inglese, un linguaggio di programmazione o il linguaggio logico che abbiamo utilizzato talvolta nei capitoli precedenti e che vedremo più in dettaglio nel prossimo capitolo, coinvolge due aspetti che è importante distinguere: la *sintassi* e la *semantica*. La sintassi ha a che fare con la struttura (o la forma) delle frasi esprimibili. La semantica, invece, ha a che fare con il significato delle frasi esprimibili. Se consideriamo per un momento il linguaggio italiano con cui comunichiamo con i nostri simili, due frasi come "la mela mangia il bambino" e "il bambino mangia la mela" obbediscono entrambe alla sintassi dell'italiano, in quanto sono costruite secondo lo schema <soggetto> <verbo> <complemento-oggetto> a partire da stringhe costruite correttamente. D'altra parte, solo la seconda ha un significato, o semantica, ragionevole.

In questo capitolo ci concentreremo sul primo aspetto, la descrizione della sintassi dei linguaggi. Il punto di partenza del nostro studio è la concezione del termine *linguaggio* come sinonimo di insieme di frasi (sintatticamente) ammissibili. In altre parole, una volta fissato un *alfabeto* di elementi di base, detti anche *simboli*, un linguaggio non sarà altro che un sottoinsieme di tutte le stringhe (cioè sequenze di lunghezza arbitraria) di simboli.

Si pensi per esempio al linguaggio delle espressioni dell'aritmetica, ottenibili a partire dai numeri e dalle quattro operazioni $+$, $-$, $\times$ e $\div$. Fra tutte le possibili sequenze (o stringhe) di numeri ed operazioni aritmetiche, ve ne sono di ammissibili, come $3\times4+2$, e di non ammissibili, come $3+\times+4$. Quindi, il linguaggio delle espressioni aritmetiche può essere identificato con il sottoinsieme delle stringhe ammissibili.

In modo del tutto analogo, un linguaggio di programmazione può essere identificato con l'insieme delle proprie stringhe ammissibili, che comunemente chiamiamo programmi. Secondo questo punto di vista, descrivere un linguaggio significa sostanzialmente descrivere l'insieme delle stringhe del linguaggio stesso, ovvero avere un metodo per:

1. decidere quali stringhe fanno parte di tale insieme, e quali non ne fanno parte, oppure

2. costruire tale insieme, enumerando le stringhe che lo compongono.

Il problema insomma è: *come identificare l'insieme delle stringhe ammissibili, che caratterizza un linguaggio?* Esistono due approcci principali a questo problema. Il primo si basa su uno strumento, detto *automa*, che è in grado di riconoscere (o accettare) tutte e sole le stringhe che fanno parte di un linguaggio. Il secondo si basa su uno strumento, detto *grammatica*, che è in grado di generare (o costruire) tutte e sole le stringhe che fanno parte di un linguaggio.

Molti studi sono stati dedicati alla definizione e al riconoscimento dei linguaggi. La teoria che è stata sviluppata è conosciuta con il nome di *teoria degli automi* o *teoria dei linguaggi formali* e le sue definizioni di base e tecniche fanno parte del nucleo dell'informatica.

Questo capitolo intende fornire una guida rapida alle idee e ai risultati principali riguardanti gli automi e le grammatiche, al fine di comprendere i metodi con cui si descrive la sintassi dei linguaggi formali, siano essi linguaggi di programmazione o il linguaggio logico definito nel prossimo capitolo[1].

[1] Alcune parti di questo capitolo sono largamente ispirate dalle dispense "Elementi di Sintassi dei Linguaggi di Programmazione" di Barbuti, Mancarella, Pedreschi, Turini.

7.1 Alfabeti, Parole e Linguaggi

Alla luce delle considerazioni fatte, formalizziamo il concetto di linguaggio. Per cominciare abbiamo bisogno di fissare un alfabero.

Definizione 7.1.1. *Un* alfabeto *è un insieme finito. I suoi elementi vengono detti* simboli.

Esempi tipici di alfabeti, con cui il lettore avrà un po' di familiarità sono l'alfabeto Latino $\{a, b, c \dots z\}$, alfabeto Latino Maiuscolo $\{A, B, C \dots Z\}$ e l'alfabeto Greco $\{\alpha, \beta, \gamma, \dots, \omega\}$. Oltre agli alfabeti delle lingue naturali, possiamo considerare anche l'alfabeto delle cifre decimali $\{0, 1, \dots, 9\}$, delle cifre esadecimali $\{0, 1, \dots, 9, A, B, \dots, F\}$, delle cifre binarie $\{0, 1\}$, o l'alfabeto per i numeri romani $\{I, V, X, L, C, D, M\}$.

Il passo successivo consiste nel definire una parola su A semplicemente come una stringa su A. Abbiamo già visto la definzione di stringa (come sequenza di lunghezza arbitraria) nella Sezione 2.7. Per comodità del lettore, la riportiamo di seguito, adeguata a questo nuovo contesto.

Definizione 7.1.2. *Dato un alfabeto A, una* stringa *(o* parola*) su A è una sequenza (di lunghezza arbitraria) $a_0 \dots a_n$ tale che ogni a_i è un simbolo dell'alfabeto A (cioè $a_i \in A$) ed n è un arbitrario numero naturale.*[2] *La* lunghezza *di una stringa $a_0 \dots a_n$ è $n+1$. Esiste una sola stringa di lunghezza* 0 *che viene detta* stringa vuota *e denotata da ε*[3]*. L'insieme di tutte le stringhe su A è denotato da $A^\star$.*

Sull'alfabeto Latino sono stringhe per esempio *pippo*, *bob*, *anna*, *hrjk*, ε, *iiiiiii*. Su quello Latino Maiuscolo sono stringhe $PIPPO$, $BOB, ANNA, HRJK$, ε, $IIIIIII$. Sull' alfabeto delle cifre decimali: 1981, 27, 10, 500, 0010. Sull'alfabeto delle cifre esadecimali $F2$, $23A$, $A55C$, $3A8$. Sull'alfabeto delle cifre binarie 10, 101, 1101, 111111. Sull'Alfabeto dei numeri romani I, IV, XII, $IIII$, VX.

Nella Definizione 5.3.1, abbiamo dato una definizione induttiva di $A^\star$. Di seguito riportiamo una diversa (ma equivalente) definizione induttiva che ci sarà molto utile nel seguito.

Definizione 7.1.3. *Sia A un alfabeto. L'insieme $A^\star$ delle* STRINGHE *su A è il più piccolo insieme che soddisfa:*

1. [CLAUSOLA BASE] $\varepsilon \in A^\star$*, dove ε è la* stringa vuota.
2. [CLAUSOLA INDUTTIVA] *Se $w \in A^\star$ e $a \in A$ allora $aw \in A^\star$.*

La definizione precedente si distingue dalla Definizione 5.3.1, solamente nella clausola induttiva dove si conclude che $aw \in A^\star$, invece di concludere (come nella Definizione 5.3.1) che $wa \in A^\star$.

Nel Capitolo 5, abbiamo visto che le stringhe possono essere concatenate (Definizione 5.3.4). Riportiamo qua sotto una definizione non induttiva e abbastanza intuitiva di concatenazione.

Definizione 7.1.4. *La* concatenazione di stringhe *è una funzione $\_ \cdot \_ \colon A^\star \times A^\star \to A^\star$ definita per tutte le stringhe $u, v \in A^\star$ come*

$$u \cdot v = uv.$$

Per esempio se l'alfabeto è quello Latino si ha che

$$anna \cdot rella = annarella \qquad man \cdot gio = mangio \qquad hrj \cdot tyr = hrjtyr$$

Se invece consideriamo l'alfabeto Latino Maiuscolo

$$ANNA \cdot RELLA = ANNARELLA \qquad ANNA \cdot \varepsilon = ANNA$$

O per l'alfabero delle cifre decimali

$$19 \cdot 81 = 1981$$

Si noti che in alcuni linguaggi di programmazione come Java e Javascript, l'operatore $\cdot$ è denotato da $+$.

Possiamo finalmente dare una definizione del concetto di linguaggio.

[2] Useremo di solito *parola* invece di *stringa* quando la sequenza di caratteri è intesa essere una parola di un linguaggio naturale.

[3] Assumiamo che $\varepsilon \notin A$.

Definizione 7.1.5. *Dato un alfabeto A, un* linguaggio su A *è un insieme $L \subseteq A^\star$. L'insieme di tutti i linguaggi è denotato con $\mathcal{P}(A^\star)$.*

Per esempio, il vocabolario italiano è un linguaggio sull'alfabeto Latino $\{a, b, c \dots z\}$. L'Italiano e l'Inglese sono linguaggi sull'alfabeto che contiene i caratteri latini maiuscoli, minuscoli, spazi e punteggiatura.

L'insieme delle espressioni aritmetiche è un linguaggio su $\{0, 1, 2 \dots 9, +\times, -, \div\}$. Si noti che non tutte le stringhe su tale alfabeto sono espressioni aritmetiche: per esempio $5 + 4$ è una espressione aritmetica, ma $5 + \times 4 \div$ non lo è.

In modo del tutto analogo l'insieme dei numeri romani è un linguaggio su $\{I, V, X, L, C, D, M\}$. Si noti che non tutte le stringhe su tale alfabeto sono numeri romani: per esempio VX non è un numero romano.

Secondo questa visione, la maggior parte dei *linguaggi di programmazione* sono essenzialmente sottoinsiemi di `ASCII`*, dove `ASCII` è l'alfabeto dei caratteri alfabetici, numeri e di interpunzione presenti, secondo uno standard internazionale, sulla tastiera alfanumerica di ogni computer.[4] In altre parole, un programma non è altro che una stringa di caratteri alfanumerici `ASCII`. Ovviamente, non tutte le stringhe siffatte sono programmi accettabili, così come i diversi linguaggi di programmazione corrispondono a diversi insiemi di tali stringhe. Ogni linguaggio di programmazione ha la propria sintassi, ovvero le proprie regole che descrivono la struttura delle stringhe `ASCII` che corrispondono a programmi ammissibili, o sintatticamente ben formati.

Le Sezioni 7.2 e 7.3 sono dedicate alla presentazione di due metodi, gli automi e le grammatiche, rivolti proprio a descrivere la struttura sintattica dei linguaggi. Prima di far ciò, è interessante però studiare la struttura algebrica dell'insieme di tutti i linguaggi $\mathcal{P}(A^\star)$.

Operazioni su linguaggi

I linguaggi su uno stesso alfabeto A possono essere combinati, in modi molto simili a come vengono combinate le relazioni su uno stesso insieme. Iniziamo illustrando nel seguente esempio, tre linguaggi "notevoli".

Esempio 7.1.6. *Per un qualsiasi alfabeto A, esistono sempre i seguenti linguaggi:*

- *Il linguaggio vuoto $\varnothing = \{\}$,*
- *Il linguaggio che contiene solo la stringa vuota $\{\varepsilon\}$,*
- *Il linguaggio completo $A^\star$.*

È interessante notare l'analogia con le relazioni su A in cui si ha la relazione vuota ($\varnothing_{A,A}$), la relazione identità (id_A) e la relazione completa ($A \times A$).

Come le relazioni, i linguaggi possono essere combinati, utilizzando le solite operazioni insiemistiche:

- Unione: $L_1 \cup L_2 = \{w \in A^\star \mid w \in L_1 \text{ oppure } w \in L_2\}$;
- Intersezione: $L_1 \cap L_2 = \{w \in A^\star \mid w \in L_1 \text{ e } w \in L_2\}$;
- Differenza: $L_1 \setminus L_2 = \{w \in A^\star \mid w \in L_1 \text{ e } w \notin L_2\}$;
- Complemento: $\overline{L} = \{w \in A^\star \mid w \notin L\}$.

Si noti che nel complemento l'insieme universo $\mathcal{U}$ è inteso essere il linguaggio completo $A^\star$.

Utilizzando la concatenazione di stringhe, si possono concatenare i linguaggi.

Definizione 7.1.7. *La funzione $\_ \cdot \_ \colon \mathcal{P}(A^\star) \times \mathcal{P}(A^\star) \to \mathcal{P}(A^\star)$ è definita per tutti i linguaggi $L_1, L_2 \in \mathcal{P}(A^\star)$ come*

$$L_1 \cdot L_2 = \{w \in A^\star \mid \text{ esistono } w_1 \in L_1, w_2 \in L_2 \text{ tali che } w = w_1 \cdot w_2\}.$$

[4] Alcuni linguaggi usano altri standard per i caratteri, come UNICODE, ma il discorso non cambia.

Esempio 7.1.8. *Mostriamo alcuni esempi di concatenazione di linguaggi senza dichiarare esplicitamente l'alfabeto.*

- *Sia* $L_1 = \{anna, so\}$ *e* $L_2 = \{rella\}$, $L_1 \cdot L_2 = \{annarella, sorella\}$;
- *Sia* $L_1 = \{anna, so\}$ *e* $L_2 = \{rella, le\}$, $L_1 \cdot L_2 = \{annarella, sorella, annale, sole\}$;
- *Sia* $L_1 = \{2, 4, 8\}$ *e* $L_2 = \{3, 5, 7\}$, $L_1 \cdot L_2 = \{23, 25, 27, 43, 45, 47, 83, 85, 87\}$;
- *Sia* $L_1 = \{\varepsilon, V, L, D\}$ *e* $L_2 = \{I, II, III\}$,

$$L_1 \cdot L_2 = \{I, II, III, VI, VII, VIII, LI, LII, LIII, DI, DII, DIII\};$$

- *Sia* $L_1 = \emptyset$ *e* $L_2 = \{anna, so\}$, $L_1 \cdot L_2 = \emptyset$.

Teorema 7.1.9. *Per tutti gli alfabeti* A *e per tutti i linguaggi* $L, L_1, L_2, L_3 \in \mathcal{P}(A^\star)$*, valgono le uguaglianze nella Tabella 7.1.*

associatività	$L_1 \cdot (L_2 \cdot L_3) = (L_1 \cdot L_2) \cdot L_3$
unità	$L \cdot \{\varepsilon\} = L = \{\varepsilon\} \cdot L$
assorbimento	$L \cdot \varnothing = \varnothing = \varnothing \cdot L$
distributività di $\cdot$ su $\cup$ (sinistra)	$L_1 \cdot (L_2 \cup L_3) = (L_1 \cdot L_2) \cup (L_1 \cdot L3)$
distributività di $\cdot$ su $\cup$ (destra)	$(L_1 \cup L_2) \cdot L_3 = (L_1 \cdot L_3) \cup (L_2 \cdot L3)$

Tabella 7.1: Leggi per la composizione di linguaggi

La dimostrazione è lasciata per esercizio.

Esercizio 7.1.10. *Dimostrare il Teorema 7.1.9.*

Le leggi in Tabella 7.1, assieme ad associatività, commutatività e unità di $\cup$ (che ovviamente valgono) ci dicono che $(\mathcal{P}(A^\star), \cup, \emptyset, \cdot, \{\varepsilon\})$ forma una struttura ben nota in algebra chiamata *semi-anello*. È interessante notare che la concatenazione $\cdot$ di linguaggi obbedisce alle stesse leggi della composizione ; di relazioni (se pensiamo al linguaggio $\{\varepsilon\}$ come la relazione id_A).

Analogamente alle relazioni (Definizione 3.3.13) possiamo definire il linguaggio L^n per ogni numero naturale n.

Definizione 7.1.11. *Sia* A *un insieme e* $L \in \mathcal{P}(A^\star)$ *un linguaggio. Per ogni numero naturale* $n \in \mathbb{N}$ *definiamo* L^n *induttivamente:*

1. [Clausola base] $L^0 = \{\varepsilon\}$.
2. [Clausola induttiva] $L^{n+1} = L \cdot L^n$.

Esempio 7.1.12. *Sia* $A = \{a, b, c\}$ *e* $L = \{ab, bb\}$*. Allora* $L^0 = \{\varepsilon\}$.

$$\begin{array}{rcll} L^1 & = & L \cdot L^0 & \textit{(Clausola Induttiva)} \\ & = & L \cdot \{\varepsilon\} & \textit{(Clausola Base)} \\ & = & L & \textit{(unità)} \\ & = & \{ab, bb\} & \end{array}$$

In modo analogo

$$\begin{array}{rcll} L^2 & = & L \cdot L^1 & \textit{(Clausola Induttiva)} \\ & = & \{ab, bb\} \cdot \{ab, bb\} & \\ & = & \{abab, abbb, bbab, bbbb\} & \end{array}$$

e

$$\begin{array}{rcll} L^3 & = & L \cdot L^2 & \textit{(Clausola Induttiva)} \\ & = & \{ab, bb\} \cdot \{abab, abbb, bbab, bbbb\} & \\ & = & \{ababab, ababbb, abbbab, abbbbb, bbabab, bbabbb, bbbbab, bbbbbb\} & \end{array}$$

Se $L = \{ab\}$*, allora* $L^0 = \{\epsilon\}$, $L^1 = \{ab\}$, $L_2 = \{abab\}$, $L^3 = \{ababab\}$ *e così via ...*

Osservazione 7.1.13. *Quando il linguaggio L consiste di una sola stringa w, cioè $L = \{w\}$, si ha che L^n consiste sempre di una sola stringa per tutti gli $n \in \mathbb{N}$. Questa stringa è sempre la concatenazione di w con se stessa n volte. Nel resto di queste note, ed in particolare nelle grammatiche libere da contesto, utilizzeremo spesso la notazione w^n per denotare l'unica stringa del linguaggio $\{w\}^n$, cioè la stringa che consiste della concatenazione di w con se stessa n volte. Per esempio, ab^3 è proprio ababab.*

Definizione 7.1.14. *La stella di Kleene è una funzione $(\_)^\star \colon \mathcal{P}(A^\star) \to \mathcal{P}(A^\star)$ definita per tutti i linguaggi $L \in \mathcal{P}(A^\star)$ come*

$$L^\star = \bigcup_{n \in \mathbb{N}} L^n$$

Intuitivamente la stella di Kleene di un linguaggio L è il linguaggio

$$\{\varepsilon\} \cup L \cup L \cdot L \cup L \cdot (L \cdot L) \cup L \cdot (L \cdot (L \cdot L)) \cup \dots$$

Esempio 7.1.15. *Esempi sull'alfabeto $A = \{a, b, c\}$.*

- *Se $L = \{a\}$, allora $L^\star = \{\varepsilon, a, aa, aaa, aaaa, \dots\}$*
- *Se $L = \{a, b\}$, allora $L^\star = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, \dots\}$*
- *Se $L = \{ab\}$, allora $L^\star = \{\varepsilon, ab, abab, ababab, \dots\}$*
- *Se $L = \{ab, c\}$, allora $L^\star = \{\varepsilon, ab, c, abc, cab, ababc, abcab, cabab \dots\}$*
- *Se $L = \{aa, bb\}$, allora $L^\star = \{\varepsilon, aa, bb, aabb, bbaa, aaaa, bbbb, aabbaa \dots\}$*

Teorema 7.1.16. *Per tutti gli alfabeti A e per tutti i linguaggi $L, L_1, L_2 \in \mathcal{P}(A^\star)$, valgono le uguaglianze e le inclusioni nella Tabella 7.2.*

riflessività	$\{\varepsilon\} \subseteq L^\star$
transitività	$L^\star \cdot L^\star \subseteq L^\star$
chiusura	$L \subseteq L^\star$
idempotenza	$(L^\star)^\star = L^\star$
$\star$-id	$\{\varepsilon\}^\star = \{\varepsilon\}$
$\star$-vuoto	$\varnothing^\star = \{\varepsilon\}$
distributività di $\star$ su $\cup$	$L_1^\star \cup L_2^\star \subseteq (L_1 \cup L_2)^\star$

Tabella 7.2: Leggi della stella di Kleene per linguaggi.

La dimostrazione è lasciata per esercizio.

Esercizio 7.1.17. *Dimostrare il Teorema 7.1.16.*

Osservazione 7.1.18 (Analogia con l'algebra relazionale)**.** *Nel Capitolo 2 abbiamo introdotto operazioni su relazioni e le loro leggi algebriche. Nel Capitolo 3 abbiamo esteso queste leggi per la stella di Kleene (chiusura riflessiva e transitiva). Considerando l'insieme $\mathcal{P}(A \times A)$ di tutte le relazioni su A con le operazioni di $\cup$, ;, $\varnothing_{A,A}$ e id_A otteniamo il ben noto* frammento di Kleene *dell'algebra delle relazioni:*

$$(\mathcal{P}(A \times A)\,,\, \cup\,,\, \varnothing_{A,A}\,,;\,,\, Id_A\,,\, (-)^\star)$$

Per il linguaggi abbiamo delle operazioni analoghe:

$$(\mathcal{P}(A^\star)\,,\, \cup\,,\, \varnothing\,,\, \cdot\,,\, \{\varepsilon\}\,,\, (-)^\star)$$

Le Tabelle 7.1 e 7.2 ci dicono che queste due strutture soddisfano le stesse proprietà algebriche.

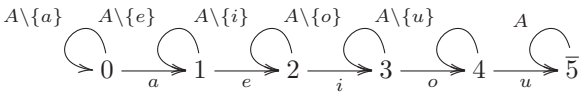


Figura 7.1: Un automa che riconosce tutte le parole che contengono le cinque vocali in ordine alfabetico. A è l'alfabeto Latino.

7.2 Automi

Svariati strumenti di calcolo sono descritti attraverso *macchine a stati*, cioè vengono descritti attraverso transizioni di stato, letture di input e produzioni di output. Tipici esempi di macchine a stati che giocano un ruolo chiave in Informatica sono le macchine di Mealy e di Moore (utilizzate nella progettazione dei processori) o le ben note macchine di Turing (che possono esprimere tutti i programmi calcolabili). In questa sezione introdurremo un esempio classico di macchina a stati, gli *automi*, che permettono di riconoscere le stringhe di un linguaggio. L'idea è che queste macchine prendono come input una stringa e la leggono carattere per carattere: la lettura di un certo carattere fa innescare un passaggio di stato.

Per concretizzare questa idea, consideriamo un problema specifico di riconoscimento di stringhe: "quali parole inglesi contengono le cinque vocali in ordine alfabetico"? Per rispondere a questa domanda, possiamo utilizzare la lista di parole fornita da molti sistemi operativi. Alcune delle parole di questo file che contengono le cinque vocali in ordine, sono: $facetious$, *sacrilegious*. Si noti come anche la seconda parola va bene secondo la formulazione del problema: basta ignorare la prima occorrenza da sinistra della vocale i in *sacrilegious*. Esaminiamo come una macchina molto semplice può trovare tutte le parole che contengono le cinque vocali in ordine. La macchina può leggere ciascuna parola ed esaminarne i caratteri. Iniziando l'esame della parola da sinistra verso destra la macchina cerca una a. Diciamo che la macchina si trova nello "stato 0" fintanto che non trova una a, mentre quando la trova passa nello "stato 1". Nello stato 1, la macchina cerca una e e, quando ne trova una, passa nello "stato 2". Si procede in questo modo fino a raggiungere lo "stato 4" in cui si cerca una u. Se viene trovata una u, allora la parola contiene le vocali, ordinate alfabeticamente, e la macchina può terminare nello stato di riconoscimento, "stato 5", in cui riconosce la parola. Esaminando il resto della parola, la macchina continua a rimanere nello stato 5 in presenza di ogni nuovo carattere letto, dato che sappiamo che la stringa in questione è riconosciuta indipendentemente da ciò che segue u. Possiamo interpretare lo stato i come l'indicazione che la macchina ha già trovato le prime i vocali, in ordine alfabetico, per $i = 0, 1, ..., 5$. I sei stati riassumono tutto quello che il programma deve ricordare durante l'esame della parola, da sinistra a destra. Per esempio, nello stato 0, quando la macchina cerca una a non ha bisogno di ricordare se ha già incontrato una e. La ragione è che questa e non è preceduta da nessuna a e quindi non può servire come e nella sottosequenza *aeiou*.

In Figura 7.1 è rappresentata la macchina a stati descritta nei paragrafi precedenti. Gli stati $0, \ldots, 5$ sono visti come i nodi di un grafo e i passaggi di stato, in gergo tecnico *transizioni*, come degli archi. Gli archi sono etichettati da quei simboli dell'alfabeto A la cui lettura innesca la transizione. Per esempio dallo stato 0 leggendo la lettera a si passa allo stato 1. Invece leggendo un qualsiasi altro simbolo dell'alfabeto (cioè un qualsiasi elemento di $A \setminus \{a\}$) la macchina rimane nello stesso stato 0. Si noti che sopra lo stato 5 è disegnata una sbarra. Tale simbolo denota che lo stato è di accettazione: se la lettura di una stringa conduce a tale stato, allora la stringa contiene le vocali in ordine alfabetico.

Formalizziamo l'intuzione illustrata fino ad ora con la seguente definzione.

Definizione 7.2.1. *Sia A un alfabeto. Un* automa *sull'alfabeto A è una tripla $\mathcal{A} = (S, T, F)$ dove:*

- *S è un insieme, detto* insieme degli stati*;*
- *$T \subseteq (A \times S) \times S$ è una relazione detta* relazione di transizione*, che associa ad ogni lettera dell'alfabeto $a \in A$ e ad ogni stato di partenza $x \in S$ zero o più stati di arrivo.*
- *$F \subseteq S$ è l'insieme degli* stati finali *(anche detti stati di accettazione).*

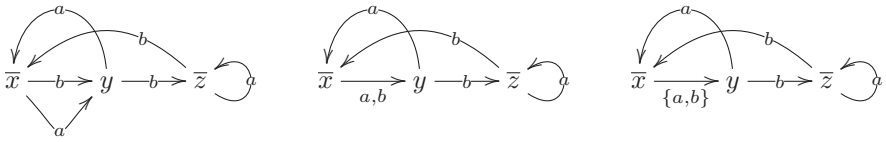


Figura 7.2: Tre diverse rappresentazione grafiche dello stesso automa: nell'automa di sinistra tutti gli archi sono etichettate da esattamente un simbolo. Nell'automa di centro un arco è etichetta con due simboli e nell'automa di destra lo stesso arco è etichettato con un insieme di simboli.

L'automa $\mathcal{A}$ si dice a stati finiti *se l'insieme degli stati S è finito.*

Intuitivamente, gli elementi di S sono tutti gli stati, e quelli di F sono gli stati di accettazione. Un elemento $((a, x), y) \in (A \times S) \times S$ rappresenta una transizione che porta l'automa dallo stato di partenza x allo stato di arrivo y leggendo il simbolo $a \in A$.

Esempio 7.2.2. *La macchina a stati che riconoscce le stringhe che contengono le vocali in ordine alfabetico (rappresentata in Figura 7.1) è un automa: l'insieme degli stati è $S = \{0, \ldots, 5\}$, la relazione di transizione è*

$$\begin{array}{rll} T = & \{((a,0),1),\ ((b,1),2),\ ((c,2),3),\ ((d,3),4),\ ((e,4),5)\} & \cup \\ & \{((i,0),0) \mid i \in A \setminus \{a\}\} & \cup \\ & \{((i,1),1) \mid i \in A \setminus \{b\}\} & \cup \\ & \{((i,2),2) \mid i \in A \setminus \{c\}\} & \cup \\ & \{((i,3),3) \mid i \in A \setminus \{d\}\} & \cup \\ & \{((i,4),4) \mid i \in A \setminus \{e\}\} & \cup \\ & \{((i,5),5) \mid i \in A\} & \end{array}$$

mentre l'insieme degli stati finali è $F = \{5\}$.

Questo esempio mostra quanto la rappresentazione di automi attraverso relazioni possa essere poco intuitiva se paragonata alla rappresentazione grafica (illustrata in Figura 7.1). Per questa ragione, è conveniente spendere qualche parola sulla notazione grafica che è proprio quella che utilizzeremo d'ora in avanti.

Un automa $\mathcal{A} = (S, T, F)$ viene rappresentato in maniera diagrammatica da un grafo orientato con etichette. Gli stati $x \in S$ sono rappresentati come i nodi del grafo. Gli stati finali $y \in F$ sono nodi con una sbarra orizzontale, tipo $\overline{y}$. Ogni transizione $((a, x), y) \in T$ è rappresentata da un arco orientato da x a y con etichetta a (l'etichetta può trovarsi, sopra, sotto o al centro dell'arco).

Esempio 7.2.3. *Sia A l'alfabeto $\{a, b\}$ e $\mathcal{A} = (S, T, F)$ definito come $S = \{x, y, z\}$, $F = \{x, z\}$ e*

$$\begin{array}{rll} T & = \{ & ((a,x),y),\ ((b,x),y), \\ & & ((a,y),x),\ ((b,y),z), \\ & & ((a,z),z),\ ((b,z),x)\}. \end{array}$$

Tale automa è rappresentato dal grafo a sinistra della Figura 7.2.

Talvolta, quando ci sono due (o più) transizioni con gli stessi nodi di partenza e di arrivo e con simboli diversi, si possono accorpare i due (o più) archi scrivendone uno solo, come illustrato per esempio dalla transizione

$$x \xrightarrow[a,b]{} y$$

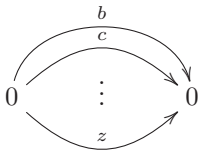
nell'automa al centro della Figura 7.2. Infine, quando i simboli sono molti, può essere conveniente etichettare l'arco con l'insieme dei simboli, come illustrato, per esempio, dalla transizione

$$x \xrightarrow[\{a,b\}]{} y$$

nell'automa alla destra della Figura 7.2. Si noti, come nell'automa in Figura 7.1, l'arco

$$0 \xrightarrow{A \setminus \{a\}} 0$$

potrebbe essere rappresentato come segue.



Definizione 7.2.4. *Sia $\mathcal{A} = (S, T, F)$ un automa sull'alfabeto A. Per ogni $a \in A$, si definisce la relazione $T_a \subseteq S \times S$ come*

$$T_a = \{(x, y) \mid ((a, x), y) \in T\}$$

Per ogni stringa $w \in A^\star$, si definisce la relazione $T_w \subseteq S \times S$ per induzione come segue

1. [Clausola base] $T_\varepsilon = id_S$.
2. [Clausola induttiva] $T_{aw} = T_a; T_w$.

Se $(x, y) \in T_w$ si dice che lo stato y è raggiungibile *da x con la stringa w o che x può raggiungere y con la stringa w.*

Esempio 7.2.5. *Si consideri l'automa dell'Esempio 7.2.3 rappresentato in Figura 7.2 sull'alfabeto $A = \{a, b\}$. Si ha che T_a e T_b sono le seguenti relazioni.*

$$T_a = \{(x, y),\ (y, x),\ (z, z)\} \qquad T_b = \{(x, y),\ (y, z),\ (z, x)\}$$

Intuitivamente T_a contiene tutte le transizioni con simbolo a e T_b contiene tutte le transizioni con simbolo b. Pensando alla rappresentazione grafica (illustrata in Figura 7.2), T_a è l'insieme di archi con etichetta a e T_b l'insieme degli archi con etichetta b. A partire da queste due relazioni possiamo costruire la relazione T_w per ogni $w \in A^\star$. Per esempio per $w = abb$ si ha che

$$\begin{array}{rcll} T_{abb} & = & T_a; T_{bb} & (\text{Clausola induttiva}) \\ & = & T_a; (T_b; T_b) & (\text{Clausola induttiva}) \\ & = & T_a; (T_b; (T_b; T_\varepsilon)) & (\text{Clausola induttiva}) \\ & = & T_a; (T_b; (T_b; id_S)) & (\text{Clausola base}) \\ & = & T_a; T_b; T_b & (\textit{associatività, unità}) \end{array}$$

È facile convincersi che per ogni stringa $w = a_0 \dots a_n$, la relazione T_w è esattamente

$$T_{a_0 \dots a_n} = T_{a_0}; \dots; T_{a_n}$$

Tenendo in mente l'uguaglianza riportata sopra è immediato calcolare T_w. Riportiamo di seguito tutti gli esempi con stringhe di lunghezza 2.

$$\begin{array}{rcl} T_{aa} & = & \{(x, x),\ (y, y),\ (z, z)\} \\ T_{ab} & = & \{(x, z),\ (y, y),\ (z, x)\} \\ T_{ba} & = & \{(x, x),\ (y, z),\ (z, y)\} \\ T_{bb} & = & \{(x, z),\ (y, x),\ (z, y)\} \end{array}$$

Mostriamo anche un esempio con una stringa di lunghezza 3.

$$T_{aaa} \quad = \quad \{(x, y),\ (y, x),\ (z, z)\}$$

L'esempio appena illustrato, da un'intuizione chiara del significato di T_a e T_w. La relazione T_a contiente tutte e solo le coppie (x, y) tali che c'è una transizione da x a y con simbolo a, cioè $((a, x), y) \in T$. Nella relazione T_w ci sono tutte e sole le coppie (x, y) tali che c'è un walk (un cammino) da x a y etichettato con w. Per essere più precisi, se $w = a_0 \dots a_n$, la relazione T_w è esattamente

$$T_{a_0 \dots a_n} = \{(x, y) \mid \exists z_1, \dots z_n \in S \text{ tali che } ((a_0, x), z_1) \in T, \dots, ((a_n, z_n), y) \in T\}.$$

Esempio 7.2.6. *Consideriamo nuovamente l'automa in Figura 7.1 e concentriamoci sullo stato* 0.

È facile vedere che lo stato 5 *è raggiungibile dallo stato* 0 *con la stringa aeiou. Graficamente si può visualizzare il seguente walk:*

$$0 \xrightarrow[a]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[i]{} 4 \xrightarrow[u]{} 5$$

Si osservi che 5 *è raggiungibile da* 0 *anche con la stringa baeiou, poichè* $(0,0) \in T_b$ *e* $(0,5) \in T_{aeiou}$. *Graficamente si può visualizzare il seguente walk:*

$$0 \xrightarrow[b]{} 0 \xrightarrow[a]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[o]{} 4 \xrightarrow[u]{} 5$$

Lo stesso vale per cbaeiou o bcbaeiou.

$$0 \xrightarrow[c]{} 0 \xrightarrow[b]{} 0 \xrightarrow[a]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[o]{} 4 \xrightarrow[u]{} 5$$

$$0 \xrightarrow[b]{} 0 \xrightarrow[c]{} 0 \xrightarrow[b]{} 0 \xrightarrow[a]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[o]{} 4 \xrightarrow[u]{} 5$$

Più in generale, per ogni stringa w_0 *che non contiene a vale che* 5 *è raggiungibile da* 0 *attraverso la stringa* w_0*aeiou: infatti* $(0,0) \in T_{w_0}$ *e* $(0,5) \in T_{aeiou}$. *Con un ragionamento analogo, si può vedere che per tutte le stringhe* w_1 *che non contengono il simbolo e vale che* 5 *è raggiungibile da* 0 *attraverso* aw_1*eiou. Mostriamo di seguito la visualizzazione dei walk per* $w_1 = b$, $w_1 = cb$ *e* $w_1 = bcb$.

$$0 \xrightarrow[a]{} 1 \xrightarrow[b]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[o]{} 4 \xrightarrow[u]{} 5$$

$$0 \xrightarrow[a]{} 1 \xrightarrow[c]{} 1 \xrightarrow[b]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[o]{} 4 \xrightarrow[u]{} 5$$

$$0 \xrightarrow[a]{} 1 \xrightarrow[b]{} 1 \xrightarrow[c]{} 1 \xrightarrow[b]{} 1 \xrightarrow[e]{} 2 \xrightarrow[i]{} 3 \xrightarrow[o]{} 4 \xrightarrow[u]{} 5$$

Ragionando in modo simile si ha che 5 *è raggiungibile da* 0 *attraverso la stringa*

$$w_0 a w_1 e w_2 i w_3 o w_4 u w_5 \tag{7.1}$$

dove w_5 *è una stringa arbitraria, mentre* w_0, w_1, w_2, w_3, w_4 *sono stringhe arbitrarie in cui non compaiono, rispettivamente, i simboli a, e, i, o, u.*

La relazione T_w è fondamentale per definire il linguaggio accettato da uno stato dell'automa.

Definizione 7.2.7. *Sia* $\mathcal{A} = (S, T, F)$ *un automa sull'alfabeto* A. *La funzione* $\langle\langle\cdot\rangle\rangle \colon S \to \mathcal{P}(A^\star)$ *è definita per ogni stato* $x \in S$ *come*

$$\langle\langle x \rangle\rangle = \{w \in A^\star \mid \text{ esiste } y \in F \text{ tale che } (x,y) \in T_w\}.$$

Il linguaggio $\langle\langle x \rangle\rangle$ *è detto il* linguaggio accettato *da* x. *Se* $w \in \langle\langle x \rangle\rangle$, *si dice che la stringa* w *è* accettata *dallo stato* x *(o appartiene al linguaggio accettato da* x*). Se* $w \notin \langle\langle x \rangle\rangle$, *si dice che la stringa* w non è accettata *dallo stato* x *(o non appartiene al linguaggio accettato da* x*).*

Intuitivamente, per vedere se una stringa w è accettata da uno stato x si devono guardare tutti gli stati y che sono raggiungibili da x con w e, se tra questi c'è almeno uno stato finale, allora la stringa w è accettata, altrimenti non è accettata.

Esempio 7.2.8. *Si consideri l'automa dell'Esempio 7.2.3 rappresentato in Figura 7.2 sull'alfabeto* $A = \{a, b\}$ *e consideriamo lo stato* x.

Iniziamo con la stringa vuota ε. *Si ha che* $T_\varepsilon = id_S$ *e quindi* $(x,x) \in T_\varepsilon$ *e, fatto ancor più interessante,* x *è l'unico stato raggiungibile da* x *con* ε. *Visto che* $x \in F$, *allora* ε *è accettata da* x, *cioè*

$$\varepsilon \in \langle\langle x \rangle\rangle.$$

Consideriamo la stringa a: si ha che $(x,y) \in T_a$ *e che, inoltre,* y *è l'unico stato raggiungibile da* x *attraverso la stringa a. Visto che* $y \notin F$, *allora la stringa a non è accettata da* x, *cioè*

$$a \notin \langle\langle x \rangle\rangle.$$

Esattamente lo stesso argomento vale per b, possiamo quindi concludere che

$$b \notin \langle\langle x \rangle\rangle.$$

Proviamo adesso aa. L'unico stato raggiungibile da x con aa è lo stesso x che appartiene ad F. Quindi aa è accettata da x. Proviamo adesso ab. L'unico stato raggiungibile da x con ab è z che appartiene ad F. Quindi ab è accettata da x. Proviamo adesso ba. L'unico stato raggiungibile da x con ba è lo stesso x che appartiene ad F. Quindi ba è accettata da x. Proviamo adesso bb. L'unico stato raggiungibile da x con bb è z che appartiene ad F. Quindi bb è accettata da x. In sintesi:

$$aa, ab, ba, bb \in \langle\langle x \rangle\rangle.$$

Per aaa, si ha che l'unico stato raggiungibile da x con aaa è lo stato y che non appartiene ad F. Pertanto aaa non è accettata da x, cioè

$$aaa \notin \langle\langle x \rangle\rangle.$$

Esempio 7.2.9. *Consideriamo nuovamente l'automa in Figura 7.1 e concentriamoci sullo stato* 0*. Visto che l'unico stato di accettazione è* 5*, una qualsiasi stringa w è accettata da* 0 *se e solo se* 5 *è raggiungibile da* 0 *attraverso w (cioè $(0, 5) \in T_w$). Abbiamo visto che tutte le stringhe della forma specificata da* (7.1) *permettono allo stato* 0 *di raggiungere* 5*. Pertanto tutte queste stringhe vengono accettate. Si noti che queste sono esattamente tutte le parole in cui c'è almeno un'occorenza delle vocali a, e, i, o, u in ordine alfabetico, come avevamo desiderato all'inizio di questa sezione.*

Esercizio 7.2.10. *Sia A l'alfabeto Latino. Disegnare un automa su A in cui uno stato accetta tutte e solo le parole che iniziano con al, come per esempio algoritmo e algebra.*

Esercizio 7.2.11. *Si consideri l'alfabeto $A = \{a, b\}$ ed il linguaggio*

$$L = \{w \in A^\star \mid \text{il simbolo } a \text{ occorre un numero pari di volte in } w\}.$$

Per esempio $aba \in L$ mentre $abb \notin L$. Definire un automa in cui uno stato accetta esattamente il linguaggio L.

Automi Deterministici e Non

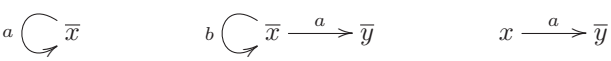
A questo punto, una considerazione è di fondamentale importanza. Negli esempi che abbiamo visto fino ad adesso, le relazioni di transizione sono delle funzioni: associano ad ogni coppia $(a, x) \in A \times S$ uno ed un solo stato di arrivo $y \in S$. In altre parole, da un qualsiasi stato x, leggendo un qualsiasi simbolo a, si transisce in esattamente uno stato y. Ne segue che le relazioni T_a, per ogni $a \in A$, e le relazioni T_w, per ogni $w \in A^\star$, sono delle funzioni da S in S: ogni stato x raggiunge con w esattamente uno stato y. Questi automi sono chiamati *automi deterministici*.

Definizione 7.2.12. *Un automa $\mathcal{A} = (S, T, F)$ si dice* deterministico *se la relazione di transizione $T \subseteq (A \times S) \times S$ è una funzione.*

Per esempio, gli automi Figura 7.1 e in Figura 7.2 sono deterministici.

La definizione generale di automa (Definizione 7.2.1) prevede però che la relazione di transizione $T \subseteq (A \times S) \times S$ sia, in tutta generalità, una relazione e non necessariamente una funzione. Ci possono quindi essere anche degli automi che non sono deterministici: questi sono noti come *automi non-deterministici*.

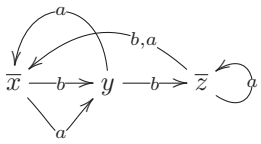
Esempio 7.2.13. *Si consideri l'alfabeto $A = \{a, b\}$. I seguenti tre automi non sono deterministici, in quanto la relazione di transizione non è* totale*.*



(7.2)

Infatti, nell'automa di sinistra e in quello di destra non esiste uno stato z tale che $((b, x), z) \in T$. Si osservi che, più in generale, x non può raggiungere alcuno stato con una qualsiasi stringa che inzia con b. Nell'automa di centro non esiste uno stato z tale che $((b, y), z) \in T$.

Il seguente automa non è deterministico in quanto la relazione di transizione non è univalente*.*



(7.3)

Infatti, vale sia che $((a, z), x) \in T$ *che* $((a, z), z) \in T$*. Intuitivamente, leggendo il simbolo a, lo stato z può transire sia in x che in se stesso. Si osservi che con la stringa ab lo stato z può raggiungere due stati: sia y che x. Il primo non è di accettazione, mentre il secondo lo è. Il lettore probabilmente si chiederà:*

la stringa ab è accettata dallo stato z?

La risposta è sì*: per definizione di linguaggio accettato (Definizione 7.2.7) la stringa ab è accettata da z se esiste* almeno *uno stato di accettazione che è raggiungibile da y attraverso ab. E in questo caso, tale stato esiste: è lo stato x.*

L'esempio precedente illustra un fenomeno importante che avviene negli automi che non sono deterministici: un dato stato può raggiungere con la stessa stringa diversi stati. La stringa è accettata se tra tutti questi ce n'è almeno uno di accettazione.

Questa osservazione ci permette di illustrare uno dei concetti fondamentali dell'informatica teorica: il *non determinismo*.

Osservazione 7.2.14. *Gli algoritmi che utilizziamo solitamente sono deterministici: dato uno stesso input si arriva sempre allo stesso output. Invece, negli algoritmi non deterministici uno stesso input può portare ad output diversi. Mentre gli algoritmi deterministici possono essere pensati come* funzioni *`INPUT`*$\to$ *`OUTPUT`, gli algoritmi non deterministici sono delle* relazioni *tra `INPUT` e `OUTPUT`.*

È interessante notare che le macchine a stati con cui ci interfacciamo quotidianamente (i computer) sono deterministiche: procedono attraverso una serie di transizioni di stato in cui lo stato di arrivo è determinato *dallo stato di partenza.*

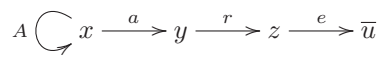
Il non determinismo acquista un significato ancor più specifico quando, nel contesto dell'Informatica Teorica (ad esempio nelle macchine di Turing non deterministiche), si considerano problemi con output booleano, o `t` o `f`: un algoritmo non deterministico può eseguire più computazioni che portano a risultati diversi ma, se almeno una computazione ha successo (risposta `t`), allora il problema ha risposta `t`.

Nell'esempio illustrato sopra, il problema è: $ab \in \langle\langle z \rangle\rangle$*? Partendo dallo stato z e prendendo in input la stringa ab si può arrivare sia allo stato y che non è di accettazione (output `f`) che allo stato x che è di accettazione (output `t`). Il problema ha quindi risposta `t`.*

La portata del non determinismo in informatica va ben al di là degli automi e non può essere discussa in questo corso introduttivo. Vogliamo però rammentare al lettore che il famoso problema $P = NP$ *(uno dei sette problemi del millennio) è intimamente collegato a questa idea di non determinismo.*

Nonostante che le macchine a stati con cui ci interfacciamo quotidianamente siano deterministiche, le macchine non deterministiche sono degli strumenti utilissimi per specificare problemi e modellare le loro soluzioni. Il seguente esempio illustra una comodità offerta dagli automa non deterministici.

Esempio 7.2.15. *Siamo interessati a trovare tutte le parole che finiscono con are, come andare, mangiare, care o bare (nella lingua italiana sono 11137). A tale scopo possiamo progettare un automa in cui uno stato accetta tutte e sole queste parole. Fissiamo A essere l'alfabeto Latino e consideriamo il seguente automa.*



(7.4)

Siamo interessati a $\langle\langle x \rangle\rangle$*, il linguaggio accettato dallo stato x. Essendo u l'unico stato di accettazione (cioè* $F = \{u\}$*) una stringa w è accettata da x se e soltanto se u è raggiungibile da x attraverso w. I seguenti walk mostrano che le parole care e bare sono accettate da x.*

$$x \xrightarrow{c} x \xrightarrow{a} y \xrightarrow{r} z \xrightarrow{e} u$$

$$x \xrightarrow{b} x \xrightarrow{a} y \xrightarrow{r} z \xrightarrow{e} u$$

È importante adesso notare che la relazione T_{care} non contiene solo (x, u), ma anche la coppia (x, x). Infatti, dopo la lettura di c (che lascia l'automa nello stato x) la lettura di a può innescare due diverse transizioni: una che finisce nello stato y, come mostrato sopra, oppure una che torna in x, come illustrato di seguito.

$$x \xrightarrow{c} x \xrightarrow{a} x \xrightarrow{r} x \xrightarrow{e} x$$

Lo stesso vale per bare.

$$x \xrightarrow{b} x \xrightarrow{a} x \xrightarrow{r} x \xrightarrow{e} x$$

Il lettore potrebbe adesso pensare che questo comportamento sia anomalo e che si possa eliminare considerando il seguente automa.

$$A\backslash\{a\} \circlearrowright x' \xrightarrow{a} y' \xrightarrow{r} z' \xrightarrow{e} \overline{u'}$$

In realtà tale automa non è ciò che desideriamo in quanto lo stato x' non accetta la stringa andare: infatti leggendo la prima a, dallo stato x' si transisce nello stato y' e, leggendo la n da questo stato non si può andare in alcuno stato. In altre parole nessuno stato è raggiungibile da x' con la stringa andare e pertanto andare non è accettata.

La stringa andare invece è accettata dallo stato x dell'automa in (7.4)*, come mostrato dal seguente walk da x a u.*

$$x \xrightarrow{a} x \xrightarrow{n} x \xrightarrow{d} x \xrightarrow{a} y \xrightarrow{r} z \xrightarrow{e} u$$

Intuivamente x può scegliere come comportarsi con a: se tornare su se stesso oppure se transire in y. Quando legge la prima a di andare, resta su se stesso; quando legge la seconda, transisce in y.

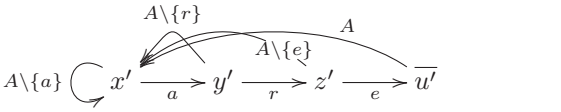
Le stesse considerazioni valgono per la stringa mangiare e, più in generale per qualsiasi stringa della forma $w_1$are dove w_1 è un'arbitraria stringa su $A^\star$. Infatti si ha che $(x, x) \in T_{w_1}$ e $(x, u) \in T_{are}$, e quindi $(x, u) \in T_{w_1are}$. Visto che $u \in F$, allora $w_1$are è acettata dallo stato x. Questo dimostra che lo stato x accetta tutte le parole che terminano in are.

Esercizio 7.2.16. *Dimostrare che per ogni alfabeto A, per ogni automa $\mathcal{A} = (S, T, F)$ sull'alfabeto A, per ogni stato $x \in S$, simbolo $a \in A$ e stringa $w \in A^\star$ vale che:*

1. *$\varepsilon \in \langle\langle x \rangle\rangle$ se e solo se $x \in F$;*
2. *$aw \in \langle\langle x \rangle\rangle$ se e solo se esiste $y \in S$ tale che $(x, y) \in T_a$ e $w \in \langle\langle y \rangle\rangle$.*

La costruzione dei sottoinsiemi

L'Esempio 7.2.15 mostra che lo stato x dell'automa non deterministico in (7.4) accetta il linguaggio di tutte le parole che finisco in *are* (cioè $\langle\langle x \rangle\rangle = \{w_1are \mid w_1 \in A^\star\}$). In realtà, è possibile creare anche un automa deterministico con uno stato che accetta lo stesso linguaggio. Tale automa è illustrato di seguito.



(7.5)

Si noti innanzitutto che l'automa è deterministico, per ogni coppia $A \times S$ esiste esattamente uno stato successore. Inoltre, il linguaggio accettato da x' è il linguaggio di tutte le parole che terminano in *are*, cioè le parole della forma w_1are per $w_1 \in A^\star$. In altre parole, il linguaggio accettato da x nell'automa non deterministico in (7.4) è lo stesso del linguaggio accettato dallo stato x' nell'automa deterministico (7.5).

Osservazione 7.2.17. *Visto che le macchine a stati reali sono tipicamente deterministiche l'automa in* (7.5) *può essere pensato come una* implementazione *realmente eseguibile. L'automa non deterministico in* (7.4)*, invece non può essere realmente esguito su una macchina a stati deterministica, ma è comunque utile perchè rappresenta una* specifica*, cioè una definizione formale del comportamento (nel caso degli automi, del linguaggio) desiderato.*

Questo esempio suggerisce una domanda:

dato uno stato x di un automa qualsiasi (a stati finiti), è sempre possibile costruire un automa *deterministico* (a stati finiti) in cui uno stato accetta lo stesso linguaggio di x?

La risposta è stata fornita da Michael Rabin e Dana Scott nel 1959 (questa scoperta ha valso loro il prestigioso premio Turing): sì, è sempre possibile trasformare un automa in uno deterministico utilizzando la *costruzione dei sottoinsiemi*. Inoltre se l'automa originale è a stati finiti, l'automa deterministico risultante da questa costruzione è anche esso a stati finiti. La costruzione è illustrata nella seguente definizione.

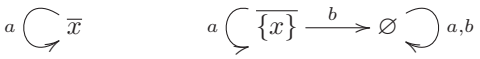
Definizione 7.2.18. *Dato un automa $\mathcal{A} = (S, T, F)$, l'automa deterministico*

$$Det(\mathcal{A}) = (\mathcal{P}(S), T^\sharp, F^\sharp)$$

è definito come segue:

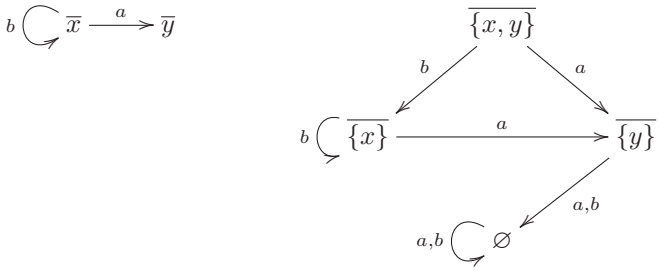
- *L'insieme degli stati è $\mathcal{P}(S)$, cioè l'insieme dei sottoinsiemi di stati dell'automa originale $\mathcal{A}$;*
- *La funzione di transizione $T^\sharp\colon A \times \mathcal{P}(S) \to \mathcal{P}(S)$ è definita per ogni $a \in A$ ed ogni sottoinsieme $X \subseteq S$ come*
$$T^\sharp(a, X) = \{z \mid \text{ esiste } x \in X \text{ tale che } ((a, x), z) \in T\};$$
- *L'insieme degli stati finali $F^\sharp \subseteq \mathcal{P}(S)$ è definito come*
$$F^\sharp = \{X \subseteq S \mid \text{ esiste } x \in X \text{ e } x \in F\}$$

Esempio 7.2.19. *Prendiamo come alfabeto $A = \{a, b\}$. Consideriamo l'automa a sinistra in (7.2): $S = F = \{x\}$ e $T = \{((a, x), x)\}$. Pertanto l'insieme degli stati in $Det(\mathcal{A})$ è $\mathcal{P}(S) = \{\varnothing, \{x\}\}$, la funzione di transizione è $T^\sharp = \{((a, \{x\}), \{x\}),\ ((b, \{x\}), \varnothing),\ ((a, \varnothing), \varnothing),\ ((b, \varnothing), \varnothing)\}$ e $F^\sharp = \{\{x\}\}$. Di seguito illustriamo l'automa non deterministico originale (a sinistra) ed il corrispondente automa deterministico.*



Si noti che per ogni simbolo dell'alfabeto $A = \{a, b\}$ c'è esattamente una transizione che esce da ogni stato. In particolare nell'automa determinizzato c'è una transizione da $\{x\}$ a $\varnothing$ con il simbolo b, visto che $T^\sharp(b, \{x\})$ è, per definizione, l'insieme di tutti gli stati raggiungibili da x con b nell'automa originale, cioè nessuno. Esattamente per la stessa ragione, $\varnothing$ ha sempre una transizione su se stesso, per ogni simbolo in A. Si noti inoltre che $\varnothing \notin F$, dal momento che non contiene alcuno stato.

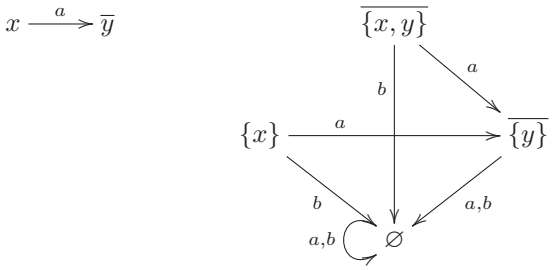
Per $\mathcal{A}$ l'automa al centro in (7.2), $Det(\mathcal{A})$ è riportato di seguito sulla destra.



(7.6)

In questo automa, sono degne di nota le transizioni dello stato: $\{x, y\}$. Infatti $T^\sharp(a, \{x, y\}) = \{x\}$ poichè x è l'unico stato raggiungibile con a o da x o da y. Per la stessa ragione $T^\sharp(b, \{x, y\}) = \{y\}$ poichè y è l'unico stato raggiungibile con b o da x o da y.

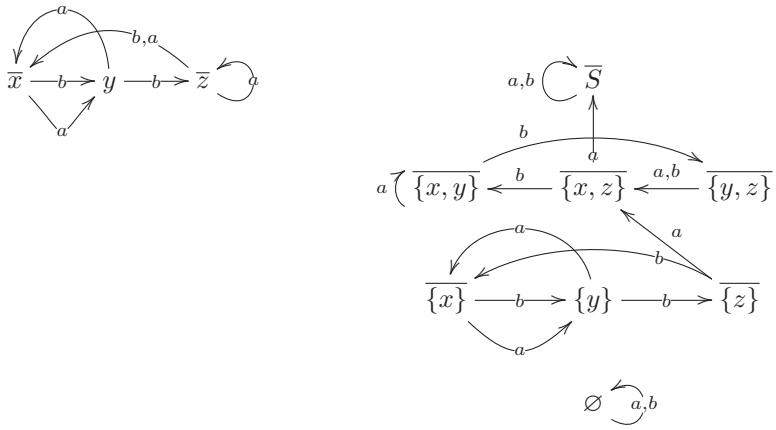
Per $\mathcal{A}$ l'automa a destra in (7.2)*, $Det(\mathcal{A})$ è riportato di seguito sulla destra.*



(7.7)

Osserviamo che $\{x, y\}$ è di accettazione (cioè $\{x, y\} \in F^\sharp$) nonostante che nell'automa originale x non sia di accettazione (cioè $x \notin F$). Infatti è sufficiente che $y \in F$ per avere che $\{x, y\} \in F^\sharp$.

Per $\mathcal{A}$ l'automa in (7.3)*, $Det(\mathcal{A})$ è riportato di seguito sulla destra.*



Si noti che il non determinismo è stato rimosso: con il simbolo a, lo stato $\{z\}$ raggiunge esattamente uno stato $\{x, z\}$. Infatti per definizione $T^\sharp(a, \{z\}) = \{x, z\}$ è l'insieme di tutti gli stati raggiungibili da z con a. Inoltre, $T^\sharp(a, \{x, z\}) = \{x, y, z\} = S$, poichè x con a transisce in y e z con a transisce sia in x che in z.

La costruzione dei sottoinsiemi è interessante perchè preserva il linguaggio accettato, come enunciato dal seguente teorema.

Teorema 7.2.20 (Teorema di corrispondenza)**.** *Sia $\mathcal{A} = (S, T, F)$ un automa e $Det(\mathcal{A}) = (\mathcal{P}(S), T^\sharp, F^\sharp)$ il corrispondente automa deterministico. Per ogni stato $x \in S$ dell'automa $\mathcal{A}$ vale che:*

$$\langle\langle x \rangle\rangle = \langle\langle \{x\} \rangle\rangle$$

7.3 Grammatiche Libere da Contesto

Nella sezione precedente abbiamo visto gli *automi*, certe macchine a stati che permettono di *riconoscere* le stringhe di un linguaggio. In questa sezione consideriamo le *grammatiche libera da contesto* che permettono di descrivere linguaggi, *generando* le stringhe in modo ricorsivo.

Un'importante applicazione delle grammatiche è la specifica della sintassi dei linguaggi di programmazione: le grammatiche costituiscono, infatti, una notazione comoda e concisa per descrivere la sintassi di un tipico linguaggio di programmazione e, alla fine di questa sezione, vedremo un semplice esempio a sostegno di questa affermazione. Nel prossimo capitolo vedremo come queste grammatiche permettono di specificare la sintassi del linguaggio logico.

Durante tutta questa sezione utilizzeremo un esempio ben noto al lettore: il linguaggio delle *espressioni aritmetiche*. Per prima cosa è opportuno puntualizzare che le espressioni che stiamo per definire sono diverse da quelle introdotte nel Capitolo 5. In questa sezione, pensiamo alle espressioni aritmetiche come stringhe sull'alfabeto

$$A = \{+, -, \times, \div\} \cup \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}. \tag{7.8}$$

Sin dalle scuole elementari siamo abituati a manipolare queste espressioni e sappiamo bene che scrivere $5 + \times \div$ non ha alcun significato, mentre scrivere $5 \times 4 + 1$ è corretto. Dobbiamo adesso pensare che sia $5 + \times \div$ che $5 \times 4 + 1$ sono entrambe stringhe sull'alfabeto A in (7.8), ma solo la seconda appartiene al linguaggio delle espressioni artimetiche. Per esprimere questo linguaggio in modo preciso e sistematico (in modo che un computer lo possa riconoscere) utilizziamo la seguente grammatica libera da contesto.

$$\begin{array}{lcl} \langle ExA \rangle & \rightsquigarrow & \langle Num \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle + \langle ExA \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle - \langle ExA \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle \times \langle ExA \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle \div \langle ExA \rangle \end{array} \tag{7.9}$$

Intuitivamente, la grammatica ci dice che un'espressione aritmetica ($\langle ExA \rangle$) o è un numero ($\langle Num \rangle$) oppure la somma ($+$), la sottrazione ($-$), la moltiplicazione ($\times$), la divisione ($\div$) di due espressioni aritmetiche.

Introduciamo adesso un po' di termini tecnici.

- I simboli tra parentesi angolate, come $\langle ExA \rangle$ o $\langle Num \rangle$, vengono chiamati *categorie sintattiche* o *simboli non-terminali*. Intuitivamente queste denotano linguaggi. Vedremo dopo, la definizione formale di tali linguaggi.
- Il simbolo $\rightsquigarrow$ è un *metasimbolo*, che si legge "può essere".
- Gli altri simboli $+, -, \times, \div$ sono i *simboli terminali*, cioè i simboli dell'alfabeto del linguaggio generato (nel nostro caso A in (7.8)).

Nella grammatica in (7.9) non è specificato cosa è un numero (rappresentato dalla categoria sintattica $\langle Num \rangle$). Per completarla, consideriamo la seguente grammatica che specifica che un numero è una sequenza di cifre (categoria sintattica $\langle Cif \rangle$), cioè una sequenza di simboli in $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$.

$$\begin{array}{lcl} \langle Cif \rangle & \rightsquigarrow & 0 \\ \langle Cif \rangle & \rightsquigarrow & 1 \\ \langle Cif \rangle & \rightsquigarrow & 2 \\ \langle Cif \rangle & \rightsquigarrow & 3 \\ \langle Cif \rangle & \rightsquigarrow & 4 \\ \langle Cif \rangle & \rightsquigarrow & 5 \\ \langle Cif \rangle & \rightsquigarrow & 6 \\ \langle Cif \rangle & \rightsquigarrow & 7 \\ \langle Cif \rangle & \rightsquigarrow & 8 \\ \langle Cif \rangle & \rightsquigarrow & 9 \\ \langle Num \rangle & \rightsquigarrow & \langle Cif \rangle \\ \langle Num \rangle & \rightsquigarrow & \langle Num \rangle \langle Cif \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle Num \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle + \langle ExA \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle - \langle ExA \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle \times \langle ExA \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle ExA \rangle \div \langle ExA \rangle \end{array} \tag{7.10}$$

Continuiamo ad introdurre un po' di terminologia. Nelle grammatiche, ogni riga viene detta una *produzione*. Per esempio

$$\langle ExA \rangle \quad \rightsquigarrow \quad \langle ExA \rangle - \langle ExA \rangle$$

è una produzione. Per essere più precisi, una produzione consiste di un simbolo non terminale a sinistra di $\rightsquigarrow$ e, alla sua destra, una sequenza di simboli terminali e non. Il simbolo non terminale a sinistra di $\rightsquigarrow$ viene detto la *testa* della produzione, mentre la sequenza di simboli a destra di $\rightsquigarrow$ viene detta il *corpo* della produzione.

Se S è l'insieme dei simboli non terminali ed A è l'insieme dei simboli terminali, allora a destra di $\rightsquigarrow$ c'è un elemento dell'insieme $(S \cup A)^\star$ (si assume sempre che gli insiemi A ed S siano disgiunti).

Definizione 7.3.1. *Una grammatica libera da contesto $\mathcal{G}$ sull'alfabeto A consiste di una coppia (S, P) dove:*

- *S è l'insieme dei simboli non termimali (S è disgiunto da A);*
- *P è un insieme di produzioni: ogni produzione ha la forma*

$$\langle X \rangle \rightsquigarrow w$$

dove $\langle X \rangle \in S$ e $w \in (A \cup S)^\star$. Si noti che w può essere anche la stringa vuota ε.

Una grammatica $\mathcal{G}$ si dice finita se l'insieme dei simboli terminali S è finito.

Esercizio 7.3.2. *Fissiamo un alfabeto A e un insieme S di simboli non terminali. Esiste una biiezione tra l'insieme di tutte le produzioni su A e S e l'insieme di tutte le relazioni tra S e $(S \cup A)^\star$?*

Per alleggerire la notazione, si possono rappresentare più produzioni su una sola riga: si introduce un ulteriore metasimbolo | che si legge "oppure" (si noti che lo stesso simbolo si usa per gli insiemi, ma con un significato completamente diverso) e le produzioni

$$\begin{array}{lcl} \langle X \rangle & \rightsquigarrow & w_1 \\ \langle X \rangle & \rightsquigarrow & w_2 \\ \cdots & & \\ \langle X \rangle & \rightsquigarrow & w_n \end{array}$$

si rappresentano come

$$\langle X \rangle \quad \rightsquigarrow \quad w_1 \mid w_2 \mid \cdots \mid w_n.$$

Per esempio, la grammatica in (7.10) può essere espressa in modo conciso come segue.

$$\begin{array}{rcl} \langle Cif \rangle & \rightsquigarrow & 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\ \langle Num \rangle & \rightsquigarrow & \langle Cif \rangle \mid \langle Num \rangle \langle Cif \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle Num \rangle \mid \langle ExA \rangle + \langle ExA \rangle \mid \langle ExA \rangle - \langle ExA \rangle \mid \\ & & \langle ExA \rangle \times \langle ExA \rangle \mid \langle ExA \rangle \div \langle ExA \rangle \end{array} \tag{7.11}$$

Il significato non cambia: un'espressione aritmetica *può essere* un numero, *oppure* una somma, *oppure* una sottrazione *oppure* ...

Osservazione 7.3.3. *La sintassi di molti linguaggi formali, come linguaggi di programmazione, linguaggi di specifica, protocolli di rete, linguaggi logici è definita con la notazione utilizzata sopra. Talvolta, il metasimbolo $\rightsquigarrow$ è rimpiazzato da $::=$ e, in certi contesti, i simboli non terminali non si trovano all'interno delle parentesi angolate $\langle$ e $\rangle$.*

Questo tipo di definizione è spesso chiamata Backus Normal Form *o* Backus-Naur Form*, o con l'acronimo BNF, perché fu proposta da John Backus della definizione del linguaggio di programmazione ALGOL.*

Continuiamo ad arricchire le espressioni aritmetiche ed illustriamo come rappresentare espressioni aritmetiche con variabili, come per esempio $x + 3 \times y$. Innanzitutto estendiamo l'alfabeto A in modo da poter rappresentare le variabili con degli identificatori (stringhe di caratteri $\{a, b, c, \ldots, z\}$)

$$A = \{+, -, \times, \div\} \cup \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \cup \{a, b, c, \ldots, z\}. \tag{7.12}$$

Poi estendiamo la precedente grammatica con la categoria sintattica degli identificatori ($\langle Ide \rangle$) che intuitivamente contiene tutte le stringhe formate dai caratteri.

$$\begin{array}{rcl} \langle Cif \rangle & \rightsquigarrow & 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\ \langle Num \rangle & \rightsquigarrow & \langle Cif \rangle \mid \langle Num \rangle \langle Cif \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle Num \rangle \mid \langle ExA \rangle + \langle ExA \rangle \mid \langle ExA \rangle - \langle ExA \rangle \mid \\ & & \langle ExA \rangle \times \langle ExA \rangle \mid \langle ExA \rangle \div \langle ExA \rangle \mid \\ & & \langle Ide \rangle \\ \langle Ide \rangle & \rightsquigarrow & \langle Car \rangle \mid \langle Ide \rangle \langle Car \rangle \\ \langle Car \rangle & \rightsquigarrow & a \mid b \mid \cdots \mid z \end{array} \tag{7.13}$$

Si noti che in questa grammatica un'espressione aritmetica può essere anche un identificatore. Abbiamo quindi che un'espressione aritmetica può essere per esempio $x + 3 \times y$.

La grammatica in (7.13) può essere resa più compatta aggiungendo la categoria sintattica degli operatori aritmetici ($\langle OpA \rangle$).

$$\begin{array}{rcl} \langle OpA \rangle & \rightsquigarrow & + \mid - \mid \times \mid \div \\ \langle Cif \rangle & \rightsquigarrow & 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\ \langle Num \rangle & \rightsquigarrow & \langle Cif \rangle \mid \langle Num \rangle \langle Cif \rangle \\ \langle ExA \rangle & \rightsquigarrow & \langle Num \rangle \mid \langle ExA \rangle \langle OpA \rangle \langle ExA \rangle \mid \langle Ide \rangle \\ \langle Ide \rangle & \rightsquigarrow & \langle Car \rangle \mid \langle Ide \rangle \langle Car \rangle \\ \langle Car \rangle & \rightsquigarrow & a \mid b \mid \cdots \mid z \end{array} \tag{7.14}$$

Come abbiamo anticipato, ogni categoria sintattica denota un linguaggio. Al fine di spiegare come tale linguaggio è generato, dobbiamo introdurre il concetto di albero di derivazione sintattica.

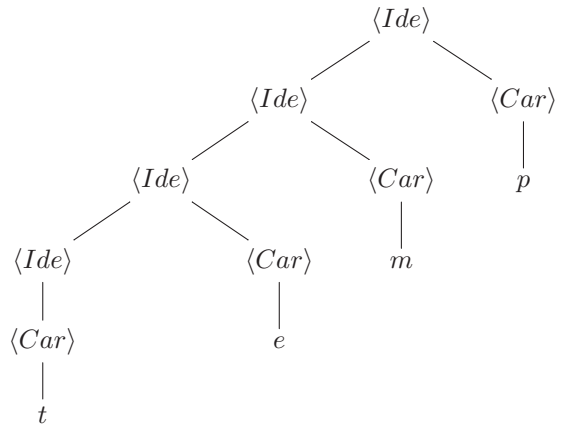
Definizione 7.3.4. *Sia (S, P) una grammatica libera da contesto sull'alfabeto A. Un* albero di derivazione sintattica *(o* parse tree*) è un albero radicato dove:*

- *ogni nodo interno è etichettato da un simbolo non terminale (cioè un simbolo in S);*
- *ogni foglia è etichettata da un simbolo terminale (cioè un simbolo in A) o la stringa vuota;*
- *ogni nodo interno v rappresenta l'applicazione di una produzione, ovvero deve esistere una produzione tale che:*
  - (a) *l'etichetta del nodo v è la testa della produzione;*
  - (b) *l'etichette dei figli di v, da sinistra a destra, formano il corpo della produzione.*

Sia $w \in A^\star$ la stringa ottenuta leggendo da sinistra a destra i simboli terminali che etichettano le foglie dell'albero (la stringa vuota si ignora). Si dice w è la stringa testimoniata dall'albero o che l'albero è un albero di derivazione (sintattica) *di w.*

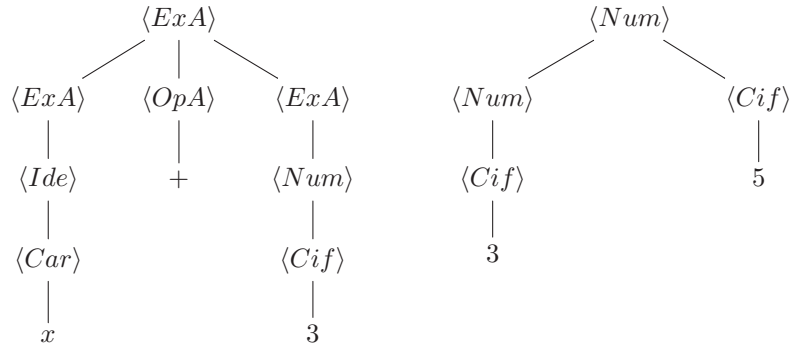
Osservazione 7.3.5. *Si noti che, utilizzando la terminologia del Capitolo 4 un parse tree è un albero* ordinale*: l'ordine dei figli conta.*

Per esempio, data la grammatica in (7.14), un albero di derivazione per la stringa *temp* è rappresentato di seguito.



Si noti che ogni nodo interno è etichettato da un simbolo non terminale (⟨*Ide*⟩ o ⟨*Car*⟩) ed ogni foglia è etichettata da un simbolo terminale. Leggendo le foglie da sinistra a destra si ha proprio la stringa *temp* (che potrebbe essere l'identificatore di una variabile temporanea).

Gli alberi di derivazione per le stringhe $x + 3$ e 35 sono rappresentati di seguito rispettivamente sulla sinistra e sulla destra.



Esercizio 7.3.6. *Data la grammatica libera da contesto in* (7.14)*, esiste un albero di derivazione per* $5 + \times \div$ *?*

Osservazione 7.3.7. *Un programma nella maggior parte dei linguaggi di programmazione è una stringa sull'alfabeto* `ASCII`*. L'interprete o il compilatore del linguaggio trasformano sempre questa stringa nel'albero di derivazione, per poi eseguirlo o tradurlo. Infatti la semantica dei linguaggi di programmazione è solitamente definita in modo ricorsivo, sugli alberi e non sulle stringhe.*

Lo stesso avviene, in modo implicito, quando manipoliamo espressioni matematiche o logiche. Il bravo informatico, quando riflette su un programma o su una espressione, deve sempre chiedersi come prima cosa qual'è l'albero corrispondente.

A questo punto possiamo definire formalmente qual'è il linguaggio generato da una categoria sintattica di una grammatica.

Definizione 7.3.8. *Sia* (S, P) *una grammatica libera da contesto sull'alfabeto* A*. Sia* $\langle X \rangle \in S$ *un simbolo non terminale. Il linguaggio generato da* $\langle X \rangle$ *è l'insieme delle stringhe* $w \in A^\star$ *tali che esiste un albero di derivazione per* w *avente come radice un nodo etichettato con* $\langle X \rangle$*. Denoteremo questo linguaggio con* $\langle\langle \langle X \rangle \rangle\rangle$*.*

Per esempio, la stringa $x + 3$ è un'espressione aritmetica, cioè appartiene al linguaggio generato da ⟨*ExA*⟩; la stringa 35 è un numero, cioè appartiene al linguaggio generato da ⟨*Num*⟩; la stringa *temp* è un identificatore (di variabile), cioè appartiene al linguaggio generato da ⟨*Ide*⟩;

Esercizio 7.3.9. *Data la grammatica libera da contesto in* (7.14)*, la stringa* 35 *appartiene al linguaggio generato da* ⟨*ExA*⟩*? E temp?*

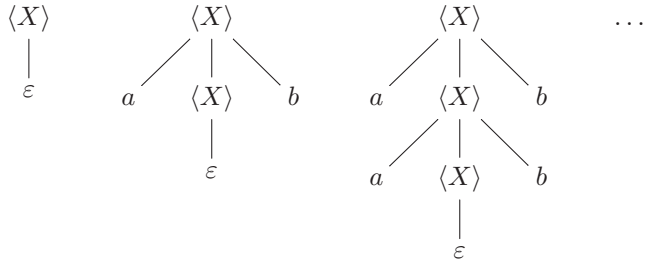
Esempio 7.3.10. *Ricordiamo dall'Osservazione 7.1.13, che per ogni stringa $w \in A^\star$, w^n è la concatenazione della stringa w n-volte. Prendiamo adesso l'alfabeto $A = \{a, b\}$ e consideriamo il linguaggio $L \subseteq A^\star$ definito come*

$$L = \{a^n b^n \mid n \in \mathbb{N}\}.$$

Per esempio si ha che $\varepsilon \in L$ (si prenda $n = 0$), $ab \in L$ (si prenda $n = 1$), $aabb \in L$ (si prenda $n = 2$) e $aaabbb \in L$ (si prenda $n = 3$). Si consideri adesso la seguene grammatica.

$$\langle X \rangle \quad \rightsquigarrow \quad \varepsilon \mid a\langle X \rangle b \tag{7.15}$$

È facile convincersi che il linguaggio generato da $\langle X \rangle$ è esattamente L: infatti per ogni $n \in \mathbb{N}$, la stringa $a^n b^n$ è testimoniata da un albero di derivazione analogo a quelli illustrati di seguito per $n = 0$, $n = 1$ e $n = 2$.



Viceversa, le stringhe $a^n b^n$ sono le uniche generabili da tale grammatica: tutti gli alberi di derivazione (e quindi tutte le stringhe generate) hanno esattamente la forma degli alberi di derivazione illustrati sopra.

Osservazione 7.3.11. *Ricordiamo al lettore che le spressioni erano state definite induttivamente nel Capitolo 5. È importante notare che anche la definizione di una grammatica può essere vista come una definizione induttiva. Consideriamo ad esempio, la grammatica in* (7.15)*: questa definisce $\langle\langle\langle X \rangle\rangle\rangle \subseteq A^\star$ induttivamente come:*

1. [Clausola base] $\varepsilon \in \langle\langle\langle X \rangle\rangle\rangle$.
2. [Clausola induttiva] *Se $w \in \langle\langle\langle X \rangle\rangle\rangle$, allora $awb \in \langle\langle\langle X \rangle\rangle\rangle$.*

Esempio 7.3.12. *Ci sono molte grammatiche che non fanno uso dell'induzione, ma sono comunque interessanti perchè specificano formati di stringhe che possono essere utili in svariati contesti. Un esempio tipico, ben noto al lettore, è la grammatica degli* indirizzi postali.

$$\begin{aligned}
\langle IndirizzoPostale \rangle &\rightsquigarrow \langle Destinatario \rangle \langle EOL \rangle \langle Indirizzo \rangle \langle EOL \rangle \langle Località \rangle \\
\langle Destinatario \rangle &\rightsquigarrow \langle Titolo \rangle \langle Nome \rangle \langle Cognome \rangle \\
\langle Indirizzo \rangle &\rightsquigarrow \langle Via \rangle \langle NumeroCivico \rangle \\
\langle Località \rangle &\rightsquigarrow \langle CAP \rangle \langle Comune \rangle (\langle Provincia \rangle) \\
\langle CAP \rangle &\rightsquigarrow \langle Cif \rangle \langle Cif \rangle \langle Cif \rangle \langle Cif \rangle \langle Cif \rangle
\end{aligned}$$

La categoria sintattica $\langle EOL \rangle$ stà per un qualche carattere di fine linea (dall'inglese End Of Line)

Esercizio 7.3.13. *Dare una grammatica libera da contesto che genera il linguaggio $\{a^n b^n \mid n \in \mathbb{N}^+\}$.*

Esercizio 7.3.14. *Dare una grammatica libera da contesto che genera il linguaggio $\{a^n c b^n \mid n \in \mathbb{N}\}$.*

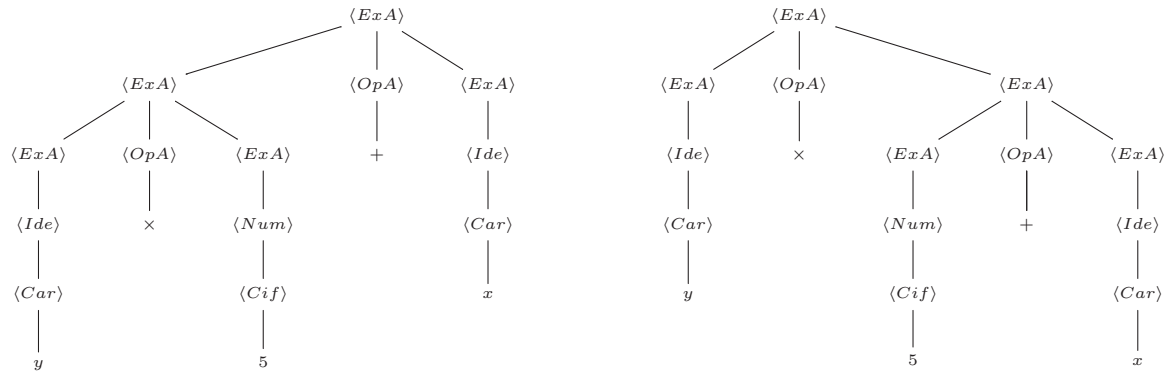
Esercizio 7.3.15. *Dare una grammatica libera da contesto che genera il linguaggio $\{a^{2n} b^n \mid n \in \mathbb{N}\}$.*

Esercizio 7.3.16. *Data la grammatica libera da contesto in* (7.9)*, qual'è il linguaggio generato da $\langle ExA \rangle$? E da $\langle Num \rangle$?*

Esercizio 7.3.17. *Data la grammatica libera da contesto in* (7.14)*, si disegnino, se esistono, uno o più alberi di derivazione per le stringhe $y \times 5 + x$, $x + y \div 0$ e $-5 \div x$.*

Ambiguità

Le grammatiche libere da contesto possono essere *ambigue* in un senso ben preciso che illustriamo di seguito. Continuiamo ad utilizzare come esempio il linguaggio delle espressioni aritmetiche e la grammatica in (7.14). Consideriamo la stringa $y \times 5 + x$. Ci sono due diversi alberi di derivazione per questa stringa che illustriamo di seguito.



Intuitivamente ai due alberi di derivazione corrispondono due diverse espressioni con parentesi: quello di sinistra corrisponde a $(y \times 5) + x$ e quello di destra a $y \times (5 + x)$. Si noti che la semantica delle due espressioni è molto diversa: per esempio per $x = 3$ e $y = 2$, la prima vale 13, mentre la seconda vale 16. Per le espressioni aritmetiche, si risolve questo problema introducendo una precedenza sugli operatori: si dice per esempio che $\times$ lega più di $+$. Quindi l'espressione aritmetica $y \times 5 + x$ viene considerata come $(y \times 5) + x$. Per esprimere l'espressione $y \times (5 + x)$ è necessario quindi introdurre le parentesi tonde. Espendiamo prima l'alfabeto con i simboli per le parentesi

$$A = \{+, -, \times, \div\} \cup \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\} \cup \{a, b, c, \ldots, z\} \cup \{(,)\}. \tag{7.16}$$

e poi la grammatica come segue.

$$\begin{array}{rcl} \langle OpA \rangle & \rightsquigarrow & + \mid - \mid \times \mid \div \\ \langle Cif \rangle & \rightsquigarrow & 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\ \langle Num \rangle & \rightsquigarrow & \langle Cif \rangle \mid \langle Num \rangle \langle Cif \rangle \\ \langle ExA \rangle & \rightsquigarrow & (\, \langle ExA \rangle \,) \mid \langle Num \rangle \mid \langle ExA \rangle \langle OpA \rangle \langle ExA \rangle \mid \langle Ide \rangle \\ \langle Ide \rangle & \rightsquigarrow & \langle Car \rangle \mid \langle Ide \rangle \langle Car \rangle \\ \langle Car \rangle & \rightsquigarrow & a \mid b \mid \cdots \mid z \end{array} \tag{7.17}$$

Osservazione 7.3.18. *Un problema analogo avviene anche con il linguaggio naturale. Per esempio, nella lingua italiana, la frase*

Gianni vede la vecchia col binocolo

può essere interpretata in due modi: il binocolo appartiene alla vecchia oppure a Gianni.

Gianni vede (la vecchia col binocolo)

oppure

(Gianni vede la vecchia) col binocolo

Tutto ciò ci fa riflettere sull'espressività degli alberi: in un certo senso, gli alberi contengono più "informazione" delle stringhe. Ma allora perchè storicamente l'umanità ha sviluppato tantissimi linguaggi scritti con stringhe e non con alberi?

A questo proposito è importante ricordare che, nel corso degli anni, gli informatici hanno sviluppato tanti linguaggi (di programmazione o di specifica) la cui sintassi sono alberi (come html o xml) o, addirittura grafi (come uml o tensor flow). L'idea dietro i linguaggi grafici è che, come i

diagrammi di Eulero-Venn o i diagrammi per le relazioni, la sintassi grafica/diagrammatica è in qualche senso più intuitiva. Forse, un giorno, programmeremo solo disegnando diagrammi e non digitando caratteri su una tastiera...

Definizione 7.3.19. *Una grammatica libera da contesto* (S, P) *è detta* ambigua *se esiste una stringa* $w \in A^\star$ *ed almeno due diversi alberi di derivazione che testimoniano* w *e che hanno la stessa etichetta nella radice.*

Per esempio, la grammatica in (7.17) è ambigua. Esistono delle tecniche molto eleganti per disambiguare le grammatiche, ma queste vanno al di la dello scopo di questo corso.

La grammatica di un semplice linguaggio imperativo

Mostriamo adesso la grammatica libera da contesto per un semplice linguaggio di programmazione. L'idea è di riutilizzare quanto abbiamo fatto fino ad ora per le espressioni aritmetiche ed estenderlo con le categorie sintattiche per comandi ($\langle Com \rangle$) ed espressioni booleane ($\langle ExB \rangle$).

Iniziamo con le espressioni booleane. Estendiamo prima l'alfabeto

$$A = \{+, -, \times, \div\} \cup \{0, \ldots, 9\} \cup \{a, b, c, \ldots, z\} \cup \{(,)\} \cup \{\wedge, \vee, !, \} \cup \{>, \geq, =, <, \leq\} \tag{7.18}$$

e poi diamo la seguente grammatica.

$$\begin{array}{lcl} \langle ExB \rangle & \rightsquigarrow & (\,\langle ExB \rangle\,) \mid \langle CoB \rangle \mid \langle ExB \rangle \langle OpB \rangle \langle ExB \rangle \mid !\langle ExB \rangle \mid \\ & & \langle Ide \rangle \mid \langle ExA \rangle \, \langle OpC \rangle \, \langle ExA \rangle \\ \langle OpC \rangle & \rightsquigarrow & > \mid \geq \mid < \mid \leq \mid = \mid \, ! = \\ \langle OpB \rangle & \rightsquigarrow & \wedge \mid \vee \\ \langle CoB \rangle & \rightsquigarrow & true \mid false \end{array} \tag{7.19}$$

In altre parole un'espressione booleana può essere un'espressione booleana tra parentesi tonde, oppure una costante booleana (categoria sintattica $\langle CoB \rangle$), cioè le stringhe $true$ e $false$, oppure l'and ($\wedge$) o l'or ($\vee$) di due espressioni booleane, il not (!) di un'espressione booleana, una variabile, oppure il *confronto* di due espressioni aritmetiche. Tale confronto è fatto attraverso gli operatori di confronto (categoria sintattica $\langle OpC \rangle$) che sono maggiore ($>$), maggiore o uguale ($\geq$), minore ($<$), minore o uguale ($\leq$), uguale ($=$) o diverso ($! =$).

Per i comandi, estendiamo ulteriormente l'alfabeto

$$A = \{+, -, \times, \div\} \cup \{0, \ldots, 9\} \cup \{a, b, c, \ldots, z\} \cup \{(,)\} \cup \{\wedge, \vee, !, \} \cup \{>, \geq, =, <, \leq\} \cup \{;, :, \{, \}\} \tag{7.20}$$

e consideriamo la seguente grammatica.

$$\begin{array}{lcll} \langle Com \rangle & \rightsquigarrow & skip \mid & \text{(Comando vuoto)} \\ & & \langle Ide \rangle := \langle ExA \rangle \mid & \text{(Assegnamento)} \\ & & \langle Com \rangle \, ; \, \langle Com \rangle \mid & \text{(Composizione)} \\ & & if\,(\,\langle ExB \rangle\,)\ then\ \{\,\langle Com \rangle\,\}\, else\ \{\langle Com \rangle\,\} \mid & \text{(Branching condizionale)} \\ & & while\,(\,\langle ExB \rangle\,)\ do\ \{\,\langle Com \rangle\,\} \mid & \text{(Iterazione)} \end{array} \tag{7.21}$$

Unendo la grammatica per le espressioni aritmetiche con quelle delle espressioni booleane e dei comandi, otteniamo la grammatica di un semplice linguaggio di programmazione.

$$
\begin{array}{rcl}
\langle OpA\rangle & \rightsquigarrow & + \mid - \mid \times \mid \div \\
\langle OpB\rangle & \rightsquigarrow & \wedge \mid \vee \\
\langle OpC\rangle & \rightsquigarrow & > \mid \geq \mid < \mid \leq \mid = \mid \,!= \\
\langle Cif\rangle & \rightsquigarrow & 0 \mid 1 \mid 2 \mid 3 \mid 4 \mid 5 \mid 6 \mid 7 \mid 8 \mid 9 \\
\langle CoB\rangle & \rightsquigarrow & true \mid false \\
\langle Car\rangle & \rightsquigarrow & a \mid b \mid \cdots \mid z \\
\langle Ide\rangle & \rightsquigarrow & \langle Car\rangle \mid \langle Ide\rangle\,\langle Car\rangle \\
\langle Num\rangle & \rightsquigarrow & \langle Cif\rangle \mid \langle Num\rangle\,\langle Cif\rangle \\
\langle ExA\rangle & \rightsquigarrow & (\,\langle ExA\rangle\,) \mid \langle Num\rangle \mid \langle ExA\rangle\langle OpA\rangle\langle ExA\rangle \mid \langle Ide\rangle \\
\langle ExB\rangle & \rightsquigarrow & (\,\langle ExB\rangle\,) \mid \langle CoB\rangle \mid \langle ExB\rangle\langle OpB\rangle\langle ExB\rangle \mid !\langle ExB\rangle \mid \\
 & & \langle Ide\rangle \mid \langle ExA\rangle\,\langle OpC\rangle\,\langle ExA\rangle \\
\langle Com\rangle & \rightsquigarrow & skip \mid \\
 & & \langle Ide\rangle := \langle ExA\rangle \mid \\
 & & \langle Com\rangle\,;\,\langle Com\rangle \mid \\
 & & if\,(\,\langle ExB\rangle\,)\ then\ \{\,\langle Com\rangle\,\}\,else\ \{\langle Com\rangle\,\} \mid \\
 & & while\,(\,\langle ExB\rangle\,)\ do\ \{\,\langle Com\rangle\,\}
\end{array}
\tag{7.22}
$$

Esercizio 7.3.20. *Data la grammatica libera da contesto in* (7.22)*, si disegnino, se esistono, uno o più alberi di derivazione per le stringhe*

1. $while\,(\,true\,)\,do\,\{skip\};$
2. $if\,(\,x < 0\,)\,then\,\{x := 3\}\,else\,\{skip\};$
3. $x := 3\,;\ y := 5\,;\ z := x + y.$

Esercizio 7.3.21. *Data la grammatica libera da contesto in* (7.22)*, la stringa* $5 + 4 > 3$ *è una espressione aritmetica? Cioè, appartiene al linguaggio generato da* $\langle ExA\rangle$ *?*

7.4 Uno studio comparativo di automi e grammatiche

Nella Sezione 7.1, abbiamo introdotto i linguaggi come sottoinsiemi di $A^\star$. Abbiamo visto nella Sezione 7.2 che alcuni linguaggi possono essere riconosciuti da automi, mentre nella Sezione 7.3 che alcuni linguaggi possono essere generati da grammatiche libere da contesto. In particolare, abbiamo mostrato che le grammatiche possono essere utilizzate per la specifica della sintassi dei linguaggi di programmazione. Vale lo stesso per gli automi? Cioè, possono gli automi a stati finiti riconoscere i linguaggi di programmazione?

Come da buona pratica scientifica, per dare risposta a questa domanda, ci poniamo in una prospettiva più ampia e ci chiediamo:

1. Tutti i linguaggi generati da grammatiche finite sono anche riconoscibili da automi a stati finiti?

Se la risposta a questa domanda fosse affermativa, allora sapremmo che, in particolare, i linguaggi di programmazione sono riconoscibili da automi (visto che sono generati da grammatiche). Prima di provare a rispondere a questa domanda, ci interroghiamo sulla domanda opposta:

2. Tutti i linguaggi riconoscibili da automi a stati finiti sono generati da grammatiche finite?

La risposta alla seconda domanda è positiva. Per dimostrarlo, illustriamo una costruzione che prende come input un automa e restituisce una grammatica che riconosce lo stesso linguaggio.

L'idea è molto semplice. Dato un automa $\mathcal{A} = (S, T, F)$ definiamo una grammatica $\mathcal{G}_\mathcal{A} = (S_\mathcal{A}, P_\mathcal{A})$ dove l'insieme dei simboli non terminali $S_\mathcal{A}$ è esattamente l'insieme contenente tutti e soli i simboli $\langle x \rangle$ per ogni stato dell'automa $x \in S$. Più precisamente

$$S_\mathcal{A} = \{\langle x \rangle \mid x \in S\}. \tag{7.23}$$

Ad esempio, prendendo come $\mathcal{A}$ l'automa in (7.4), la grammatica $\mathcal{G}_\mathcal{A}$ corrispondente ha come insieme di simboli non terminali $S_\mathcal{A} = \{\langle x \rangle, \langle y \rangle, \langle z \rangle, \langle u \rangle\}$.

Le produzioni in $P_\mathcal{A}$ sono costruite a partire dalla relazione di transizione T e l'insieme degli stati finali F. In particolare, per ogni stato finale $x \in F$, c'è una produzione

$$\langle x \rangle \rightsquigarrow \varepsilon$$

in $P_\mathcal{A}$. Per ogni transizione $((a, x), y) \in T$, c'è la produzione

$$\langle x \rangle \rightsquigarrow a\langle y \rangle.$$

In sintesi, l'insieme delle produzioni è

$$P_\mathcal{A} = \{\langle x \rangle \rightsquigarrow \varepsilon \mid x \in F\} \cup \{\langle x \rangle \rightsquigarrow a\langle y \rangle \mid ((a, x), y) \in T\}. \tag{7.24}$$

Ad esempio, prendendo come $\mathcal{A}$ l'automa in (7.4), la grammatica $\mathcal{G}_\mathcal{A}$ ha come produzioni

$$\begin{array}{lcl}
\langle u \rangle & \rightsquigarrow & \varepsilon \\
\langle z \rangle & \rightsquigarrow & e\langle u \rangle \\
\langle y \rangle & \rightsquigarrow & r\langle z \rangle \\
\langle x \rangle & \rightsquigarrow & a\langle y \rangle \\
\langle x \rangle & \rightsquigarrow & a\langle x \rangle \mid b\langle x \rangle \mid \cdots \mid z\langle x \rangle
\end{array}$$

Tale costruzione è riassunta nella seguente definizione.

Definizione 7.4.1. *Dato un qualsiasi automa $\mathcal{A} = (S, T, F)$, la corrispondente grammatica libera da contesto è $\mathcal{G}_\mathcal{A} = (S_\mathcal{A}, P_\mathcal{A})$ dove, l'insieme dei simboli non terminali $S_\mathcal{A}$ e l'insieme delle produzioni $P_\mathcal{A}$ sono definiti rispettivamente come in* (7.23) *e* (7.24)*.*

Per maggiore chiarezza, illustriamo un ulteriore esempio.

Esempio 7.4.2. *La grammatica corrispondente all'automa in* (7.3) *è*

$$\begin{array}{lcl}
\langle z \rangle & \rightsquigarrow & \varepsilon \mid a\langle z \rangle \mid a\langle x \rangle \mid b\langle x \rangle \\
\langle y \rangle & \rightsquigarrow & a\langle x \rangle \mid b\langle z \rangle \\
\langle x \rangle & \rightsquigarrow & a\langle x \rangle \mid b\langle x \rangle
\end{array}$$

Esercizio 7.4.3. *Per ognuno dei tre automi in* (7.2)*, costruire la corrispondente grammatica libera da contesto.*

La proprietà fondamentale di tale costruzione è che preserva il linguaggio accettato. Questo significa che una stringa $w \in A^\star$ è accettata dallo stato $x \in S$ (in simboli $w \in \langle\langle x \rangle\rangle$) se e solo se w appartiene al linguaggio generato dalla categoria sintattica $\langle x \rangle$ della grammatica corrispondente (in simboli $w \in \langle\langle\langle x \rangle\rangle\rangle$).

Teorema 7.4.4. *Per tutti gli automi $\mathcal{A} = (S, T, F)$, vale che per tutte le parole $w \in A^\star$ e tutti gli stati $x \in S$*

$$w \in \langle\langle x \rangle\rangle \text{ se e solo se } w \in \langle\langle\langle x \rangle\rangle\rangle.$$

Lasciamo la dimostrazione di questo teorema come esercizio per il lettore.

Esercizio 7.4.5. *Dimostrare che per qualsiasi grammatica $\mathcal{G}_\mathcal{A} = (S_\mathcal{A}, P_\mathcal{A})$ corrispondente ad un automa $\mathcal{A}$ vale che per tutti gli $\langle x \rangle \in S_\mathcal{A}$, $a \in A$ e $w \in A^\star$:*

1. *$\varepsilon \in \langle\langle\langle x \rangle\rangle\rangle$ se e solo se $\langle x \rangle \rightsquigarrow \varepsilon \ \in P_\mathcal{A}$;*
2. *$aw \in \langle\langle\langle x \rangle\rangle\rangle$ se e solo se $\langle x \rangle \rightsquigarrow a\langle y \rangle \ \in P_\mathcal{A}$ e $w \in \langle\langle\langle y \rangle\rangle\rangle$ per qualche $\langle y \rangle \in S_\mathcal{A}$.*

Esercizio 7.4.6. *Dimostrare il Teorema 7.4.4. Si suggerisce di usare l'induzione su $w \in A^\star$ e i due punti dell'Esercizio 7.2.16 e i due punti dell'Esercizio 7.4.5*

Se $\mathcal{A}$ è a stati finiti allora anche $\mathcal{G}_\mathcal{A}$ è finita e, pertanto, la seconda domanda ha risposta positiva.

La prima domanda invece ha risposta negativa. Un controesempio è fornito dal linguaggio generato dalla grammatica nell'Esempio 7.3.10. Come illustrato di seguito questo linguaggio non può essere riconosciuto da alcun automa a stati finiti.

Teorema 7.4.7. *Sia $A = \{a, b\}$ e sia $L = \{a^n b^n \mid n \in \mathbb{N}\} \subseteq A^\star$. Non esiste alcun automa a stati finiti $\mathcal{A} = (S, T, F)$ e alcuno stato $x \in S$ tale che $\langle\langle x \rangle\rangle = L$.*

Dimostrazione. La dimostrazione procede per assurdo: assumiamo la negazione della tesi, per mostrare che si giunge ad una contraddizione.

Supponiamo per assurdo che esista un automa $\mathcal{A} = (S, T, F)$ sull'alfabeto A dove

1. esiste un $x \in S$ tale che $\langle\langle x \rangle\rangle = L$;
2. l'insieme di stati S ha una qualche cardinalità finita m.

Visto che m è un numero naturale, si ha che per definizione di L, la stringa $a^m b^m$ appartiene al linguaggio L.

Per definizione di linguaggio accettato, vale che esistono $y \in S$ e $z \in F$ tali che $(x, y) \in T_{a^m}$ e $(y, z) \in T_{b^m}$.

Per definizione di T_{a^m} si ha che esistono $x_i \in S$ per $i \in \{0, \ldots, m\}$ tali che $x = x_0$, $y = x_m$ e $((a, x_i), x_{i+1})$ per $i \in \{0, \ldots m-1\}$. In altre parole deve esistere un walk

$$x = x_0 \xrightarrow{a} x_1 \xrightarrow{a} \ldots \xrightarrow{a} x_{m-1} \xrightarrow{a} x_m = y \ .$$

Si noti che la sequenza di x_i definisce una funzione $f\colon \{0, \ldots, m\} \to S$ che mappa ogni i in x_i. Visto che $|\{0, \ldots, m\}| = m + 1$ e $|S| = m$, allora la funzione f non è iniettiva per *il principio delle buche dei piccioni* (Osservazione 6.2.6). Pertanto esistono due elementi j e k dell'insieme $\{0, \ldots, m\}$ tali che $j \neq k$ e $x_j = x_k$. Visto che $j \neq k$ sappiamo che o $j < k$ o $k < j$. Assumiamo che $j < k$ (il caso $k > j$ è del tutto analogo).

Dal momento che $x_j = x_k$ possiamo costruire il walk

$$x = x_0 \xrightarrow{a} x_1 \xrightarrow{a} \ldots \xrightarrow{a} x_j \xrightarrow{a} x_{k+1} \xrightarrow{a} \ldots \xrightarrow{a} x_{m-1} \xrightarrow{a} x_m = y$$

e pertanto $(x, y) \in T_{a^o}$ dove $o = j + m - k$ (si noti che vale sempre $o < m$). Utilizzando adesso che $(y, z) \in T_{b^m}$, si ha che $(x, z) \in T_{a^o b^m}$ e visto che $z \in F$, la stringa $a^o b^m$ appartiene al linguaggio di x (in simboli $a^o b^m \in \langle\langle x \rangle\rangle$). Ma questo va contro l'ipotesi 1. che $\langle\langle x \rangle\rangle = \{a^n b^n \mid n \in \mathbb{N}\}$ dal momento che $o < m$.

Abbiamo raggiunto una contraddizione e quindi dimostrato il teorema. ■

Osservazione 7.4.8. *Il Teorema 7.4.7 ci dice che $\{a^n b^n \mid n \in \mathbb{N}\}$ non può essere riconosciuto da un automa a stati finiti. Utilizzando un argomento simile, si può dimostrare che anche il linguaggio delle* parentesi bilanciate *sull'alfabeto $\{(,)\}$ non può essere riconosciuto da un automa a stati finiti. Tale linguaggio consiste di tutte le stringhe di parentesi, possibilmente annidate, che si aprono e chiudono lo stesso numero di volte. Sono ad esempio stringhe di questo linguaggio* ((())) *e* ((()()(()))) *ma non* (((. *Questo esempio suggerisce che anche la sintassi della maggior parte dei linguaggi di programmazione non può essere riconosciuta da automi a stati finiti.*

Per questa ragione, la sintassi dei linguaggi di programmazione è specificata solitamente attraverso grammatiche, mentre gli automi sono utilizzati per la così detta analisi lessicale. *In estrema sintesi, l'analisi lessicale legge il codice sorgente un carattere alla volta e lo traduce in lessemi (token). Lo studente potrà approfondire questi argomenti in corsi più avanzati.*

I Teoremi 7.4.4 e 7.4.7 ci dicono assieme una cosa molto importante: il primo stabilisce che le grammatiche sono *espressive almeno quanto* gli automi, il secondo che sono *strettamente più espressive* degli automi. Questo significa che l'insieme dei linguaggi riconosciuti dagli automi (a stati finiti) è *strettamente incluso* nell'insieme dei linguaggi generati da grammatiche (finite). I linguaggi

del primo tipo sono detti *regolari* e quelli del secondo sono detti *liberi da contesto*. Queste due tipologie di linguaggi formano le prime due classi della ben nota gerarchia di Chomsky, riportata in Figura 7.3. Lo studente ineteressato avrà modo di conoscere ulteriori classi della gerarchia in corsi più avanzati.

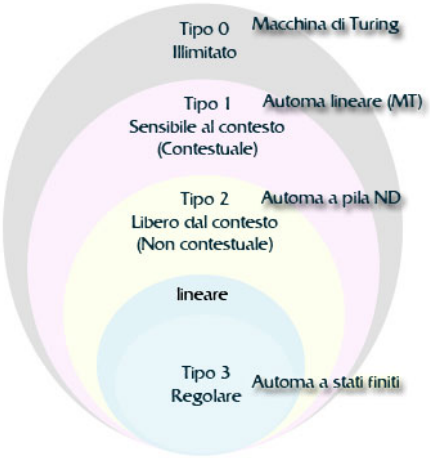


Figura 7.3: La Gerarchia di Chomsky.

7.5 Espressioni Regolari

In Figura 7.3 la parola "regolare" è utilizzata per denotare i linguaggi riconoscibili da automi a stati finiti. Questi linguaggi sono chiamati così in quanto godono di una caratterizzazione algebrica in termini di *espressioni regolari*. In questa sezione introduciamo le espressioni regolari come un primo esempio di linguaggio formale: illustriamo la loro sintassi (data attraverso una grammatica libera da contesto), la loro semantica (definita per induzione) ed enunciamo il sopramenzionato risultato di caratterizzazione.

Le espressioni regolari sono delle espressioni per denotare linguaggi su un alfabeto fissato A. Le espressioni regolari sono usate in molti contesti applicativi: nei motori di ricerca, nei comandi *search and replace* negli editor di testo, in molte utilities di text processing e nell'analisi lessicale. Inoltre molti linguaggi di programmazione provvedono librerie per le espressioni regolari.

Sintassi

Come gli automi e le grammatiche, anche le espressioni regolari sono definite per un alfabeto A fissato. Le espressioni regolari sono certe stringhe usll'alfabeto A esteso con gli operatori $\{+, \cdot, ^\star\}$ e le costanti $\{0, 1\}$. Cioè, l'alfabeto delle espressioni regolari è

$$A \cup \{+, \cdot, ^\star\} \cup \{0, 1\} \tag{7.25}$$

In queste note, come nella maggior parte delle presentazioni delle espressioni regolari, si assume che A sia disgiunto da $\{+, \cdot, ^\star\} \cup \{0, 1\}$, ovvero che $+, \cdot, ^\star, 0, 1$ siano in qualche senso caratteri speciali che non appartengono ad A.

In modo del tutto analogo alle espressioni aritmetiche, non tutte le stringhe sull'alfabeto (7.25) sono espressioni regolari. Ad esempio per $A = \{a, b\}$, $a+$ e $a^\star b$ non sono espressioni regolari mentre $a + 1$, $a \cdot b^\star$ e $a \cdot a \cdot b$ lo sono. Per definire in modo preciso cosa sia un'espressione regolare è conveniente utilizzare la seguente grammatica libera da contesto.

$$\begin{array}{rcll} \langle RExp_A \rangle & \rightsquigarrow & a \mid & (\text{ se } a \in A) \\ & & 0 \mid & \\ & & 1 \mid & \\ & & \langle RExp_A \rangle + \langle RExp_A \rangle \mid & \\ & & \langle RExp_A \rangle \cdot \langle RExp_A \rangle \mid & \\ & & \langle RExp_A \rangle^\star & \end{array} \tag{7.26}$$

La prima produzione $\langle RExp_A \rangle \rightsquigarrow a$ va pensata come tante (esattamente $|A|$), produzioni. Ad esempio se $A = \{a, b\}$, allora si hanno due produzioni: $\langle RExp_A \rangle \rightsquigarrow a$ e $\langle RExp_A \rangle \rightsquigarrow b$. In estrema sintesi, tale produzione ci dice che ogni simbolo in A è una espressione regolare. La seconda e la terza produzione ci dice che anche 0 e 1 sono espressioni regolari. La quarta e la quinta produzione ci dice che se e_1 ed e_2 sono espressioni regolari, allora anche $e_1 + e_2$ ed $e_1 \cdot e_2$ sono espressioni regolari, mentre la sesta ci dice che se e è un'espressione regolare allora anche $e^\star$ è un'espressione regolare.

Definizione 7.5.1. *Una espressione regolare è una stringa che appartiene a* $\langle\langle \langle RExp_A \rangle \rangle\rangle$. *Utilizziamo* $RExp_A$ *per l'insieme di tutte le espressioni regolari. Cioè* $Rexp_A = \langle\langle \langle RExp_A \rangle \rangle\rangle$.

Esempio 7.5.2. *Fissiamo* $A = \{a, b, c\}$ *e consideriamo le seguente stringhe.*

- $0 + 1$ *è una espressione regolare*
- $1 + 1$ *è una espressione regolare*
- $1 \cdot b$ *è una espressione regolare*
- $1 \cdot b^\star$ *è una espressione regolare*
- $1 \cdot b^\star + a \cdot b^\star$ *è una espressione regolare*
- $+\cdot^\star$ *non è una espressione regolare*

Esercizio 7.5.3. *Per le prime cinque stringhe in Example 7.5.2 costruire tutti gli alberi di derivazione sintattica rispetto alla grammtica in* (7.26).

Prima di considerare la semantica di queste espressioni è opportuno considerare ancora la loro sintassi. Come per le espressioni aritmetiche, la grammatica per le espressioni regolari è *ambigua*. Infatti, il lettore che ha svolto correttamente l'esercizio precedente, ha potuto constatare che per per la stringa $1 \cdot b^\star + a \cdot b^\star$ esiste più di un parse tree. Per definire delle espressioni regolari è necessario risolvere questa ambiguità. Come per le espressioni aritmetiche, si risolve questo problema introducendo una precedenza sugli operatori: si impone che * lega più di $\cdot$ e che $\cdot$ lega più di $+$. Queste precedenze sono riassunte nella seguente tabella.

operatore	livello di precedenza
$+$	0
$\cdot$	1
$\star$	2

Si consideri ad esempio l'espressione $a + 1 \cdot b$. Tale espressione è testimoniata da due diversi parse trees: in uno l'operatore più vicino alla radice è $+$ e nell'altro l'operatore più vicino alla radice è il $\cdot$. Intuitivamente, il primo corrisponde a $a + (1 \cdot b)$, mentre il secondo a $(a + 1) \cdot c$. I livelli di precedenza ci dicono che quando ci troviamo di fronte all'espressione $a + 1 \cdot b$, dobbiamo considerarla come generata dal parse tree $a + (1 \cdot b)$ visto che $\cdot$ lega di più (cioè ha precedenza più alta) di $+$. Per indicare l'espressione generata dal secondo parsee tree, utilizzaremo come sempre le *parentesi tonde*. In queste note evitiamo di redefinire la grammatica con le parentesi ed ogni volta che le incontriamo ci riferiamo al significato appena illustrato.

Esempio 7.5.4. *Fissiamo* $A = \{a, b, c\}$ *e consideriamo le seguente stringhe.*

- $0 + 1 \cdot a$ *è* $0 + (1 \cdot a)$ *e non* $(0 + 1) \cdot a$.

- $c + a^\star$ *è* $c + (a^\star)$ *e non* $(c + a)^\star$.
- $c \cdot b^\star$ *è* $c \cdot (b^\star)$ *e non* $(c \cdot b)^\star$.

Dopo aver definito la semantica sarà chiaro che gli operatori + e · sono associativi e pertanto espressioni come $a + b + c$ o $a \cdot b\cdot$ hanno entrambe due diversi parse trees ciascuna (corrispondenti a $a + (b + c)$ e $(a + b) + c$, rispettivamente, $a \cdot (b \cdot c)$ e $(a \cdot b) \cdot c$), ma i due hanno lo stesso significato. Pertanto, possiamo evitare di mettere parentesi in queste situazioni.

Semantica

La semantica per le espressioni regolari assegna ad ogni espressione $e \in RExp_A$ un linguaggio in $\mathcal{P}(A^\star)$. Intuitivamente:

- a denota il linguaggio $\{a\}$
- 0 denota il linguaggio vuoto $\varnothing$
- 1 denota il linguaggio $\{\varepsilon\}$
- + esegue l'unione dei linguaggi ($\cup$)
- · esegue la concatenazione di linguaggi ($\cdot$)
- * esegue la stella di Kleene di linguaggi (*)

Si ricorda al lettore che queste operazioni sui linguaggi (e le loro proprietà algebriche) sono state studiate alla fine della Sezione 7.1.

Esempio 7.5.5. *Consideriamo a livello intuitivo alcuni esempi sull'alfabeto* $A = \{a, b, c\}$.

- $a \cdot b$ *denota il linguaggio* $\{a\} \cdot \{b\} = \{ab\}$.
- $a \cdot b + c$ *denota il linguaggio* $(\{a\} \cdot \{b\}) \cup \{c\} = \{ab\} \cup \{c\} = \{ab, c\}$.
- $a \cdot (b + c)$ *denota il linguaggio* $a \cdot (\{b\} \cup \{c\}) = a \cdot \{b, c\} = \{ab, ac\}$.
- $(a \cdot b) + 1$ *denota il linguaggio* $(\{a\} \cdot \{b\}) \cup \{\varepsilon\} = \{ab\} \cup \{\varepsilon\}$.
- $(a \cdot b)^\star + 1$ *denota il linguaggio* $(\{a\} \cdot \{b\})^\star \cup \{\varepsilon\} = \{ab\}^\star \cup \{\varepsilon\} = \{\varepsilon, ab, abab, ababab, \dots\}$.

Una volta stabilita un'intuizione preliminare è opportuno definire la semantica in maniera *formale*, cioè come una funzione dal dominio sintattico (in questo caso le espressioni regolari) a quello semantico (in questo caso i linguaggi).

Definizione 7.5.6. *La semantica delle espressioni regolari è una funzione* $\mathcal{S}\colon RExp_A \to \mathcal{P}(A^\star)$ *definita ricorsivamete per ogni* $e \in RExp_A$ *come*

$$\mathcal{S}(e) = \begin{array}{ll} \{a\} & se\ e = a \\ \varnothing & se\ e = 0, \\ \{\varepsilon\} & se\ e = 1 \\ \mathcal{S}(e_1) \cup \mathcal{S}(e_2) & se\ e = e_1 + e_2 \\ \mathcal{S}(e_1) \cdot \mathcal{S}(e_2) & se\ e = e_1 \cdot e_2 \\ \mathcal{S}(e_1)^\star & se\ e = e_1^\star \end{array}$$

Illustriamo tale definizione con il seguente esempio.

$$\begin{array}{rcl} \mathcal{S}(\,(a \cdot b)^\star + c\,) & = & \mathcal{S}(\,(a \cdot b)^\star\,) \cup \mathcal{S}(\,c\,) \\ & = & \mathcal{S}(\,a \cdot b\,)^\star \cup \mathcal{S}(\,\mathbf{c}\,) \\ & = & (\mathcal{S}(\,a\,) \cdot \mathcal{S}(\,b\,))^\star \cup \mathcal{S}(\,\mathbf{c}\,) \\ & = & (\{a\} \cdot \{b\})^\star \cup \{c\} \\ & = & \{ab\}^\star \cup \{c\} \\ & = & \{\epsilon, ab, abab, ababab, \dots c\} \end{array}$$

Nella prima uguaglianza di sopra si prende e come $(a \cdot b)^\star + c$ e si utilizza la quarta clausola della Definizione 7.5.6 dove e_1 è $(a \cdot b)^\star$ ed e_2 è c. La definizione ci dice che la semantica di $(a \cdot b)^\star + c$ è esattamente l'unione della semantica di $(a \cdot b)^\star$ e della semantica di c.

Nella seconda uguaglianza, si prende e come $(a \cdot b)^\star$ e si utilizza la sesta clausola della Definizione 7.5.6 dove e_1 è $a \cdot b$. La definizione ci dice che la semantica di $(a \cdot b)^\star$ è esattamente la stella di Kleene del linguaggio ottenuto come semantica di $a \cdot b$.

Nella terza uguaglianza si procede in maniera analoga, ma utilizzando la quinta clausola. Nella quarta uguaglianza si utilizza contemporaneamente (per tre volte) la prima clausola della Definizione 7.5.6. Nella quinta e nella sesta uguaglianza si eseguono semplicemente le operazioni sui linguaggi, come definito alla fine della Sezione 7.1.

Esercizio 7.5.7. *Calcolare la semantica delle espressioni $a \cdot (b^\star)$, di $(a \cdot b)^\star$, di $a \cdot (b + c)$, di $(a \cdot b) + (a \cdot c)$ e di $(a \cdot b) + c$.*

Osservazione 7.5.8. *La semantica di molti linguaggi formali viene spesso definita in maniera analoga a quella della Definizione 7.5.6: ogni operatore* $\mathbf{op}$ *della sintassi corrisponde ad un operazione* $\overline{\mathbf{op}}$ *sul dominio semantico e la semantica di una espessione del tipo $e_1 \,\mathbf{op}\, e_2$ viene definita come $\mathcal{S}(e_1 \,\mathbf{op}\, e_2) = \mathcal{S}(e_1)\overline{\mathbf{op}}\mathcal{S}(e_2)$. Questa tipologia di definizione della semantica è spesso chiamata* semantica denotazionale*.*

Teorema di Corrispondenza

All'inizio di questa sezione abbiamo annunciato che vale una corrispondenza tra le espressioni regolari e gli automi a stati finiti. Tale corrispondenza è ben nota come teorema di Kleene. Prima introduciamo l'ultima definizione.

Definizione 7.5.9. *Un linguaggio $L \in \mathcal{P}(A^\star)$ è detto* regolare *se esiste una espressione regolare $e \in RExp_A$ che denota L, cioè tale che*

$$\mathcal{S}(e) = L.$$

Concludiamo con l'enunciato del teorema di Kleene.

Teorema 7.5.10 (Kleene)**.** *Sia A un alfabeto e $L \in \mathcal{P}(A^\star)$ un linguaggio su A. L è regolare se e solo se esiste un automa a stati finiti $\mathcal{A} = (S, T, F)$ ed uno stato $x \in S$ tale che $\langle\langle x \rangle\rangle = L$.*

CAPITOLO 8

Cenni di Logica Matematica

Nei nostri studi matematici di ogni livello, e anche nello studio di altre discipline, abbiamo incontrato innumerevoli risultati (chiamati teoremi, lemmi, proposizioni, corollari, ...), e talvolta di questi risultati ci è stata presentata una dimostrazione. Le regole della logica permettono di stabilire con precisione il significato di un enunciato matematico, rimediando a possibili ambiguità e imprecisioni del linguaggio naturale. Inoltre tali regole consentono di distinguere argomentazioni matematiche valide da quelle non valide, e quindi di capire se una dimostrazione proposta è corretta oppure no. L'obiettivo della logica non è tanto quello di stabilire la validità assoluta di un certo enunciato, ma piuttosto quello di stabilire quando, assumendo vere certe premesse, si possa affermare la validità di altri enunciati: in questo caso diremo che questi ultimi sono *conseguenze logiche* delle premesse.

Per questo motivo la logica è la base di ogni ragionamento matematico, ma essa ha anche numerose applicazioni in diversi campi dell'informatica: la progettazione di computer, l'intelligenza artificiale, l'ingegneria del software, la programmazione e tanti altri. Per esempio si diffonde sempre di più l'uso di tecniche di dimostrazione per certificare che un certo hardware o software si comporterà sempre correttamente.

Data la complessità dell'attività deduttiva e inferenziale degli esseri umani sono state sviluppate diverse logiche che si propongono di catturare aspetti sempre più complessi del ragionamento. Noi ci limitiamo a considerare le *logiche classiche*, che trattano enunciati che possono assumere uno e uno solo dei valori booleani dell'insieme $Bool = \{\mathtt{t}, \mathtt{f}\}$, introdotto nella Sezione 1.1. Useremo i valori di *Bool* solo per la semantica delle formule, mentre useremo una notazione diversa per rappresentare tali valori nella formule.

Notazione 8.0.1 (rappresentazione sintattica dei valori booleani)**.** *Per gli elementi dell'insieme* $Bool = \{\mathtt{t}, \mathtt{f}\}$ *useremo i simboli* T *(per* $\mathtt{t}$*) e* F *(per* $\mathtt{f}$*) per la sintassi, cioè per la loro rappresentazione simbolica nelle formule.*

Definizione 8.0.2 (proposizioni)**.** *Una* PROPOSIZIONE *è un enunciato dichiarativo (per esempio una frase in linguaggio naturale) che "afferma qualcosa" e per il quale si può dire:*

Principio del terzo escluso: che è vero oppure è falso (non ci sono altre possibilità)

Principio di non contradditorietà: che non è al tempo stesso sia vero che falso.

Per esempio, *"Quanti sono i numeri primi minori di 100?"* non è una proposizione, perché non "afferma qualcosa". Al contrario, *"Sono biondo naturale"* è una proposizione, che può essere vera o falsa a seconda del soggetto che la legge.

Esempio 8.0.3 (riconoscere le proposizioni)**.** *Vediamo per ognuna delle frasi seguenti se sono proposizioni oppure no.*

- Firenze è la capitale d'Italia.
  Si tratta di una proposizione, in quanto afferma un fatto che può essere vero o falso, ma non entrambi. In particolare, la proposizione è falsa per il significato usuale che associamo alle parole Firenze, capitale *e* Italia.
- È possibile che Firenze sia la capitale d'Italia.
  Non afferma qualcosa, ma esprime una valutazione sul fatto che l'enunciato Firenze è la capitale d'Italia *sia vero, pertanto non è una proposizione.*

- Firenze è la capitale d'Italia?
  Si tratta di un enunciato interrogativo che non afferma qualcosa, quindi non è una proposizione.

- Firenze è stata la capitale d'Italia.
  È una proposizione perché afferma un fatto che, in particolare, è vero.

- Se solo Firenze fosse la capitale d'Italia!
  È un enunciato che esprime un desiderio ma non afferma qualcosa, quindi non è una proposizione.

- Probabilmente Firenze è la capitale d'Italia.
  Non è una proposizione per gli stessi motivi della seconda frase vista sopra.

Le proposizioni semplici, che rappresenteremo astrattamente con lettere come $A, B, C, \ldots$, possono essere composte per formare proposizioni più complesse usando i *connettivi logici*, alcuni dei quali li abbiamo già incontrati nella Sezione 1.3 (*and*, *or* e *not*). Nella Sezione 8.1, presentando la sintassi e la semantica del Calcolo Proposizionale, introdurremo questi connettivi come operatori algebrici, e descriveremo come ottenere il valore di verità di proposizioni complesse a partire da quello dei componenti. Nella Sezione 8.2 presenteremo le tavole di verità e il concetto di tautologia, e discuteremo della correttezza di semplici inferenze logiche. La Sezione 8.3 è dedicata al problema di dimostrare tautologie nel calcolo proposizionale. Vedremo come si possono usare dimostrazioni per sostituzione a questo scopo, e inquadreremo questo tipo di dimostrazioni nel contesto più generale dei Sistemi di Dimostrazioni (o Proof Systems), che forniscono una definizione formale di dimostrazione. Infine discuteremo della correttezza di alcune classiche tecniche di dimostrazione.

Nella Sezione 8.5 introdurremo la Logica dei Predicati, una logica più espressiva del calcolo proposizionale che grazie ai quantificatori e ai termini consente di esprimere asserti che menzionano esplicitamente elementi di un *dominio*, cioè un insieme che in casi concreti può contenere numeri, persone, o entità di altro tipo. Anche per la logica dei predicati presenteremo la sintassi e la semantica. Inoltre introdurremo delle leggi che riguardano i quantificatori, che ci permetteranno di dimostrare la validità di formule usando dimostrazioni per sostituzione.

Gli aspetti della logica matematica cui siamo interessati in questo capitolo non forniscono strumenti per dimostrare automaticamente dei teoremi o per "creare" ragionamenti o inferenze complesse, ma ci permettono di *analizzare* dimostrazioni, ragionamenti e inferenze per controllare che siano logicamente corretti. Per analizzare logicamente un ragionamento occorre *formalizzare* le proposizioni che esso contiene, cioè renderne esplicita la struttura logica: questo serve per controllare se la conclusione è una conseguenza logica delle premesse e per riconoscere ed escludere ragionamenti sbagliati che porterebbero a conclusioni erronee. Sia per il calcolo proposizionale che per la logica dei predicati discuteremo come formalizzare delle semplici proposizioni espresse in linguaggio naturale in formule logiche, e come verificare che semplici inferenze siano corrette.

8.1 Il Calcolo Proposizionale: sintassi e semantica

Il *calcolo proposizionale* (chiamato anche *logica proposizionale*) costituisce il nucleo di tutte le logiche classiche. In questa sezione mostriamo come scrivere (la sintassi) e come interpretare (la semantica) delle formule proposizionali, anche con l'ausilio delle tavole di verità introdotte nella Sezione 8.2. Successivamente, nella Sezione 8.3 presenteremo le leggi algebriche che descrivono le proprietà dei connettivi logici e discuteremo come utilizzare le dimostrazioni per sostituzione per dimostrare l'equivalenza logica di formule proposizionali. Questa tecnica può essere usata per dimostrare che certe formule sono *tautologie*, cioè sempre vere.

Ricordiamo che, nonostante la sua semplicità, il calcolo proposizionale ha svariate applicazioni in informatica, come per esempio le operazioni bit-a-bit e i circuiti binari, il flusso di controllo nella programmazione, le analisi delle specifiche, le interrogazioni su basi di dati, la ricerca booleana nei motori di ricerca, e molte altre ancora che qui non tratteremo ma che incontrerete nel corso di laurea in Informatica.

Sintassi

Le formule del calcolo proposizionale sono ottenute a partire da un insieme di *simboli proposizionali*, che rappresentano dei fatti o enunciati basilari, componendoli in modo arbitrario con le operazioni di negazione, congiunzione, disgiunzione e implicazione (semplice o doppia), chiamati anche *connettivi logici*.[1]

Definizione 8.1.1 (Sintassi del calcolo proposizionale)**.** *Fissato un insieme $X = \{A, B, C, \ldots\}$ di* SIMBOLI PROPOSIZIONALI*, l'insieme delle* FORMULE PROPOSIZIONALI **Prop** *è il linguaggio generato dalla categoria sintattica $\langle Prop\rangle$ della seguente grammatica:*[2]

Linguaggio proposizionale

$$\begin{array}{rcl} \langle Prop\rangle & \rightsquigarrow & \langle Atom\rangle \mid \neg\langle Atom\rangle \mid \langle Prop\rangle\langle OpB\rangle\langle Prop\rangle \\ \langle Atom\rangle & \rightsquigarrow & \mathsf{T} \mid \mathsf{F} \mid \langle X\rangle \mid (\langle Prop\rangle) \\ \langle OpB\rangle & \rightsquigarrow & \wedge \mid \vee \mid \Rightarrow \mid \Leftarrow \mid \Leftrightarrow \\ \langle X\rangle & \rightsquigarrow & A \mid B \mid C \mid \ldots \end{array}$$

Dalla definizione si evince che l'alfabeto della grammatica è l'insieme

$$A_{Prop} = X \cup \{\Leftrightarrow, \Rightarrow, \Leftarrow, \wedge, \vee, \neg, \mathsf{T}, \mathsf{F}, (,)\}$$

mentre le categorie sintattiche sono $\{\langle Prop\rangle, \langle Atom\rangle, \langle OpB\rangle, \langle X\rangle\}$. Useremo di solito le lettere $P, Q, R, \ldots$ per denotare generiche formule proposizionali, e $A, B, C, \ldots$ per i simboli proposizionali.

La categoria sintattica $\langle Atom\rangle$ genera le formule ATOMICHE, $\langle OpB\rangle$ genera i CONNETTIVI LOGICI, e $\langle X\rangle$ i simboli proposizionali.

Esempio 8.1.2 (formule proposizionali)**.** *Dalla definizione segue che le seguenti stringhe sono formule proposizionali:*

$$\mathsf{T} \qquad A \wedge (B \vee \neg C) \qquad ((A \wedge B \vee \mathsf{F}) \Rightarrow \neg A \vee (C \wedge \mathsf{T})) \qquad (A \Rightarrow B) \wedge \neg(B \wedge \neg C) \Rightarrow (A \Rightarrow C)$$

Invece non sono formule proposizionali per esempio le stringhe $A \wedge \vee B$, $(A \neg C)$ e $A \neg$, perché non sono ottenibili con le produzioni della grammatica della Definizione 8.1.1.

I connettivi logici possono essere letti in vari modi, di cui menzioniamo i più comuni:

- La NEGAZIONE $\neg P$ può essere letta *"non P"*, *"not P"*, *"non è vero che P vale"*, ...
- La CONGIUNZIONE $P \wedge Q$ può essere letta *"P e Q"*, *"P and Q"*, *"P e anche Q"*,...
- La DISGIUNZIONE $P \vee Q$ può essere letta *"P o Q"*, *"P or Q"*, *"P oppure Q"*, ...
- L'IMPLICAZIONE $P \Rightarrow Q$ può essere letta *"se P allora Q"*, *"P implica Q"*, *"P solo se Q"*, *"P è condizione sufficiente per Q"*, ... Nell'implicazione $P \Rightarrow Q$ la formula proposizionale P è chiamata la *premessa*, mentre Q è la *conseguenza* o *conclusione*.
- La CONSEGUENZA $P \Leftarrow Q$ può essere letta *"P è conseguenza di Q"*, *"P se Q"*, *"P if Q"*, *"P è condizione necessaria per Q"*, ...
- La DOPPIA IMPLICAZIONE $P \Leftrightarrow Q$ può essere letta *"P sse Q"*, *"P se e solo se Q"*, *"P iff Q"*, *"P è condizione necessaria e sufficiente per Q"*, ...

Osservazione 8.1.3. *La conseguenza $P \Leftarrow Q$ può essere considerata come un modo alternativo di scrivere l'implicazione $Q \Rightarrow P$. Quindi $Q \Rightarrow P$ può essere letta anche come* "P se Q", "P if Q", "P è condizione necessaria per Q".

[1] L'introduzione dei connettivi logici come operatori su formule (in qualche modo analoghi ai ben noti operatori algebrici su espressioni aritmetiche) risale al 1854, con la pubblicazione dell'opera "The Laws of Thought" di George Boole (1815–1864), importante logico e matematico inglese. I valori *booleani* prendono il nome da lui.

[2] Usando la terminologia della Definizione 7.3.8, $\mathbf{Prop} = \langle\langle\langle Prop\rangle\rangle\rangle$.

Formalizzare proposizioni

Per "formalizzare" intendiamo il procedimento di estrazione, da una proposizione in italiano, di una formula del calcolo proposizionale che ne ha la stessa struttura logica. Di solito questo procedimento non è molto complicato. Seguiamone i passi nel caso della proposizione $\boxed{\textit{"Piove e fa freddo."}}$.

- Come prima cosa occorre introdurre un simbolo proposizionale per ogni proposizione elementare, cioè ogni pezzo della frase che riconosciamo essere una proposizione (ha uno e un solo valore di verità), e che non può essere scomposta in proposizioni più piccole. Nel nostro esempio, fissiamo i simboli proposizionali P per *"piove"* e Fr per *"fa freddo"*. (Non usiamo F per *"fa freddo"* per evitare confusione con F. Osserviamo pure che i simboli proposizionali possono essere stringhe arbitrarie, non solo caratteri.)
- Successivamente si costruisce la formula collegando le occorrenze dei simboli proposizionali con connettivi logici, in modo da rispecchiare fedelmente il significato originale. Nell'esempio, la formula risultante è $\boxed{P \wedge Fr}$, dato che *"e"* rappresenta chiaramente una congiunzione.

La scelta dei connettivi non è sempre ovvia poiché in italiano possono essere resi in molti modi diversi.

Esempio 8.1.4 (da proposizioni in linguaggio naturale a formule proposizionali)**.** *Vediamo come formalizzare altre proposizioni simili.*

1. $\boxed{\text{"Piove ma non fa freddo."}}$ *Usando i simboli proposizionali appena introdotti, una ragionevole formalizzazione è* $\boxed{P \wedge \neg Fr}$. *Infatti anche* "ma" *rappresenta una congiunzione, anche se con un significato leggermente diverso da* "e".

2. $\boxed{\text{"Se piove o fa freddo allora ci si copre."}}$ *Usiamo i simboli P e Fr come sopra, e introduciamo C per* "ci si copre". *Allora la formula risultante è* $\boxed{(P \vee Fr) \Rightarrow C}$. *Infatti* "se ... allora ..." *rappresenta un'implicazione, e* "o" *una disgiunzione.*

3. $\boxed{\text{"Se piove non si esce o si prende l'ombrello."}}$ *Usiamo E per* "si esce" *e O per* "si prende l'ombrello". *Una ragionevole formalizzazione è* $\boxed{P \Rightarrow (\neg E \vee O)}$.

Si noti che un'altra formalizzazione possibile, data l'ambiguità della proposizione in italiano, sarebbe quella che pone in disgiunzione gli enunciati "Se piove non si esce" e "si prende l'ombrello": $(P \Rightarrow \neg E) \vee O$. *Il contesto e il buon senso ci portano a scartare questa possibilità.*

Semantica

La semantica di una formula proposizionale, cioè il suo valore di verità, può essere calcolato per induzione strutturale in base al suo albero di derivazione. In generale tale valore non è determinato univocamente ma dipende da una *interpretazione*, cioè dal valore di verità assegnato ai simboli proposizionali che essa contiene, proprio come il valore di un'espressione algebrica dipende dal valore assegnato alle variabili che contiene.

Definizione 8.1.5 (interpretazione)**.** *Un'*INTERPRETAZIONE $\mathcal{I} : X \to \{\mathtt{f}, \mathtt{t}\}$ *è una funzione che assegna un valore di verità a ogni simbolo proposizionale.*

Di seguito useremo la notazione $\mathcal{I} = \{\ldots, A \mapsto \mathtt{t}, \ldots\}$ (oppure $\mathcal{I} = \{\ldots, A \mapsto \mathtt{f}, \ldots\}$) per indicare un'interpretazione che assegna il booleano vero (oppure falso) al simbolo A.

Fissato il valore dei simboli proposizionali, la semantica di una formula proposizionale è ottenuta, per induzione strutturale, valutando i connettivi logici che compaiono in essa in base al loro significato. Abbiamo anticipato il significato dei connettivi $\wedge, \vee$ e $\neg$ negli Esempi 2.5.11 e 2.5.12: lo ricordiamo nella seguente definizione insieme a quello degli altri connettivi, utilizzando delle *tavole di verità*.

Definizione 8.1.6 (connettivi logici come funzioni)**.** *Come visto nell'Esempio 2.5.11, la negazione è una funzione $\neg : Bool \to Bool$, descritta dalla tavola di verità che segue a sinistra. La prima*

colonna elenca i possibili valori assunti da $x \in Bool$, e la seconda colonna mostra i corrispondenti valori di $\neg x$.

x	$\neg x$
f	t
t	f

x	y	$x \wedge y$	$x \vee y$	$x \Rightarrow y$	$x \Leftarrow y$	$x \Leftrightarrow y$
f	f	f	f	t	t	t
f	t	f	t	t	f	f
t	f	f	t	f	t	f
t	t	t	t	t	t	t

Analogamente, i connettivi logici binari $\wedge, \vee, \Rightarrow, \Leftarrow, \Leftrightarrow$ sono delle funzioni da $Bool \times Bool$ a $Bool$ definite dalla seconda tavola di verità. Nella parte a sinistra della doppia linea verticale sono elencate le quattro coppie di valori booleani che possono assumere x e y. Per ognuna di queste coppie, nella parte destra della tavola sono riportati sulla stessa riga i valori delle formule $x \wedge y$, $x \vee y$, $x \Rightarrow y$, $x \Leftarrow y$ e $x \Leftrightarrow y$.

Abbiamo già commentato nella Sezione 1.3 e nell'Esempio 2.5.12 i connettivi $\wedge$ e $\vee$, per cui ci limitiamo a osservare che la tavola di verità conferma il significato che gli avevamo attribuito:

- *($\wedge$) La congiunzione $x \wedge y$ è vera se sia x che y sono vere, altrimenti è falsa. Infatti nella colonna relativa a $x \wedge y$ c'è una sola riga con* t*, corrispondente ai valori* t *e* t *per x e y.*
- *($\vee$) La disgiunzione $x \vee y$ è vera se x è vera oppure y è vera oppure entrambe sono vere, altrimenti è falsa. Infatti nella colonna relativa a $x \vee y$ c'è una sola riga con* f*, corrispondente ai valori* f *e* f *per x e y.*

Per quanto riguarda l'implicazione $x \Rightarrow y$, analizzando la terzultima colonna della tavola di verità vediamo che essa è falsa se la premessa x è vera e la conseguenza y è falsa, mentre è vera in tutti gli altri casi. Intuitivamente, possiamo pensare a $x \Rightarrow y$ come a una regola ("se x è vera allora y deve essere vera"), e il suo valore booleano ci dice se la regola è rispettata oppure no. Allora se x è falsa la regola è automaticamente rispettata perché non si applica affatto (le prime due righe della tavola, per $x =$ f*). Se x è vera e anche y è vera, la regola è rispettata (l'ultima riga). L'unico caso in cui la regola non è rispettata, e quindi l'implicazione vale* f*, è quando x è vera ma y è falsa (la penultima riga della tavola).*

Per la conseguenza $x \Leftarrow y$ si applicano considerazioni simili a quelle per l'implicazione, scambiando i ruoli di x e y per l'Osservazione 8.1.3.

Infine dall'ultima colonna della tavola si vede che la doppia implicazione $x \Leftrightarrow y$ è vera se e solo se x e y hanno lo stesso valore.

Esempio 8.1.7 (formalizzare proposizioni con implicazione)**.** *Ora che abbiamo descritto in modo preciso il significato dell'implicazione, vediamo come lo possiamo usare per formalizzare correttamente una proposizione. Consideriamo le proposizioni elementari* "io vado al cinema"*, rappresentata dalla variabile proposizionale V, e* "tu resti a casa"*, rappresentata da R. Come possiamo formalizzare la proposizione* "Affinché io vada al cinema è necessario che tu resti a casa"*?*

Riconosciamo che la frase esprime una regola: se V è vera, necessariamente R deve essere vera, quindi la regola è violata solo se V è vera e R è falsa. Questo corrisponde precisamente alla regola associata all'implicazione $V \Rightarrow R$, come descritta sopra, che quindi è una adeguata formalizzazione della proposizione.

Consideriamo ora "Io vado al cinema se tu resti a casa"*. In questo caso la regola sancita dalla proposizione è violata quando* "tu resti a casa" *ma* "io non vado al cinema"*, e quindi una adeguata formalizzazione è $R \Rightarrow V$.*

Definizione 8.1.8 (Semantica del calcolo proposizionale)**.** *Data una interpretazione $\mathcal{I} : X \to \{\mathsf{f}, \mathsf{t}\}$, il* VALORE RISPETTO AD $\mathcal{I}$ *delle formule proposizionali è dato dalla funzione*

$$[\![\_]\!]_{\mathcal{I}} : \mathbf{Prop} \to \{\mathsf{f}, \mathsf{t}\}$$

definita sulle formule proposizionali per induzione strutturale come segue:

1. $[\![\mathsf{T}]\!]_{\mathcal{I}} = \mathsf{t}$ *e* $[\![\mathsf{F}]\!]_{\mathcal{I}} = \mathsf{f}$*;*
2. $[\![A]\!]_{\mathcal{I}} = \mathcal{I}(A)$ *per ogni $A \in X$;*

3. $[\![(P)]\!]_{\mathcal{I}} = ([\![P]\!]_{\mathcal{I}})$ *per ogni* $P \in \mathbf{Prop}$*;*
4. $[\![\neg Q]\!]_{\mathcal{I}} = \neg[\![Q]\!]_{\mathcal{I}}$ *per ogni formula atomica* Q*;*
5. $[\![P \ op \ Q]\!]_{\mathcal{I}} = [\![P]\!]_{\mathcal{I}} \ op \ [\![Q]\!]_{\mathcal{I}}$ *per ogni connettivo* $op \in \{\wedge, \vee, \Rightarrow, \Leftarrow, \Leftrightarrow\}$ *e per ogni* $P, Q \in \mathbf{Prop}$.

Si osservi come le clausole della Definizione 8.1.8 corrispondano alle produzioni della grammatica di Definizione 8.1.1, confermando che $[\![\_]\!]_{\mathcal{I}}$ è definita per induzione strutturale: le clausole 1–3 corrispondono alla produzione $\langle Prop\rangle \rightsquigarrow \langle Atom\rangle$ (più precisamente alle produzioni di $\langle Atom\rangle$), la clausola 4 a $\langle Prop\rangle \rightsquigarrow \neg\langle Atom\rangle$ e la clausola 5 a $\langle Prop\rangle \rightsquigarrow \langle Prop\rangle\langle OpB\rangle\langle Prop\rangle$.

Esempio 8.1.9 (valutazione di una formula)**.** *Consideriamo la formula* $P = (A \wedge B) \vee \neg C$ *e l'interpretazione* $\mathcal{I} = \{A \mapsto \mathtt{t}, B \mapsto \mathtt{f}, C \mapsto \mathtt{f}, \ldots\}$,[3] *e calcoliamo il valore* $[\![P]\!]_{\mathcal{I}}$ *secondo la Definizione 8.1.8.*

$$
\begin{array}{rcll}
[\![(A \wedge B) \vee \neg C]\!]_{\mathcal{I}} & = & [\![(A \wedge B)]\!]_{\mathcal{I}} \vee [\![\neg C]\!]_{\mathcal{I}} & \textit{(per la clausola 5 con } op = \vee\textit{)} \\
& = & ([\![A \wedge B]\!]_{\mathcal{I}}) \vee \neg[\![C]\!]_{\mathcal{I}} & \textit{(clausola 3 e clausola 4)} \\
& = & ([\![A]\!]_{\mathcal{I}} \wedge [\![B]\!]_{\mathcal{I}}) \vee \neg[\![C]\!]_{\mathcal{I}} & \textit{(clausola 5 con } op = \wedge\textit{)} \\
& = & (\mathtt{t} \wedge \mathtt{f}) \vee \neg\mathtt{f} & \textit{(clausola 2, tre volte)} \\
& = & \mathtt{f} \vee \mathtt{t} & \textit{(definizione di } \wedge \textit{ e } \neg \textit{ da Definizione 8.1.6)} \\
& = & \mathtt{t} & \textit{(definizione di } \vee\textit{)}
\end{array}
$$

Concludiamo questa sezione introducendo la definizione di modello di un insieme di formule proposizionali e il concetto di conseguenza logica.

Definizione 8.1.10 (modello, equivalenza, conseguenza logica)**.** *Data una formula proposizionale* P *e una interpretazione* $\mathcal{I}$*, diciamo che* $\mathcal{I}$ *è un* MODELLO *di* P *se* P *è vera in* $\mathcal{I}$*, cioè se* $[\![P]\!]_{\mathcal{I}} = \mathtt{t}$*. Per questa importante nozione introduciamo una apposita notazione:*

$$\mathcal{I} \models P \qquad\qquad (\mathcal{I} \textit{ è modello di } P)$$

Se invece $[\![P]\!]_{\mathcal{I}} = \mathtt{f}$ *scriviamo* $\mathcal{I} \not\models P$*. La notazione si estende nel modo ovvio a un insieme di formule* Γ *(letto "Gamma"): scriviamo* $\mathcal{I} \models \Gamma$ *se* $\mathcal{I} \models P$ *per ogni* $P \in \Gamma$*, mentre scriviamo* $\mathcal{I} \not\models \Gamma$ *se c'è almeno una formula* $P \in \Gamma$ *tale che* $\mathcal{I} \not\models P$*.*

Due formule proposizionali P *e* Q *sono* LOGICAMENTE EQUIVALENTI *se hanno gli stessi modelli, cioè assumono lo stesso valore di verità per qualunque interpretazione. In questo caso scriviamo:*

$$P \equiv Q \qquad\qquad (P \textit{ e } Q \textit{ sono logicamente equivalenti})$$

Data una formula proposizionale P *e un insieme di formule* Γ*, diciamo che* P è una CONSEGUENZA LOGICA di Γ *se* P *è vera in ogni interpretazione che rende vere tutte le formule di* Γ*, oppure, equivalentemente, se ogni modello di* Γ *è anche un modello di* P*. In questo caso scriviamo:*

$$\Gamma \models P \qquad\qquad (P \textit{ è conseguenza logica di } \Gamma)$$

È opportuno sottolineare che dalla definizione di modello segue che per qualunque interpretazione $\mathcal{I}$ vale $\mathcal{I} \models \varnothing$, dove $\varnothing$ è l'insieme vuoto di formule. Infatti se Γ è un insieme di formule, intuitivamente per verificare che valga $\mathcal{I} \models \Gamma$ dobbiamo prendere una ad una le formule $P \in \Gamma$ e verificare che $\mathcal{I} \models P$. Ma se $\Gamma = \varnothing$ non c'è niente da verificare, e quindi $\mathcal{I} \models \varnothing$ vale automaticamente (o, come si dice in questo contesto, *vacuamente*).

Dalla Definizione 8.1.10 segue che possiamo definire l'equivalenza logica in termini della conseguenza logica, come enunciato nella seguente proposizione di cui lasciamo la semplice dimostrazione al lettore.

Proposizione 8.1.11 (equivalenza e conseguenza logica)**.** *Date due formule proposizionali* P *e* Q *vale che*

$$P \equiv Q \qquad \textit{se e solo se} \qquad \{P\} \models Q \quad e \quad \{Q\} \models P$$

[3] In realtà l'interpretazione è una funzione $\mathcal{I}\colon X \to \{\mathtt{f}, \mathtt{t}\}$ dove X è un insieme potenzialmente infinito di simboli, ma ci interessa il suo valore solo per i simboli proposizionali contenuti nella formula considerata.

8.2 Tavole di verità e tautologie

Nell'Esempio 8.1.9 abbiamo visto come calcolare il valore di una formula proposizionale P in un'interpretazione $\mathcal{I}$, usando le clausole della Definizione 8.1.8 e il significato dei connettivi logici. Le *tavole di verità* permettono di fare questa valutazione in modo più semplice ma equivalente.

Consideriamo di nuovo la formula $((A \wedge B) \vee \neg C)$ e l'interpretazione $\mathcal{I} = \{A \mapsto \mathtt{t}, B \mapsto \mathtt{f}, C \mapsto \mathtt{f}, \ldots\}$. Una volta assegnati i valori di verità ai simboli A, B e C, possiamo ricavare i valori di verità associati alle sottoformule $(A \wedge B)$ e $\neg C$ e quindi all'intera formula $((A \wedge B) \vee \neg C)$. Possiamo rappresentare questo ragionamento in modo compatto con la seguente tavola:

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
t	f	f	t	f	f	**t**	t	f
			(1)	(2)	(1)	**(3)**	(2)	(1)

Nella prima riga abbiamo a destra della doppia riga verticale la formula e a sinistra i simboli proposizionali che vi compaiono. Nella seconda riga, sotto i simboli proposizionali abbiamo il valore fissato dall'interpretazione, e sotto i connettivi logici il risultato determinato dal loro significato. Il valore di verità della formula è quello riportato sotto il *connettivo principale*, cioè l'unico che non compare nella formula come argomento di un altro connettivo: in questo caso è $\vee$. I numeri in parentesi sotto le colonne indicano l'ordine con cui viene compilata la tavola: si parte dalle colonne dei simboli proposizionali e si applicano i connettivi nell'ordine stabilito dalle parentesi (o dai livelli di precedenza dei connettivi, come vedremo). Nel nostro caso l'ordine è il seguente:

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
t	f	f	**t**		**f**			**f**

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
t	f	f	t	**f**	f		**t**	f

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
t	f	f	t	f	f	**t**	t	f

Quante sono le possibili diverse interpretazioni per una data formula proposizionale P? Considerando solo il valore delle interpretazioni per i simboli proposizionali contenuti in P, è facile convincersi che sono 2^n, dove n è il numero di simboli proposizionali distinti che compaiono in P. Per esempio, per la formula $((A \wedge B) \vee \neg C)$ abbiamo due possibili valori per A, due per B e due per C, per un totale di $2^3 = 8$ distinte interpretazioni. Una formula con 10 simboli proposizionali diversi avrebbe $2^{10} = 1024$ interpretazioni.

Data una formula proposizionale $P \in \mathbf{Prop}$, una *tavola di verità per* P elenca, una per riga, tutte le possibili interpretazioni $\mathcal{I}$ per P e il corrispondente valore di verità $[\![P]\!]_{\mathcal{I}}$ della formula. Nella Figura 8.1 mostriamo come si può costruire la tavola di verità per la formula $((A \wedge B) \vee \neg C)$, partendo dall'elenco di tutte le possibili interpretazioni e poi annotando progressivamente i connettivi con i valori calcolati.

Osservazione 8.2.1. *La grammatica della Definizione 8.1.1 è ambigua, perché ci sono delle stringhe in* **Prop** *che sono testimoniate da alberi di derivazione diversi. Questo significa che valutando una formula in una data interpretazione come appena descritto, ci possiamo trovare nella situazione di dover scegliere quale connettivo valutare prima, ottenendo potenzialmente risultati diversi.*

Nel seguito assumiamo che tra i connettivi logici sussistano i seguenti livelli di precedenza (in ordine crescente) e useremo le parentesi quando necessario per indicare esplicitamente l'ordine di valutazione dei connettivi (come si fa usualmente con le espressioni algebriche):

connettivo	*livello di precedenza*
$\Leftrightarrow$	*0*
$\Rightarrow, \Leftarrow$	*1*
$\wedge, \vee$	*2*
$\neg$	*3*

0. Possibili assegnamenti di verità ai simboli proposizionali:

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
f	**f**	**f**						
f	**f**	**t**						
f	**t**	**f**						
f	**t**	**t**						
t	**f**	**f**						
t	**f**	**t**						
t	**t**	**f**						
t	**t**	**t**						

1. Valutazione dei simboli proposizionali nella formula:

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
f	f	f	**f**		**f**			**f**
f	f	t	**f**		**f**			**t**
f	t	f	**f**		**t**			**f**
f	t	t	**f**		**t**			**t**
t	f	f	**t**		**f**			**f**
t	f	t	**t**		**f**			**t**
t	t	f	**t**		**t**			**f**
t	t	t	**t**		**t**			**t**
			(1)		(1)			(1)

2. Valutazione delle espressioni più annidate $((A \wedge B)$ e $\neg C)$:

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
f	f	f	f	**f**	f		**t**	f
f	f	t	f	**f**	f		**f**	t
f	t	f	f	**f**	t		**t**	f
f	t	t	f	**f**	t		**f**	t
t	f	f	t	**f**	f		**t**	f
t	f	t	t	**f**	f		**f**	t
t	t	f	t	**t**	t		**t**	f
t	t	t	t	**t**	t		**f**	t
			(1)	(2)	(1)		(2)	(1)

3. Valutazione delle espressioni più annidate e non ancora valutate:

A	B	C	$((A$	$\wedge$	$B)$	$\vee$	$\neg$	$C)$
f	f	f	f	f	f	**t**	t	f
f	f	t	f	f	f	**f**	f	t
f	t	f	f	f	t	**t**	t	f
f	t	t	f	f	t	**f**	f	t
t	f	f	t	f	f	**t**	t	f
t	f	t	t	f	f	**f**	f	t
t	t	f	t	t	t	**t**	t	f
t	t	t	t	t	t	**t**	f	t
			(1)	(2)	(1)	(3)	(2)	(1)

Figura 8.1: Costruzione della tavola di verità della formula $((A \wedge B) \vee \neg C)$.

Per esempio, se dobbiamo valutare la formula $A \wedge B \Rightarrow C$, poiché $\wedge$ ha precedenza maggiore di $\Rightarrow$, la valuteremo come se fosse scritta $(A \wedge B) \Rightarrow C$. Nel caso di formule in cui le regole di precedenza non ci aiutano, perché abbiamo due connettivi dello stesso livello, avremo due casi:

- *considereremo sintatticamente errate formule come $A \wedge B \vee C$ oppure $A \Rightarrow B \Rightarrow C$, che possono assumere valori diversi per la stessa interpretazione a seconda dell'ordine di valutazione dei connettivi;*[4]
- *invece accetteremo senza problemi formule come $A \wedge B \wedge C$ oppure $A \vee B \vee C$ che pur essendo sintatticamente ambigue hanno un unico possibile valore per ogni interpretazione.*

In caso di dubbio, scrivendo formule complesse si consiglia di abbondare con le parentesi in modo da evitare qualunque rischio di ambiguità.

Esempio 8.2.2 (tavola di verità)**.** *Come ulteriore esempio di tavola di verità vediamo quella di una formula leggermente più complessa, dove compaiono anche le costanti* T *e* F*:* $((A \wedge (B \vee \mathsf{F})) \Rightarrow (\neg A \vee (C \wedge \mathsf{T})))$.

A	B	C	$((A$	$\wedge$	$(B$	$\vee$	$\mathsf{F}))$	$\Rightarrow$	$(\neg$	A	$\vee$	$(C$	$\wedge$	$\mathsf{T})))$
f	f	f	f	f	f	f	f	**t**	t	f	t	f	f	t
f	f	t	f	f	f	f	f	**t**	t	f	t	t	t	t
f	t	f	f	f	t	t	f	**t**	t	f	t	f	f	t
f	t	t	f	f	t	t	f	**t**	t	f	t	t	t	t
t	f	f	t	f	f	f	f	**t**	f	t	f	f	f	t
t	f	t	t	f	f	f	f	**t**	f	t	t	t	t	t
t	t	f	t	t	t	t	f	**f**	f	t	f	f	f	t
t	t	t	t	t	t	t	f	**t**	f	t	t	t	t	t
			(1)	(3)	(1)	(2)	(1)	(4)	(2)	(1)	(3)	(1)	(2)	(1)

Esercizio 8.2.3. *Costruire le tavole di verità per $(A \wedge B) \vee C$ e $A \wedge (B \vee C)$. Per quali interpretazioni le due formule differiscono?*

Esercizio 8.2.4. *Costruire le tavole di verità per $(A \Rightarrow B) \Rightarrow C$ e $A \Rightarrow (B \Rightarrow C)$. Per quali interpretazioni le due formule differiscono?*

Le tautologie

Abbiamo visto che il valore di verità di una formula proposizionale dipende, in generale, dall'interpretazione dei suoi simboli proposizionali. Tuttavia nel calcolo proposizionale ci interessano in particolar modo le *tautologie*, cioè le formule che risultano vere per qualunque interpretazione: esse infatti, come vedremo in seguito, ci permettono di rappresentare schemi di inferenza o di ragionamento corretti.

Definizione 8.2.5 (tautologie, contraddizioni, formule soddisfacibili)**.** *Una* TAUTOLOGIA *è una formula proposizionale che è sempre vera, per qualunque interpretazione. Sfruttando la notazione introdotta nella Definizione 8.1.10, se P è una tautologia scriviamo semplicemente*

$$\models P \qquad\qquad (P \text{ è una tautologia})$$

Una CONTRADDIZIONE *(o* formula insoddisfacibile*) è una formula proposizionale che è sempre falsa, per qualunque interpretazione. Una formula proposizionale è* SODDISFACIBILE *se esiste almeno una interpretazione per la quale è vera.*

Si osservi che la notazione "$\models P$" introdotta per indicare che P è una tautologia non è altro che un'abbreviazione sintattica di "$\varnothing \models P$", che per la Definizione 8.1.10 significa che P è conseguenza logica dell'insieme vuoto di formule. Infatti $\varnothing \models P$ se e solo se (per definizione di conseguenza logica) P è vera in tutte le interpretazioni che rendono vere tutte le formule in $\varnothing$, se e solo se P è vera in tutte le interpretazioni, se e solo se P è una tautologia.

[4] Lasciamo come esercizio per il lettore il compito di trovare tali interpretazioni.

Dalla definizione segue che nella tavola di verità di una formula proposizionale la colonna sotto il connettivo principale conterrà tutti `t` se la formula è una tautologia, tutti `f` se è una contraddizione, e almeno un `t` se è soddisfacibile.

Esempio 8.2.6 (tautologie, contraddizioni e formule soddisfacibili)**.** *La seguente tavola di verità mostra che la formula $A \wedge B \Rightarrow B$ è una tautologia (e quindi anche soddisfacibile), la formula $A \wedge (B \wedge \neg A)$ è una contraddizione, mentre la formula $A \Rightarrow B$ è soddisfacibile ma non è una tautologia.*

A	B	$(A$	$\wedge$	$B)$	$\Rightarrow$	B	A	$\wedge$	$(B$	$\wedge$	$\neg$	$A)$	A	$\Rightarrow$	B
f	f	f	f	f	**t**	f	f	**f**	f	f	t	f	f	**t**	f
f	t	f	f	t	**t**	t	f	**f**	t	t	t	f	f	**t**	t
t	f	t	f	f	**t**	f	t	**f**	f	f	f	t	t	**f**	f
t	t	t	t	t	**t**	t	t	**f**	t	f	f	t	t	**t**	t
		(1)	(2)	(1)	**(3)**	(1)	(1)	**(4)**	(1)	(3)	(2)	(1)	(1)	**(2)**	(1)

Un problema fondamentale del calcolo proposizionale è quello di dimostrare che una data formula è una tautologia. Molti altri problemi interessanti si possono ridurre a questo. Per esempio, se dobbiamo dimostrare che la formula A è una contraddizione, ci basta dimostrare che $\neg A$ è una tautologia; se dobbiamo dimostrare che, assumendo che le formule A e B siano vere, allora anche C è vera (cioè che C è una conseguenza logica di $\{A, B\}$), ci basta dimostrare che $A \wedge B \Rightarrow C$ è una tautologia.

Per quanto visto sopra, per vedere se una formula è una tautologia sarebbe sufficiente costruire la sua tavola di verità e controllare che il valore della formula sia `t` su tutte le righe (quindi per ogni interpretazione). Questo procedimento, anche se può essere completamente automatizzato, può richiedere la costruzione di una tavola molto grande (come visto, il numero di righe è 2^n, dove n è il numero di simboli proposizionali della formula) ed è molto soggetto ad errori se fatto a mano. Pertanto nel seguito useremo solo raramente questa tecnica, e vedremo invece nella Sezione 8.3 come dimostrare che una formula è una tautologia usando le dimostrazioni per sostituzione introdotte nella Sezione 1.4.

Se invece dobbiamo mostrare che una formula è soddisfacibile, non c'è bisogno di costruire tutta la tavola di verità, ma è sufficiente trovare una singola interpretazione che renda la formula vera. Spesso questo può essere fatto in modo abbastanza efficiente ragionando sulla struttura della formula. Analogamente, per mostrare che una formula *non è* una tautologia è sufficiente trovare una interpretazione che renda la formula falsa.

Esempio 8.2.7 (formula non tautologica)**.** *Mostriamo, senza costruire la tavola di verità, che la formula $(((A \Rightarrow \neg B) \wedge \neg A) \Rightarrow B)$ non è una tautologia, ovvero:*

$$\not\models \quad ((A \Rightarrow \neg B) \wedge \neg A) \Rightarrow B$$

Sfruttiamo la struttura della formula per costruire un'interpretazione che renda falsa la formula. Per cominciare, osserviamo che il connettivo principale della formula è un'implicazione. L'unico caso in cui un'implicazione è falsa è quando la premessa è vera e la conseguenza è falsa (si veda la Definizione 8.1.6). Quindi l'interpretazione cercata deve associare `f` a B, la conseguenza, e `t` alla sottoformula $((A \Rightarrow \neg B) \wedge \neg A)$, la premessa. A sua volta quest'ultima formula ha come connettivo principale una congiunzione, che è vera solo se entrambi gli argomenti sono veri. Quindi in particolare $\neg A$ deve valere `t`, cioè l'interpretazione deve associare `f` a A. Abbiamo quindi individuato l'interpretazione $\{A \mapsto \mathtt{f}, B \mapsto \mathtt{f}\}$, che per il ragionamento fatto è l'unica che potrebbe rendere la formula falsa. Valutando l'intera formula in questa interpretazione otteniamo effettivamente `f` (cioè falso), come si vede dalla seguente tavola:

A	B	$((A$	$\Rightarrow$	$\neg$	$B)$	$\wedge$	$\neg$	$A)$	$\Rightarrow$	B
f	f	f	t	t	f	t	t	f	**f**	f
		(1)	(3)	(2)	(1)	(4)	(2)	(1)	**(5)**	(1)

Formalizzazione di inferenze e tautologie

Si può sfruttare la formalizzazione di proposizioni per mostrare la correttezza di inferenze o di semplici ragionamenti espressi in linguaggio naturale. Un'inferenza è corretta se la tesi è una conseguenza logica delle ipotesi, e questo si può accertare dimostrando che l'implicazione *"Ipotesi* $\Rightarrow$ *Tesi"* è una tautologia. Vediamo due esempi: un'inferenza corretta e una errata.

Esempio 8.2.8 (dimostrazione di correttezza di un'inferenza)**.** *Si dimostri la correttezza della seguente inferenza:*

"Studio oggi oppure domani, ma domani non studio, quindi studio oggi"

Per formalizzare la frase, introduciamo un simbolo proposizionale per ogni proposizione elementare, quindi SO per "studio oggi" *e SD per* "studio domani"*.*

Ora costruiamo una formula proposizionale che usa questi simboli e che esprime il significato inteso della frase:

$$(SO \vee SD) \wedge \neg SD \;\Rightarrow\; SO$$

Quindi abbiamo rappresentato "oppure" con la disgiunzione $\vee$*, "non" con la negazione* $\neg$*, "quindi" con l'implicazione* $\Rightarrow$ *e "ma" con la congiunzione "*$\wedge$*". Si noti pure che abbiamo associato in modo abbastanza flessibile le proposizioni della frase alle sottoformule: "domani" è rappresentato da SD poiché sottintende "studio domani", e "domani non studio" da* $\neg SD$*. La formula proposizionale ottenuta è una tautologia, come il lettore può facilmente verificare, confermando che l'inferenza è corretta.*

Naturalmente non tutte le inferenze sono corrette: vediamo come la formalizzazione logica ci può aiutare a rivelare l'inesattezza di un ragionamento.

Esempio 8.2.9 (dimostrazione di non correttezza di un'inferenza)**.** *È corretta la seguente inferenza?*

"Se studio oggi allora domani non studio, ma oggi non studio, quindi domani studierò"

Come nell'esempio precedente possiamo usare i simboli proposizionali SO per "studio oggi" e SD per "studio domani". Una formula proposizionale che rappresenta fedelmente il significato della frase è:

$$(SO \Rightarrow \neg SD) \wedge \neg SO \Rightarrow SD$$

Ma come possiamo dedurre dall'Esempio 8.2.7 dove abbiamo analizzato una formula con la stessa struttura (anche se con simboli proposizionali diversi), l'interpretazione $\{SO \mapsto \mathtt{f}, SD \mapsto \mathtt{f}\}$ *rende questa formula falsa, pertanto non è una tautologia. Quindi l'inferenza non è corretta, perché "studio domani" non è una conseguenza logica di "se studio oggi allora domani non studio" e di "oggi non studio".*

8.3 Dimostrazioni, nel calcolo proposizionale e oltre

Abbiamo visto nella Definizione 8.1.10 che una formula P è *conseguenza logica* di un insieme di formule Γ se P è vera in tutti i modelli di Γ. Anche se la abbiamo introdotta per il calcolo proposizionale, questa definizione è sostanzialmente indipendente dalla logica, e in particolare vale anche per la logica dei predicati che vedremo successivamente.

Il concetto di conseguenza logica di per sé non suggerisce alcun metodo pratico per mostrare che una formula P è o non è conseguenza logica di un insieme Γ di formule, o, detto altrimenti, per mostrare che è o meno "legittimo" concludere la conseguenza P dalle premesse Γ. Per mostrare che una formula NON È conseguenza logica di un insieme di premesse, un metodo ragionevole è quello di mostrare che esiste un modello delle premesse in cui la conclusione è falsa. Ciò è quanto abbiamo fatto nell'Esempio 8.2.7 (in quel caso l'insieme delle premesse era vuoto, quindi qualunque interpretazione ne era un modello).

Più problematico è invece stabilire che una formula P è conseguenza logica di un insieme di formule Γ date. Usare direttamente la definizione, cioè analizzare ogni possibile modello di Γ per assicurarsi che in esso P sia vera, sarebbe molto costoso per il calcolo proposizionale e sarebbe

semplicemente impossibile per la logica dei predicati e per molte altre logiche, perché i modelli di una formula o di un insieme di formule sono, in generale, infiniti.

È nel concetto di *dimostrazione* che sta la risposta a questo quesito: la dimostrazione (sintattica) di un asserto P a partire da una collezione di asserti dati Γ, le premesse, mostra che P è conseguenza logica di Γ. A patto di disporre di un insieme di *regole di inferenza* sufficientemente "potenti", è possibile ricondurre il concetto semantico di conseguenza logica al concetto puramente sintattico di dimostrazione. Infatti una dimostrazione è una sequenza di passi di pura manipolazione simbolica degli asserti in gioco, ciascuno corrispondente alla applicazione di una delle regole di inferenza. Intuitivamente, i vari passi di dimostrazione consentono di trarre una serie di conclusioni (conseguenze) intermedie fino ad arrivare alla conclusione desiderata.

Nelle prossime sezioni prima mostriamo come si possono impostare delle dimostrazioni per sostituzione nel calcolo proposizionale, dopo aver presentato un insieme opportuno di leggi. Poi introduciamo il concetto astratto di *sistema di dimostrazioni* e ne vediamo un'istanza concreta per il calcolo proposizionale, che permette di collocare le dimostrazioni per sostituzione in un contesto più formale. Infine discuteremo di come nel calcolo proposizionale è possibile ragionare sulla correttezza di semplici tecniche di dimostrazione.

Dimostrazione di tautologie nel Calcolo Proposizionale

Una *dimostrazione per sostituzione* nel calcolo proposizionale ha lo scopo di mostrare che una formula del tipo $P \Leftrightarrow Q$ è una tautologia, e si sviluppa come una sequenza di passi $P \Leftrightarrow R_1 \Leftrightarrow \ldots \Leftrightarrow R_n \Leftrightarrow Q$ ognuno dei quali consiste nell'applicare una legge o una tautologia già dimostrata precedentemente.

Questo concetto di *"applicare una legge per giustificare un passo"* merita di essere reso più formale. Per prima cosa introduciamo una notazione.

Notazione 8.3.1 (rimpiazzamento)**.** *Siano P, Q e R formule proposizionali. Allora $P\left[{}^{Q}/_{R}\right]$ è definito essere la formula ottenuta da P rimpiazzando una specifica occorrenza della sottoformula R con la formula Q. Per esempio, abbiamo*

$$(A \wedge B \Rightarrow (C \Rightarrow D))\left[{}^{\neg C \vee D}/_{C \Rightarrow D}\right] \quad := \quad (A \wedge B \Rightarrow (\neg C \vee D))$$

dove $:=$ indica che il membro sinistro è definito essere il membro destro. Se la formula R occorre più volte come sottoformula in P assumeremo che sia indicata esplicitamente (per esempio sottolineandola) l'occorrenza da rimpiazzare. Inoltre se necessario aggiungeremo delle parentesi per evitare che la formula risultante sia ambigua. Per esempio,

$$\neg(A \Rightarrow B) \vee (\underline{B} \wedge C)\left[{}^{B \vee D}/_{B}\right] \quad := \quad \neg(A \Rightarrow B) \vee ((B \vee D) \wedge C)$$

La seguente regola di inferenza, il *Principio di sostituzione*, stabilisce che se $Q \Leftrightarrow R$ è una tautologia, allora anche $P \Leftrightarrow P\left[{}^{R}/_{Q}\right]$ è una tautologia.

Principio di sostituzione

$$\frac{Q \Leftrightarrow R}{P \Leftrightarrow P[{}^{R}/_{Q}]}\ [\mathbf{PDS}]$$

Ogni passo di una dimostrazione per sostituzione sarà di fatto una doppia implicazione giustificata da una legge grazie al principio di sostituzione. Per esempio, la commutatività della disgiunzione è stabilita dalla legge "*commutatività:* $P \vee Q \Leftrightarrow Q \vee P$". Possiamo applicarla alla formula $((A \wedge B) \vee C \Rightarrow D)$ ottenendo il seguente passo di dimostrazione:

$$(\underline{(A \wedge B) \vee C} \Rightarrow D) \quad \Leftrightarrow \quad (C \vee (A \wedge B) \Rightarrow D) \qquad \text{(commutatività)}$$

In questo caso abbiamo sottolineato la sottoformula cui abbiamo applicato la legge, e come usuale non abbiamo menzionato esplicitamente l'uso del principio di sostituzione.

Solo per questo esempio, facciamo vedere esplicitamente che il passo di dimostrazione è *una istanza* della regola di inferenza [**PDS**], nel senso che si ottiene da essa applicando un semplice rimpiazzamento. Per prima cosa osserviamo che $P \vee Q \Leftrightarrow Q \vee P$, come ogni legge, rappresenta

un'infinità di tautologie, tutte quelle ottenibili sostituendo P e Q con arbitratrie formule.[5] Quindi in particolare anche $(A \wedge B) \vee C \;\Leftrightarrow\; C \vee (A \wedge B)$ è una tautologia. A questo punto, sostituendo in [**PDS**] la P con $((A \wedge B) \vee C \Rightarrow D)$, la Q con $(A \wedge B) \vee C$ e la R con $C \vee (A \wedge B)$, otteniamo

$$\frac{(A \wedge B) \vee C \quad \Leftrightarrow \quad C \vee (A \wedge B)}{((A \wedge B) \vee C \Rightarrow D) \quad \Leftrightarrow \quad ((A \wedge B) \vee C \Rightarrow D)[{}^{C \vee (A \wedge B)}/_{(A \wedge B) \vee C}]} \; [\mathbf{PDS}]$$

dove la doppia implicazione sotto la riga, effettuato il rimpiazzamento, è esattamente il passo di dimostrazione di sopra.

Naturalmente abbiamo bisogno di un insieme di tautologie da cui partire, verificate indipendentemente, per esempio usando le tavole di verità. Chiameremo *assiomi* queste tautologie, o anche *leggi* come avviene in altri contesti. Come il lettore noterà immediatamente, alcune di queste leggi sono analoghe a quelle per l'uguaglianza di insiemi mostrate nella Sezione 1.4, facendo corrispondere F all'insieme vuoto $\varnothing$, T all'universo $\mathcal{U}$, la negazione al complemento, la congiunzione all'intersezione, e la disgiunzione all'unione.

Proposizione 8.3.2 (alcune leggi del calcolo proposizionale)**.** *Per tutte le formule proposizionali P, Q e R le formule delle Tabelle 8.1, 8.2 e 8.3 sono tautologie.*[6]

unità	$P \vee \mathsf{F} \;\Leftrightarrow\; P$	$P \wedge \mathsf{T} \;\Leftrightarrow\; P$
assorbimento	$P \vee \mathsf{T} \;\Leftrightarrow\; \mathsf{T}$	$P \wedge \mathsf{F} \;\Leftrightarrow\; \mathsf{F}$
idempotenza	$P \vee P \;\Leftrightarrow\; P$	$P \wedge P \;\Leftrightarrow\; P$
commutatività	$P \vee Q \;\Leftrightarrow\; Q \vee P$	$P \wedge Q \;\Leftrightarrow\; Q \wedge P$
associatività	$P \vee (Q \vee R) \;\Leftrightarrow\; (P \vee Q) \vee R$	$P \wedge (Q \wedge R) \;\Leftrightarrow\; (P \wedge Q) \wedge R$
distributività	$P \vee (Q \wedge R) \Leftrightarrow (P \vee Q) \wedge (P \vee R)$	$P \wedge (Q \vee R) \Leftrightarrow (P \wedge Q) \vee (P \wedge R)$

Tabella 8.1: Leggi per disgiunzione e congiunzione

$\mathsf{T} : \mathsf{F}$	$\neg\mathsf{T} \;\Leftrightarrow\; \mathsf{F}$	
doppia negazione	$\neg\,(\neg P) \;\Leftrightarrow\; P$	
terzo escluso	$P \vee \neg P \;\Leftrightarrow\; \mathsf{T}$	
contraddizione	$P \wedge \neg P \;\Leftrightarrow\; \mathsf{F}$	
De Morgan	$\neg(P \vee Q) \;\Leftrightarrow\; \neg P \wedge \neg Q$	$\neg(P \wedge Q) \;\Leftrightarrow\; \neg P \vee \neg Q$

Tabella 8.2: Leggi per negazione

riflessività	$P \;\Leftrightarrow\; P$
simmetria	$(P \Leftrightarrow Q) \;\Leftrightarrow\; (Q \Leftrightarrow P)$
eliminazione dell'implicazione	$P \Rightarrow Q \;\Leftrightarrow\; \neg P \vee Q$
eliminazione dell'implicazione negata	$\neg(P \Rightarrow Q) \;\Leftrightarrow\; P \wedge \neg Q$
eliminazione della doppia implicazione (1)	$(P \Leftrightarrow Q) \;\Leftrightarrow\; (P \Rightarrow Q) \wedge (Q \Rightarrow P)$
eliminazione della doppia implicazione (2)	$(P \Leftrightarrow Q) \;\Leftrightarrow\; (P \wedge Q) \vee (\neg P \wedge \neg Q)$

Tabella 8.3: Leggi di altri connettivi e di eliminazione

Le leggi elencate descrivono alcune proprietà algebriche dei connettivi logici (vedi quelle per congiunzione, disgiunzione e negazione), oppure consentono di eliminare un connettivo sostituendolo

[5] Lo stesso vale per le identità algebriche: per esempio $x \cdot (y + z) = x \cdot y + x \cdot z$ rappresenta infinite uguaglianze, ottenute sostituendo le variabili con arbitrarie espressioni.

[6] Per semplicità di esposizione, non ci preoccupiamo di introdurre un insieme minimale di leggi, quindi alcune potrebbero essere dimostrate con quelle introdotte precedentemente.

con una opportuna combinazione di altri connettivi. Grazie alla legge di simmetria della doppia implicazione $((P \Leftrightarrow Q) \Leftrightarrow (Q \Leftrightarrow P))$, ogni legge può essere usata in entrambe le direzioni (sostituendo in un certo contesto il membro destro con il sinistro oppure il sinistro con il destro).

Per ognuna delle leggi elencate occorrebbe verificare che sia una tautologia: lo mostriamo solo per una legge di De Morgan, lasciando le altre come esercizio per il lettore.

Esempio 8.3.3 (correttezza delle leggi di De Morgan)**.** *Mostriamo che la prima legge di De Morgan è una tautologia (la dimostrazione per la seconda è del tutto analoga), usando la tavola di verità della formula.*

P	Q	$\neg$	$(P$	$\vee$	$Q)$	$\Leftrightarrow$	$\neg$	P	$\wedge$	$\neg$	Q
f	f	t	f	f	f	t	t	f	t	t	f
f	t	f	f	t	t	t	t	f	f	f	t
t	f	f	t	t	f	t	f	t	f	t	f
t	t	f	t	t	t	t	f	t	f	f	t
		(3)	(1)	(2)	(1)	(**4**)	(2)	(1)	(3)	(2)	(1)

Come si vede dalla colonna (4) la formula è una tautologia.

Sfruttando le leggi introdotte siamo ora in grado di dimostrare che altre formule sono tautologie. È importante sottolineare che ogni tautologia può essere usata a sua volta come giustificazione in dimostrazioni successive, con un meccanismo simile alla costruzione delle dimostrazioni di teoremi in matematica: se proviamo separatamente alcuni lemmi, essi possono essere utilizzati senza bisogno di ri-dimostrarli nella dimostrazione del teorema principale.

Per dimostrare una tautologia $P \Leftrightarrow Q$, come anticipato possiamo partire da P e cercare di trasformarla in Q con una sequenza di passi $P \Leftrightarrow R_1 \Leftrightarrow \ldots \Leftrightarrow R_n \Leftrightarrow Q$ ognuno dei quali giustificato da una legge o da una tautologia già dimostrata. Il fatto che $P \Leftrightarrow Q$ sia una tautologia segue dalla transitività di $\Leftrightarrow$. Ovviamente dato che $\Leftrightarrow$ è simmetrica possiamo anche partire da Q per arrivare a P, oppure possiamo ridurre sia P che Q ad una terza formula R equivalente ad entrambe.

Esempio 8.3.4 (complemento e assorbimento)**.** *Dimostriamo che le seguenti formule sono tautologie:*

complemento	$P \vee (\neg P \wedge Q) \;\Leftrightarrow\; P \vee Q$	$P \wedge (\neg P \vee Q) \;\Leftrightarrow\; P \wedge Q$
assorbimento	$P \vee (P \wedge Q) \;\Leftrightarrow\; P$	$P \wedge (P \vee Q) \;\Leftrightarrow\; P$

Consideriamo la prima legge del complemento. Una buona strategia, che in questo caso funziona, consiste nel partire dalla formula più complessa e nell'usare le leggi per semplificarla finché non si ottiene l'altra. Partiamo quindi da $P \vee (\neg P \wedge Q)$, sottolineando la porzione di formula alla quale viene applicata la legge indicata nella giustificazione, a meno che non sia l'intera formula.

$$\begin{aligned}
P \vee (\neg P \wedge Q) &\Leftrightarrow \underline{(P \vee \neg P)} \wedge (P \vee Q) && \textit{(distributività)}\\
&\Leftrightarrow \mathsf{T} \wedge (P \vee Q) && \textit{(terzo escluso)}\\
&\Leftrightarrow (P \vee Q) \wedge \mathsf{T} && \textit{(commutatività)}\\
&\Leftrightarrow (P \vee Q) && \textit{(unità)}
\end{aligned}$$

In modo del tutto analogo si può dimostrare la seconda legge del complemento, nella quale la congiunzione e la disgiunzione giocano un ruolo simmetrico.

Vediamo invece che la stessa strategia non funziona con la prima legge dell'assorbimento, $P \vee (P \wedge Q) \Leftrightarrow P$. Infatti, partendo dal membro sinistro verrebbe naturale applicare le leggi nella sequenza che segue:

$$\begin{aligned}
P \vee (P \wedge Q) &\Leftrightarrow \underline{(P \vee P)} \wedge (P \vee Q) && \textit{(distributività)}\\
&\Leftrightarrow P \wedge (P \vee Q) && \textit{(idempotenza)}
\end{aligned}$$

Si noti che abbiamo ottenuto il membro sinistro della seconda legge dell'assorbimento, ed è facile intuire che procedendo allo stesso modo otterremmo la formula da cui eravamo partiti, chiudendo

uno sterile ciclo. Quella che segue invece è una dimostrazione corretta della legge, come è facile verificare, ma essa usa in modo non ovvio la legge dell'unità al contrario.

$$\begin{array}{lcll} \underline{P} \vee (P \wedge Q) & \Leftrightarrow & (P \wedge \mathsf{T}) \vee (P \wedge Q) & \textit{(unità), al contrario} \\ & \Leftrightarrow & P \wedge \underline{(\mathsf{T} \vee Q)} & \textit{(distributività), al contrario} \\ & \Leftrightarrow & P \wedge \mathsf{T} & \textit{(commutatività) e (assorbimento)} \\ & \Leftrightarrow & P & \textit{(unità)} \end{array}$$

Riassumendo, spesso una buona strategia consiste nel partire dalla formula più complessa e nell'usare le leggi per semplificarla fino a ottenere la formula equivalente cercata. Ma non sempre questa strategia funziona: in tal caso occorre un pizzico di intuizione per impostare una dimostrazione concisa e corretta.

Esempio 8.3.5 (contronominale)**.** *Dimostriamo la seguente tautologia:*

$$\boxed{\textit{contronominale} \qquad P \Rightarrow Q \;\Leftrightarrow\; \neg Q \Rightarrow \neg P}$$

Partiamo dalla formula più complessa, che in questo caso è il membro destro:

$$\begin{array}{lcll} \neg Q \Rightarrow \neg P & \Leftrightarrow & \underline{\neg(\neg Q)} \vee \neg P & \textit{(eliminazione dell'implicazione)} \\ & \Leftrightarrow & Q \vee \neg P & \textit{(doppia negazione)} \\ & \Leftrightarrow & \neg P \vee Q & \textit{(commutatività)} \\ & \Leftrightarrow & P \Rightarrow Q & \textit{(eliminazione dell'implicazione), al contrario} \end{array}$$

Nel seguito, come da prassi consolidata, tenderemo a semplificare le dimostrazioni non indicando esplicitamente i passaggi nei quali applichiamo le leggi della commutatività, associatività e idempotenza.

Esempio 8.3.6 (le dimostrazioni non sono uniche)**.** *Dimostriamo ora la tautologia*

$$\neg(P \vee (\neg P \wedge Q)) \;\Leftrightarrow\; \neg P \wedge \neg Q$$

Nei riquadri che seguono mostriamo tre diverse dimostrazioni per sostituzione che partono dal membro sinistro. Come si vede, scelte diverse delle leggi da applicare nei vari passi portano a dimostrazioni di lunghezza diversa.

La dimostrazione a sinistra applica in modo sistematico le leggi per semplificare la formula a partire dall'operatore più esterno, la negazione. La dimostrazione in alto a destra invece posticipa l'uso delle leggi di De Morgan, applicandole solo quando la sottoformula più interna è stata semplificata. Infine nella dimostrazione in basso a destra abbiamo riconosciuto che si può applicare, con una opportuna sostituzione, la legge del complemento dimostrata precedentemente, il che permette di ridurre la dimostrazione a due soli passi.

$$\begin{array}{ll} & \neg(P \vee (\neg P \wedge Q)) \\ \Leftrightarrow & \textit{(De Morgan)} \\ & \neg P \wedge \underline{\neg(\neg P \wedge Q)} \\ \Leftrightarrow & \textit{(De Morgan)} \\ & \neg P \wedge (\underline{\neg(\neg P)} \vee \neg Q) \\ \Leftrightarrow & \textit{(doppia negazione)} \\ & \neg P \wedge (P \vee \neg Q) \\ \Leftrightarrow & \textit{(distributività)} \\ & \underline{(\neg P \wedge P)} \vee (\neg P \wedge \neg Q) \\ \Leftrightarrow & \textit{(contraddizione)} \\ & \mathsf{F} \vee (\neg P \wedge \neg Q) \\ \Leftrightarrow & \textit{(unità)} \\ & \neg P \wedge \neg Q \end{array}$$

$$\begin{array}{ll} & \neg(\underline{P \vee (\neg P \wedge Q)}) \\ \Leftrightarrow & \textit{(distributività)} \\ & \neg\big((\underline{P \vee \neg P}) \wedge (P \vee Q)\big) \\ \Leftrightarrow & \textit{(terzo escluso)} \\ & \neg\big(\underline{\mathsf{T} \wedge (P \vee Q)}\big) \\ \Leftrightarrow & \textit{(unità)} \\ & \neg(P \vee Q) \\ \Leftrightarrow & \textit{(De Morgan)} \\ & \neg P \wedge \neg Q \end{array}$$

$$\begin{array}{ll} & \neg(\underline{P \vee (\neg P \wedge Q)}) \\ \Leftrightarrow & \textit{(complemento)} \\ & \neg(P \vee Q) \\ \Leftrightarrow & \textit{(De Morgan)} \\ & \neg P \wedge \neg Q \end{array}$$

Come ultimo esempio di questa sezione vediamo la dimostrazione di una tautologia che non è una doppia implicazione, sfruttando il fatto che una formula P è una tautologia se e solo se lo è $P \Leftrightarrow \mathsf{T}$.

Esempio 8.3.7 (Modus Ponens)**.** *Dimostrare che la formula* $((P \Rightarrow Q) \wedge P) \Rightarrow Q$*, chiamata* Modus Ponens*, è una tautologia. Mostriamo che è equivalente a* T*.*

$$
\begin{array}{lcll}
((\underline{P \Rightarrow Q}) \wedge P) \Rightarrow Q & \Leftrightarrow & (\underline{(\neg P \vee Q) \wedge P}) \Rightarrow Q & \textit{(eliminazione dell'implicazione)} \\
& \Leftrightarrow & Q \wedge P \Rightarrow Q & \textit{(complemento)} \\
& \Leftrightarrow & \underline{\neg(Q \wedge P)} \vee Q & \textit{(eliminazione dell'implicazione)} \\
& \Leftrightarrow & (\underline{\neg Q} \vee \neg P) \ \underline{\vee \ Q} & \textit{(De Morgan)} \\
& \Leftrightarrow & \mathsf{T} \vee \neg P & \textit{(commutatività), (associatività), (terzo escluso)} \\
& \Leftrightarrow & \mathsf{T} & \textit{(assorbimento)}
\end{array}
$$

I sistemi di dimostrazioni

In questa sezione vogliamo mostrare che le dimostrazioni per sostituzione di tautologie appena viste sono basate formalmente su di un *sistema di dimostrazioni* per il calcolo proposizionale. Ne approfittiamo per introdurre questo concetto, che si può applicare a qualunque logica, nella sua generalità.

Un sistema di dimostrazioni per una data logica stabilisce in quale modo si possono costruire delle dimostrazioni partendo da un insieme di premesse e applicando le regole di inferenza disponibili. L'obiettivo di una dimostrazione è in generale di mostrare che una data formula è conseguenza logica delle premesse.

Definizione 8.3.8 (sistema di dimostrazioni)**.** *Dato un insieme di formule* Δ *(letto "Delta"), un* SISTEMA DI DIMOSTRAZIONI *(in inglese* PROOF SYSTEM*) per* Δ *è un insieme di* REGOLE DI INFERENZA $\mathcal{R}$*. Una regola di inferenza* $r \in \mathcal{R}$ *ha la struttura:*

$$\frac{P_1 \quad \cdots \quad P_n}{P} \ [r]$$

dove P *è la* CONSEGUENZA *e* $P_1, \ldots, P_n$ *sono le* PREMESSE*, per* $n \geq 0$*. Se* $n = 0$ *la regola è chiamata anche un* ASSIOMA*, altrimenti se* $n > 0$ *è chiamata anche una regola di inferenza* PROPRIA*.*

Possiamo leggere una regola di inferenza come quella mostrata così: *se abbiamo una dimostrazione per le formule* P_1*, ... e* P_n*, allora abbiamo anche una dimostrazione per* P*.* Equivalentemente, possiamo leggerla come *per dimostrare* P *è sufficiente dimostrare* P_1*, ... e* P_n*.* Si può applicare questa lettura per esempio alla regola [**PDS**] presentata sopra per il calcolo proposizionale, che ha una sola premessa: *"se abbiamo una dimostrazione per* $Q \Leftrightarrow R$*, allora abbiamo una dimostrazione per* $P \Leftrightarrow P\left[{}^{R}/_{Q}\right]$*".*

Definizione 8.3.9 (dimostrazione)**.** *Una* DIMOSTRAZIONE *in un proof system* $\mathcal{R}$ *di una formula* $Q \in \Delta$ *a partire da un insieme di premesse* $\Gamma \subseteq \Delta$ *è una sequenza di formule* $Q_1, Q_2, \ldots, Q_n$ *in cui*

(1) *ogni formula* Q_i *è un elemento di* Γ *oppure è ottenuta applicando una regola di inferenza di* $\mathcal{R}$ *a partire dalle formule in* Γ *o in* $Q_1, \ldots, Q_{i-1}$*;*

(2) Q_n *è proprio* Q*.*

Se esiste una dimostrazione di Q *a partire da* Γ *in* $\mathcal{R}$ *scriveremo*

$$\Gamma \vdash_{\mathcal{R}} Q \qquad \textit{(}Q \textit{ è dimostrabile da } \Gamma \textit{ (in } \mathcal{R}\textit{))}$$

La (1) mette in luce il fatto che, in una dimostrazione, le premesse che si possono utilizzare in un passo sono non solo le premesse date Γ, ma anche tutte le formule derivate nei passi di dimostrazione precedenti.

Il concetto di dimostrazione è puramente sintattico. La adeguatezza di un sistema di dimostrazioni nel rappresentare sintatticamente il concetto semantico di conseguenza logica può essere espressa dalle sue caratteristiche di *correttezza* e *completezza*.

Definizione 8.3.10 (correttezza e completezza)**.** *Un sistema di dimostrazioni $\mathcal{R}$ per Δ è detto* CORRETTO *se per ogni formula $P \in \Delta$ e ogni insieme di formule $\Gamma \subseteq \Delta$*

$$\Gamma \vdash_{\mathcal{R}} P \qquad \textit{implica} \qquad \Gamma \models P \qquad\qquad \textit{(correttezza)}$$

cioè se consente di derivare solo conclusioni che sono effettivamente conseguenze logiche delle premesse date. La correttezza è un requisito che qualunque sistema di dimostrazioni deve soddisfare.

Un sistema di dimostrazioni $\mathcal{R}$ è COMPLETO *se per ogni formula $P \in \Delta$ e ogni insieme di formule $\Gamma \subseteq \Delta$*

$$\Gamma \models P \qquad \textit{implica} \qquad \Gamma \vdash_{\mathcal{R}} P \qquad\qquad \textit{(completezza)}$$

cioè consente di dimostrare una formula a partire da un insieme di premesse se la prima è conseguenza logica del secondo.

Torniamo ora a parlare del calcolo proposizionale. Un sistema di dimostrazioni per tale logica ha lo scopo di dimostrare che due formule proposizionali P e Q sono logicamente equivalenti ($P \equiv Q$), oppure che una formula P è conseguenza logica di un insieme di formule Γ ($\Gamma \models P$). Per questo scopo giocano un ruolo fondamentale le tautologie: per dimostrare una equivalenza o una conseguenza logica è sufficiente dimostare che una opportuna formula proposizionale è una tautologia. Infatti abbiamo il seguente risultato:

Proposizione 8.3.11 (equivalenza e conseguenza logica come tautologie)**.** *Siano P e Q formule proposizionali, e $\Gamma = \{P_1, \ldots, P_n\}$ un insieme finito di formule proposizionali.*[7] *Allora*

1. *$P \equiv Q$ se e solo se $P \Leftrightarrow Q$ è una tautologia.*
2. *$\Gamma \models Q$ se e solo se $P_1 \wedge \ldots \wedge P_n \Rightarrow Q$ è una tautologia, dove $\Gamma = \{P_1, \ldots, P_n\}$.*

Dimostrazione.

1. Abbiamo che $P \Leftrightarrow Q$ è una tautologia se e solo se è vera per ogni interpretazione (Definizione 8.2.5), quindi se e solo se per ogni interpretazione P e Q assumono lo stesso valore (Definizione 8.1.8), se e solo se $P \equiv Q$ (Definizione 8.1.10).
2. $P_1 \wedge \ldots \wedge P_n \Rightarrow Q$ è una tautologia se e solo se è vera per ogni interpretazione, quindi se e solo se ogni interpretazione $\mathcal{I}$ che rende vera la formula $P_1 \wedge \ldots \wedge P_n$ rende anche vera Q (Definizione 8.1.8). Ma, per la stessa definizione e per la Definizione 8.1.10, $\mathcal{I}$ rende vera $P_1 \wedge \ldots \wedge P_n$ se e solo se $\mathcal{I}$ è un modello di $\{P_1, \ldots P_n\}$, quindi ogni modello di $\{P_1, \ldots P_n\}$ è anche un modello di Q, cioè $\{P_1, \ldots P_n\} \models Q$.

■

Questo fatto ci permette di rileggere le dimostrazioni di tautologie presentate nella sezione precedente come dimostrazioni di equivalenze logiche. Quelle dimostrazioni sono di fatto dimostrazioni del sistema che definiamo ora.

Definizione 8.3.12 (Proof system $\mathcal{S}_{\Leftrightarrow}$ per il Calcolo Proposizionale)**.** *Il sistema di dimostrazioni $\mathcal{S}_{\Leftrightarrow}$ sull'insieme di formule* **Prop** *è costituito dalle regole di inferenza proprie* Transitività [**TR**] *e* Principio di sostituzione [**PDS**] *e, come assiomi, dall'insieme di tautologie elencate nella Proposizione 8.3.2.*

Abbiamo già presentato e discusso la regola di inferenza [**PDS**]. La regola [**TR**] è questa:

Transitività

$$\frac{P \Leftrightarrow Q \quad Q \Leftrightarrow R}{P \Leftrightarrow R}\ [\mathbf{TR}]$$

Essa si legge *se abbiamo una dimostrazione per $P \Leftrightarrow Q$ e $Q \Leftrightarrow R$ allora abbiamo anche una dimostrazione per $P \Leftrightarrow R$*, oppure *per dimostrare $P \Leftrightarrow R$ è sufficiente dimostare $P \Leftrightarrow Q$ e $Q \Leftrightarrow R$ per una qualche formula Q.*

[7] Se Γ è un insieme infinto vale un risultato simile ma più complesso da enunciare, che tralasciamo per semplicità.

Proposizione 8.3.13 (correttezza di $\mathcal{S}_\Leftrightarrow$)**.** *Il sistema di dimostrazioni $\mathcal{S}_\Leftrightarrow$ è corretto, cioè per ogni insieme $\Gamma \subseteq$ **Prop** e per ogni formula $P \Leftrightarrow Q \in$ **Prop**, se $\Gamma \vdash_{\mathcal{S}_\Leftrightarrow} P \Leftrightarrow Q$ allora $\Gamma \models P \Leftrightarrow Q$.*

Lasciamo la semplice dimostrazione per induzione di questo importante risultato al lettore interessato. Il caso base sfrutta il fatto che gli assiomi in AX sono tautologie, mentre il passo induttivo consiste nel dimostare per le due regole di inferenza che se un'interpretazione rende vere le premesse, allora rende vera anche la conseguenza.

Abbiamo affermato che le dimostrazioni per sostituzione della sezione precedente sono dimostrazioni del proof system $\mathcal{S}_\Leftrightarrow$, ma il lettore attento potrebbe giustamente obiettare che quelle dimostrazioni non hanno la struttura prescritta dalla Definizione 8.3.9. In effetti una dimostrazione per sostituzione ci permette di presentare in modo più compatto una dimostrazione secondo quella definizione, mostrando solo i passi in cui si applica il Principio di sostituzione (le applicazione della Transitività sono implicite), e evitando di ripetere più volte la stessa formula. Per esempio, riportiamo qui per convenienza del lettore la dimostrazione della prima legge del Complemento (Esempio 8.3.4):

$$\begin{array}{rcll} P \vee (\neg P \wedge Q) & \Leftrightarrow & \underline{(P \vee \neg P)} \wedge (P \vee Q) & \text{(distributività)} \\ & \Leftrightarrow & \mathsf{T} \wedge (P \vee Q) & \text{(terzo escluso)} \\ & \Leftrightarrow & (P \vee Q) \wedge \mathsf{T} & \text{(commutatività)} \\ & \Leftrightarrow & (P \vee Q) & \text{(unità)} \end{array}$$

Usando il formato della Definizione 8.3.9 tale dimostrazione avrebbe il seguente formato: una sequenza di doppie implicazioni, per ognuna delle quali indichiamo la regola di inferenza e le premesse usate, e l'ultima delle quali è la formula dimostrata.

$$\underline{P \vee (\neg P \wedge Q)} \Leftrightarrow \underline{(P \vee \neg P) \wedge (P \vee Q)} \quad ([\mathbf{PDS}], \text{distributività}) \quad (8.1)$$

$$\underline{(P \vee \neg P)} \wedge (P \vee Q) \Leftrightarrow \underline{\mathsf{T}} \wedge (P \vee Q) \quad ([\mathbf{PDS}], \text{terzo escluso}) \quad (8.2)$$

$$P \vee (\neg P \wedge Q) \Leftrightarrow \mathsf{T} \wedge (P \vee Q) \quad ([\mathbf{TR}], (8.1) \text{ e } (8.2)) \quad (8.3)$$

$$\underline{\mathsf{T} \wedge (P \vee Q)} \Leftrightarrow \underline{(P \vee Q) \wedge \mathsf{T}} \quad ([\mathbf{PDS}], \text{commutatività}) \quad (8.4)$$

$$P \vee (\neg P \wedge Q) \Leftrightarrow (P \vee Q) \wedge \mathsf{T} \quad ([\mathbf{TR}], (8.3) \text{ e } (8.4)) \quad (8.5)$$

$$\underline{(P \vee Q) \wedge \mathsf{T}} \Leftrightarrow \underline{(P \vee Q)} \quad ([\mathbf{PDS}], \text{unità}) \quad (8.6)$$

$$P \vee (\neg P \wedge Q) \Leftrightarrow (P \vee Q) \quad ([\mathbf{TR}], (8.5) \text{ e } (8.6)) \quad (8.7)$$

Sempre per la Definizione 8.3.9, poiché nella dimostrazione abbiamo usato come premesse (come giustificazioni dei vari passi) l'insieme $\Gamma = \{$*distributività, terzo escluso, commutatività, unità*$\}$, abbiamo di fatto mostrato che

$$\Gamma \vdash_{\mathcal{S}_\Leftrightarrow} \qquad P \vee (\neg P \wedge Q) \;\Leftrightarrow\; P \vee Q$$

che per la correttezza di $\mathcal{S}_\Leftrightarrow$ ci garantisce che la legge del complemento è una conseguenza logica delle leggi in Γ, ma non immediatamente che è una tautologia: in realtà lo è perché abbiamo usato solo tautologie come premesse, come garantito dal seguente risultato.

Proposizione 8.3.14 (tautologie come premesse)**.** *Se $\Gamma \subseteq$ **Prop** contiene solo tautologie e $\Gamma \vdash_{\mathcal{S}_\Leftrightarrow} P$, allora P è una tautologia.*

Dimostrazione. Per la correttezza del proof system, da $\Gamma \vdash_{\mathcal{S}_\Leftrightarrow} P$ possiamo dedurre che $\Gamma \models P$. Quindi P è vera in tutti i modelli di Γ. Ma poiché Γ contiene solo tautologie, qualunque interpretazione è modello di Γ, e quindi di P. Di conseguenza P è una tautologia. ■

Tecniche di dimostrazione e tautologie

Abbiamo visto nella Proposizione 8.3.11 che il calcolo proposizionale permette di internalizzare i concetti di equivalenza e conseguanza logica, rappresentandoli con tautologie. Questo potere espressivo consente di verificare la correttezza di semplici inferenze o tecniche di dimostrazione con il seguente procedimento: si formalizza il "ragionamento" con una formula proposizionale e

si verifica se essa è una tautologia. Il ragionamento sarà corretto se e solo se la formula è una tautologia.[8] Per esempio, consideriamo la seguente tecnica di dimostrazione:

> Per dimostrare che $A \equiv B$ si può dimostrare che se vale A allora vale B, e se vale B allora vale A.

È giusto questo modo di ragionare? Sfruttando la Proposizione 8.3.11 possiamo rappresentare questa affermazione con la formula: $((A \Rightarrow B) \wedge (B \Rightarrow A)) \Rightarrow (A \Leftrightarrow B)$, che è una tautologia (lasciamo al lettore la dimostrazione). Questo dimostra che la tecnica di dimostrazione descritta è corretta, come ci aspettavamo.

In generale gli enunciati dei teoremi hanno spesso la struttura *"se valgono certe Ipotesi allora vale questa Tesi"*. Dimostrare un tale teorema significa dimostrare la conseguenza logica *Ipotesi* $\models$ *Tesi*. Vediamo alcune tecniche di dimostrazione, mostrando che sono corrette perchè corrispondono a delle tautologie.

Esempio 8.3.15 (dimostrazione diretta e con ipotesi non tautologiche)**.** *Una dimostrazione diretta della conseguenza logica Ipotesi* $\models$ *Tesi consiste nel dimostrare direttamente che l'implicazione* (*Ipotesi* $\Rightarrow$ *Tesi*) *è una tautologia, usando solo gli assiomi. La conseguenza logica allora vale per la Proposizione 8.3.11.*

Un'alternativa consiste nel cominciare a dimostrare la Tesi, utilizzando quando necessario una o più delle Ipotesi come premesse per proseguire nella dimostrazione. In pratica questo mostra che nel proof system c'è una dimostrazione di Tesi a partire da Ipotesi, cioè Ipotesi $\vdash$ *Tesi, che per la correttezza del proof system garantisce la conseguenza logica cercata.*

Esempio 8.3.16 (dimostrazione per assurdo)**.** *Dimostrare l'enunciato* "se valgono certe Ipotesi allora vale questa Tesi" per assurdo *significa mostrare che* "se si assumono per vere le Ipotesi e si nega la Tesi, si ottiene una contraddizione"*. Nel calcolo proposizionale possiamo rappresentare l'enunciato del teorema con* (*Ipotesi* $\Rightarrow$ *Tesi*)*, e l'enunciato della dimostrazione per assurdo come* (*Ipotesi* $\wedge \neg$ *Tesi* $\Leftrightarrow \mathsf{F}$)*, ricordando che una contraddizione è una formula sempre falsa. Quindi dimostrare la correttezza della tecnica di dimostrazione per assurdo consiste nel dimostrare che la seguente formula è una tautologia:*

$$(\mathit{Ipotesi} \Rightarrow \mathit{Tesi}) \quad \Leftrightarrow \quad (\mathit{Ipotesi} \wedge \neg \mathit{Tesi} \;\Leftrightarrow\; \mathsf{F})$$

Infatti, chiamando Ip le Ipotesi, mostriamo che entrambi i membri sono equivalenti alla stessa formula $\neg \mathit{Ip} \vee \mathit{Tesi}$*, e quindi possiamo concludere per transitività.*

$$\mathit{Ip} \Rightarrow \mathit{Tesi} \quad \Leftrightarrow \quad \neg \mathit{Ip} \vee \mathit{Tesi} \qquad \textit{(eliminazione dell'implicazione)}$$

$$\begin{array}{lll}
(\mathit{Ip} \wedge \neg \mathit{Tesi} \;\Leftrightarrow\; \mathsf{F}) & & \\
\quad \Leftrightarrow (\underline{\mathit{Ip} \wedge \neg \mathit{Tesi} \wedge \mathsf{F}}) \vee (\neg(\mathit{Ip} \wedge \neg \mathit{Tesi}) \wedge \underline{\neg(\mathsf{F})}) & & \textit{(elim. doppia implicazione)} \\
\quad \Leftrightarrow \mathsf{F} \vee (\underline{\neg(\mathit{Ip} \wedge \neg \mathit{Tesi})} \wedge \mathsf{T}) & & \textit{(assorbimento), (F:T)} \\
\quad \Leftrightarrow \neg \mathit{Ip} \vee \neg(\neg \mathit{Tesi}) & & \textit{(unità), (De Morgan)} \\
\quad \Leftrightarrow \neg \mathit{Ip} \vee \mathit{Tesi} & & \textit{(doppia negazione)}
\end{array}$$

Esempio 8.3.17 (dimostrazione per contrapposizione)**.** *Dimostrare* per contrapposizione *l'enunciato* "se valgono certe Ipotesi allora vale questa Tesi" *consiste nel dimostrare che* "se non vale la Tesi allora non valgono le Ipotesi"*. Quindi consiste nel dimostrare* (*Ipotesi* $\Rightarrow$ *Tesi*)*, l'enunciato del teorema, mostrando che vale* ($\neg$*Tesi* $\Rightarrow \neg$*Ipotesi*)*. La correttezza di questa tecnica è garantita dal fatto che la seguente formula* contronominale *è una tautologia, come mostrato nell'Esempio 8.3.5:*

$$(\mathit{Ipotesi} \Rightarrow \mathit{Tesi}) \;\Leftrightarrow\; (\neg \mathit{Tesi} \Rightarrow \neg \mathit{Ipotesi})$$

[8] Avevamo già anticipato questa tecnica negli Esempi 8.2.8 e 8.2.9: ora ne possiamo certificare la correttezza.

8.4 Esercizi su calcolo proposizionale

Esercizio 8.4.1. *Quali delle seguenti sono proposizioni?*

1. *"*$2 + 2 = 4$*"*
2. *"*$3 \times 5 = 10$*"*
3. *"La capitale dell'Australia è Sydney?"*
4. *"La capitale dell'Australia è Sydney"*
5. *"A scuola è vietato fumare"*
6. *"Non fumare a scuola!"*
7. *"Posso fumare a scuola?"*
8. *"Non so se posso fumare a scuola"*
9. *"Che ore sono?"*
10. *"Esiste un valore che sommato a 1 dà 2"*
11. *"*$x + 1 = 2$*"*

Esercizio 8.4.2. *Dire con quali connettivi possono essere resi i termini evidenziati nelle seguenti proposizioni:*

1. *"Mario è agile,* ***ma*** *Rosario è forte"*
2. *"Paolo è in campo,* ***anche se*** *ha la febbre alta"*
3. *"Johnny è in Italia* ***senza*** *avere il passaporto"*
4. *"Il fantasma appare nel castello* ***esattamente*** *a mezzanotte"*

Esercizio 8.4.3 (Quanti connettivi logici esistono [Difficile])**.** *I connettivi logici che abbiamo presentato come funzioni sui booleani nella Definizione 8.1.6 sono quelli più "intuitivi", ma esistono molte altre possibilità di combinare valori booleani. Per esempio, per analogia con gli insiemi, si potrebbe definire il connettivo* $\setminus$ *come:*

$$(x \setminus y) := (x \wedge \neg y) \tag{8.8}$$

Oppure si potrebbero definire connettivi ternari come

$$(\mathsf{if}\ x\ \mathsf{then}\ y\ \mathsf{else}\ z) \equiv (x \wedge y) \vee (\neg x \wedge z) \tag{8.9}$$

In generale un connettivo con n *argomenti è determinato da una colonna di una tavola di verità con* 2^n *righe.*

1. *Quanti possibili connettivi unari diversi esistono?*
2. *Quanti possibili connettivi binari diversi esistono?*
3. *Quanti possibili connettivi* n*-ari esistono?*
4. *Data la colonna che identifica un qualsiasi connettivo* n*-ario, definire una tecnica per costruire una formula equivalente che usi solo i connettivi* $\neg$*,* $\wedge$ *e* $\vee$*.*

Esercizio 8.4.4. *Dimostrare per induzione che ogni formula in* **Prop** *è equivalente a una formula scritta solo con i connettivi* $\wedge$ *e* $\neg$*.*

Esercizio 8.4.5. *Costruire le tavole di verità degli operatori* $(P \setminus Q)$ *e* $(\mathsf{if}\ P\ \mathsf{then}\ Q\ \mathsf{else}\ R)$ *definiti nei punti (8.8) e (8.9).*

Esercizio 8.4.6. *Costruire le tavole di verità delle seguenti formule e indicare quali di queste sono tautologie*

1. $((P \wedge (Q \vee R)) \Rightarrow (P \vee (Q \wedge R)))$
2. $((P \vee (Q \wedge R)) \Rightarrow (P \wedge (Q \vee R)))$
3. $((P \wedge (Q \Rightarrow R)) \Rightarrow (Q \Rightarrow (P \wedge R)))$
4. $((Q \Rightarrow (P \wedge R)) \Rightarrow (P \wedge (Q \Rightarrow R)))$
5. $((P \Rightarrow (Q \vee R)) \Leftrightarrow (Q \Rightarrow (P \vee R)))$
6. $((\neg(P \wedge Q) \Rightarrow R) \Rightarrow (Q \vee \neg(P \wedge R)))$
7. $\neg(Q \wedge (R \Rightarrow (P \wedge R)))$
8. $((Q \wedge R) \vee ((P \wedge Q) \vee (P \wedge R)))$
9. $\neg((P \vee (Q \vee R)) \Rightarrow (P \wedge (Q \wedge R)))$

Esercizio 8.4.7. *Per ognuna delle seguenti formule, determinare se è una tautologia oppure no. Se non è una tautologia fornire un'interpretazione che la rende falsa* senza *costruire la tavola di verità, altrimenti costruire la tavola.*

1. $(((P \wedge Q) \vee P) \Leftrightarrow P)$
2. $(P \Rightarrow (Q \Rightarrow \neg R))$
3. $(P \Rightarrow (P \vee Q))$
4. $((P \Rightarrow Q) \Rightarrow \neg P)$

Esercizio 8.4.8. *Si formalizzino le seguenti proposizioni:*

1. *"Aldo va al cinema ma Dario no"*
2. *"Luigi andrà al cinema o andrà al teatro"*
3. *"Se ho lezione di LMB allora è martedì o è venerdì"*
4. *"Non puoi montare sulle montagne russe se sei più basso di un metro e se non hai più di 16 anni"*

Esercizio 8.4.9. *Le seguenti proposizioni in linguaggio naturale esprimono delle implicazioni (semplici o doppie) tra i simboli proposizionali V (*"io vado al cinema"*) e R (*"Tu resti a casa"*). Indicare per ognuna di esse una formula proposizionale che la rappresenta.*

1. *"Io vado al cinema se tu resti a casa"*
2. *"Io vado al cinema solo se tu resti a casa"*
3. *"Io vado al cinema se e solo se tu resti a casa"*
4. *"Perché io vada al cinema è necessario che tu resti a casa"*
5. *"Perché io vada al cinema è sufficiente che tu resti a casa"*
6. *"Condizione necessaria e sufficiente perché io vada al cinema è che tu resti a casa"*

Esercizio 8.4.10. *Dire se le seguenti formule sono tautologie, contraddizioni o soddisfacibili:*

1. $((P \wedge Q) \Rightarrow (P \vee Q))$
2. $((P \wedge Q) \Rightarrow P)$

3. $((P \vee Q) \Rightarrow P)$
4. $(((P \vee (Q \vee R)) \wedge (\neg P \vee \neg Q)) \wedge ((\neg P \vee \neg R) \wedge \neg(Q \wedge R)))$

Esercizio 8.4.11. *Aldo, Barbara e Carlo sono tre studenti che hanno sostenuto un esame. Ponendo:*

- $A =$ *"Aldo ha superato l'esame"*
- $B =$ *"Barbara ha superato l'esame"*
- $C =$ *"Carlo ha superato l'esame"*

determinare le proposizioni composte che traducono le seguenti proposizioni:

1. *"Solo Carlo ha superato l'esame"*
2. *"Solo Aldo non ha superato l'esame"*
3. *"Solo uno tra Aldo, Barbara e Carlo ha superato l'esame"*
4. *"Almeno uno tra Aldo, Barbara e Carlo ha superato l'esame"*
5. *"Almeno due tra Aldo, Barbara e Carlo hanno superato l'esame"*
6. *"Al più due tra Aldo, Barbara e Carlo hanno superato l'esame"*
7. *"Esattamente due tra Aldo, Barbara e Carlo hanno superato l'esame"*

Esercizio 8.4.12. *Aldo, Barbara e Carlo sono gli unici tre membri di una commissione che vota una proposta. Ponendo:*

- $A =$ *"Aldo vota a favore"*
- $B =$ *"Barbara vota a favore"*
- $C =$ *"Carlo vota a favore"*

determinare le proposizioni composte che traducono le seguenti proposizioni:

1. *"La votazione è stata unanime"*
2. *"La proposta è passata a maggioranza"*
3. *"La proposta è stata respinta, ma non all'unanimità"*

Esercizio 8.4.13. *Sappiamo che "sono ammesse al concorso le persone che sono laureate e che hanno meno di trent'anni o hanno figli" e che:*

- *"Aldo non è laureato, ha ventisei anni e un figlio";*
- *"Barbara è laureata, ha quarant'anni e due figli";*
- *"Carlo è laureato, ha trentadue anni e non ha figli".*

Aldo può partecipare al concorso? E Barbara? E Carlo?

Esercizio 8.4.14.

1. *Sapendo che "Il colpevole è il cuoco o la cameriera" e che "Il colpevole è l'autista o la cameriera", possiamo concludere che "Il colpevole è il cuoco o l'autista"? Motivare la risposta.*
2. *Sapendo che "O la ventola è fuori asse o il meccanismo di calibrazione è alterato" e che "Ho controllato l'allineamento della ventola ed è ok", possiamo concludere che "il meccanismo di calibrazione è fuori fase"? Motivare la risposta.*

3. *Sapendo che "Risolvete tutti gli esercizi di queste note e superate l'esame" e che "Non risolvete tutti gli esercizi di queste note" possiamo concludere che "Non superate l'esame"? Motivare la risposta.*

4. *Sapendo che "Risolvete tutti gli esercizi per casa e superate l'esame" e che "Non risolvete tutti gli esercizi per casa" possiamo concludere che "Superate l'esame"? Motivare la risposta.*

Esercizio 8.4.15. *Mostrare che le seguenti formule sono tautologie usando dimostrazioni per sostituzione:*

1. $((A \wedge B) \Rightarrow (A \vee B))$
2. $(A \Rightarrow (B \Rightarrow (A \wedge B)))$
3. $(A \Rightarrow (\neg A \Rightarrow B))$
4. $((A \Rightarrow B) \Leftrightarrow \neg(A \wedge \neg B))$
5. $((A \vee B) \Leftrightarrow (\neg A \Rightarrow B))$
6. $((A \Rightarrow B) \Rightarrow ((B \Rightarrow C) \Rightarrow (A \Rightarrow C)))$
7. $((A \Rightarrow (B \Rightarrow C)) \Leftrightarrow ((A \Rightarrow B) \Rightarrow (A \Rightarrow C)))$
8. $((A \Rightarrow C) \Rightarrow ((B \Rightarrow C) \Rightarrow ((A \vee B) \Rightarrow C)))$

Esercizio 8.4.16. *Dimostrare le seguenti equivalenze logiche oppure fornire un controesempio:*

1. $((P \vee \neg Q) \equiv (\neg P \wedge Q))$
2. $((P \vee Q) \Rightarrow (P \wedge Q)) \equiv \mathsf{T}$
3. $(\neg P \Rightarrow (P \Rightarrow Q)) \equiv \mathsf{T}$
4. $(P \Rightarrow (Q \Rightarrow R)) \equiv ((P \Rightarrow Q) \Rightarrow R)$
5. $\big((P \Rightarrow (Q \Rightarrow R)) \Rightarrow \big((P \Rightarrow Q) \Rightarrow (P \Rightarrow R)\big)\big) \equiv \mathsf{T}$
6. $\big((P \Rightarrow (Q \Rightarrow R)) \Leftrightarrow \big((P \Rightarrow Q) \Rightarrow (P \Rightarrow R)\big)\big) \equiv \mathsf{T}$

8.5 Cenni di logica dei predicati: motivazioni e sintassi

Il calcolo proposizionale introdotto nella prima parte di questo capitolo costituisce il nucleo di tutte le logiche classiche, ma ha un potere espressivo alquanto limitato. Infatti anche se consente di formalizzare in modo soddisfacente la struttura logica di proposizioni anche complesse (come inferenze e dimostrazioni) non fornisce strumenti per rappresentare gli elementi del dominio del discorso, le loro proprietà e le relazioni tra di essi. La *logica dei predicati* (o *logica del primo ordine*) arricchisce il calcolo proposizionale con costrutti sintattici che permettono appunto di esprimere predicati su proprietà e relazioni tra specifici elementi del dominio. Inoltre, con i quantificatori, permette di esprimere che una proprietà vale per tutti gli elementi o per almeno un elemento del dominio.

Nel resto di questa sezione, dopo aver motivato con alcuni esempi l'introduzione di una logica più espressiva, presentiamo la sintassi i della logica dei predicati. Vedremo nella Sezione 8.6 come questa logica permette di formalizzare in modo adeguato proposizioni ben più complesse di quelle viste con il calcolo proposizionale. Quindi nella Sezione 8.7 descriveremo come deve essere arricchita la nozione di interpretazione per consentire di associare un valore di verità alle formule predicative, e ne presenteremo la semantica in modo induttivo. Infine presenteremo alcune leggi per i quantificatori nella Sezione 8.8, mostrando anche per questa logica come le dimostrazioni per sostituzione possono essere utilizzate per mostrare l'equivalenza logica di formule.

Sull'espressività della logica dei predicati

L'espressività del calcolo proposizionale è abbastanza limitata. Per esempio, pensiamo di voler formalizzare un enunciato che parla di persone, come

"Se Anna è la mamma di Bruno e Bruno ha figli, allora Anna è nonna"

Usando la tecnica discussa nella Sezione 8.1 (si veda l'Esempio 8.1.4) dobbiamo introdurre un simbolo per ogni proposizione elementare, per esempio A per *"Anna è la mamma di Bruno"*, B per *"Bruno ha figli"* e C per *"Anna è nonna"*. La formula proposizionale risultante sarebbe $((A \wedge B) \Rightarrow C)$: essa evidenzia correttamente la struttura logica (la congiunzione e l'implicazione) ma poichè le persone coinvolte non hanno una rappresentazione esplicita, la formula non permette di capire che Anna è coinvolta sia nella premessa A che nella conseguenza C dell'implicazione, e che entrambe le premesse A e B parlano di Bruno. Se poi consideriamo la frase *"Maria è la mamma di Giulia"*, questa è una proposizione elementare e quindi dobbiamo rappresentarla con un nuovo simbolo: non abbiamo la possibilità di mettere in evidenza che essa esprime la stessa relazione tra persone della proposizione rappresentata sopra da A.

Quest'analisi rende evidente che, in situazioni molto frequenti, vorremmo poter rappresentare nella sintassi delle formule: (1) le persone coinvolte, (2) le proprietà delle persone (*"aver figli"*, *"essere nonna"*) e (3) le relazioni tra persone (*"essere mamma di"*).

La logica dei predicati permette di rappresentare esplicitamente queste entità, grazie a una sintassi più ricca di quella proposizionale. Infatti, come vedremo, possiamo formalizzare le frasi viste sopra per esempio come

$$((mamma(Anna, Bruno) \wedge haFigli(Bruno)) \Rightarrow èNonna(Anna)) \qquad \text{e} \qquad mamma(Maria, Giulia)$$

in modo di gran lunga più espressivo che con le formule proposizionali.

Naturalmente poter parlare di "persone" è solo un caso particolare: potremmo voler esprimere proprietà e/o relazioni su numeri (*"7 è un numero primo"*, *"6 è il doppio di 3"*), su frutti (*"questa mela è rossa"*), su animali (*"Fido è un bassotto"*), o anche su elementi di tipologie diverse (*"se Fido è il cane di Giorgio allora mangia una mela rossa"*).

In generale nella logica dei predicati possiamo rappresentare (con dei *simboli di costante* o dei *termini*) gli *elementi* appartenenti a un certo insieme (*dominio*). Di questi elementi possiamo rappresentare delle proprietà e delle relazioni tra essi (con dei *simboli di predicato*). Negli esempi presentati sopra, abbiamo usato i simboli di costante *"Anna"*, *"Bruno"*, *"Maria"* e *"Giulia"*, i simboli di predicato con un solo argomento (detti *unari*) *"haFigli"* ed *"èNonna"* e il simbolo di predicato a due argomenti (detto *binario*) *"mamma"*.

Quantificazione esistenziale e universale

La possibilità di riferire specifici elementi del dominio di interesse in una formula è solo una premessa necessaria per introdurre i *quantificatori*, l'ingrediente più caratterizzante della sintassi della logica dei predicati.

Nella lingua italiana usiamo comunemente pronomi o aggettivi indefiniti che indicano cose o persone senza specificarne con precisione né l'identità né la quantità, come *"tutti"*, *"qualche"*, *"alcuni"*, *"ogni"*, *"nessuno"*, ecc. Per esempio, parlando di persone, possiamo dire *"tutti gli uomini sono mortali"*, *"nessuno pesa più di 100 chili"*, *"qualcuno è più alto di suo padre"*. Esaminiamone il significato:

- *Tutti* (e sinonimi): permette di asserire che una proprietà vale per tutti gli elementi del dominio o *"universo"*, nessuno escluso. Esprime quindi una *quantificazione universale*.
- *Alcuni* (e sinonimi): permette di affermare che una proprietà vale per almeno un elemento del dominio. Non dice né quanti né quali, ma garantisce l'esistenza di almeno un elemento che la soddisfa. Esprime quindi una *quantificazione esistenziale*.
- *Nessuno* (e sinonimi): permette di asserire che una proprietà non vale per alcun elemento del dominio. Attenzione, è l'opposto di *alcuni*, non è l'opposto di *tutti*! Equivale a dire *tutti ... non ...* e quindi quantifica in maniera universale la negazione di un'enunciato.

La logica dei predicati permette di rappresentare esplicitamente le quantificazioni usando le *variabili* e i *quantificatori*.

Le variabili, di solito chiamate $x, y, z, \ldots$, rappresentano generici elementi del dominio e quindi permettono di rappresentare enunciati generici che assumono un valore di verità che dipende da quali elementi vengono associati alle variabili. Per esempio, parlando dei numeri naturali, possiamo scrivere l'enunciato *"x è un numero primo"*, che possiamo rappresentare con la formula della logica dei predicati $primo(x)$. Chiaramente la verità di questa formula dipende da quale elemento sostituiamo a x: $primo(5)$ è vero, mentre $primo(12)$ è falso.

I quantificatori servono per descrivere in che modo una variabile che compare in una formula deve essere sostituita da elementi del dominio, per verificare se l'intera formula è vera o falsa. Essi hanno la seguente forma, dove in generale la formula P contiene una o più occorrenze della variabile x:

QUANTIFICAZIONE ESISTENZIALE: $(\exists x \,.\, P)$, che si legge *"esiste un x tale che P vale"*

QUANTIFICAZIONE UNIVERSALE: $(\forall x \,.\, P)$, che si legge *"per ogni x vale P"*

Per esempio, considerando il dominio dei numeri naturali, $(\exists x \,.\, primo(x))$ si legge *"esiste un x tale che $primo(x)$ vale"*, e cioè *"esiste un numero naturale che è primo"*. Si noti che mentre $primo(x)$ essendo una formula generica non ha un valore di verità, $(\exists x \,.\, primo(x))$ lo ha: esso vale "vero" *se esiste almeno un elemento nel dominio che associato a x rende la formula $primo(x)$ vera*. Quindi $(\exists x \,.\, primo(x))$ è vera.

Invece $(\forall x \,.\, primo(x))$ si legge *"per ogni x vale $primo(x)$"*, cioè *"ogni numero naturale è primo"*. Quindi $(\forall x \,.\, primo(x))$ è vera se *ogni elemento d del dominio, se associato a x, rende la formula $primo(x)$ vera*. Poichè ci sono numeri naturali che non sono primi, la formula è falsa (basta considerare la sostituzione di x con 4).

La sintassi delle formule predicative

La sintassi delle formule proposizionali presentata nella Definizione 8.1.1 è parametrica rispetto a un insieme di *simboli proposizionali* che, come abbiamo visto, ci servono per rappresentare le proposizioni elementari del dominio del discorso. Anche per la logica dei predicati la sintassi delle formule è parametrica: useremo simboli di costante, di predicato e di funzione presi da un alfabeto fissato a priori. Questi simboli non hanno un significato specifico: quando valuteremo le formule per vedere se sono vere o false su di un *dominio di interpretazione*, dovremo dire qual è il loro significato.

Definizione 8.5.1 (alfabeto del primo ordine)**.** *Un* ALFABETO DEL PRIMO ORDINE $\mathcal{A}$ *è una quadrupla* $\mathcal{A} = (\mathcal{C}, \mathcal{F}, \mathcal{P}, \mathcal{V})$ *dove:*

1. $\mathcal{C}$ *è l'insieme dei* SIMBOLI DI COSTANTE*;*

2. $\mathcal{F} = \{\mathcal{F}_n\}_{n\in\mathbb{N}^+}$ *è una famiglia di insiemi di* SIMBOLI DI FUNZIONE*. Per ogni* $n \in \mathbb{N}^+$*,* $\mathcal{F}_n$ *è l'insieme dei* SIMBOLI DI FUNZIONE DI ARIETÀ n *(o* con n argomenti*).*

3. $\mathcal{P} = \{\mathcal{P}_n\}_{n\in\mathbb{N}}$ *è una famiglia di insiemi di* SIMBOLI DI PREDICATO*. Per ogni* $n \in \mathbb{N}$*,* $\mathcal{P}_n$ *è l'insieme dei* SIMBOLI DI PREDICATO DI ARIETÀ n *(o* con n argomenti*).*

4. $\mathcal{V}$ *è l'insieme delle* VARIABILI*.*

Definizione 8.5.2 (Sintassi della logica dei predicati)**.** *Fissato un alfabeto del primo ordine* $\mathcal{A} = (\mathcal{C}, \mathcal{F}, \mathcal{P}, \mathcal{V})$*, l'insieme delle* FORMULE PREDICATIVE **Pred** *è il linguaggio generato dalla categoria sintattica* $\langle Pred\rangle$ *della seguente grammatica:*

Logica dei predicati

$$\begin{array}{lcl}
\langle Pred\rangle & \rightsquigarrow & \langle Atom\rangle \mid \neg\langle Atom\rangle \mid \langle Pred\rangle\langle OpB\rangle\langle Pred\rangle \mid \underbrace{(\forall\langle Var\rangle\,.\,\langle Pred\rangle)}_{n} \mid \underbrace{(\exists\langle Var\rangle\,.\,\langle Pred\rangle)}_{n} \\
\langle Atom\rangle & \rightsquigarrow & \mathsf{T} \mid \mathsf{F} \mid (\langle Prop\rangle) \mid \langle Pide_0\rangle \mid \underbrace{\langle Pide_n\rangle(\overbrace{\langle Term\rangle,\ldots,\langle Term\rangle}^{n})}_{}, n \in \mathbb{N}^+ \\
\underline{\langle Term\rangle} & \rightsquigarrow & \underline{\langle Var\rangle \mid \langle Const\rangle \mid \langle Fide_n\rangle(\overbrace{\langle Term\rangle,\ldots,\langle Term\rangle}^{n}), n \in \mathbb{N}^+} \\
\langle OpB\rangle & \rightsquigarrow & \wedge \mid \vee \mid \Rightarrow \mid \Leftarrow \mid \Leftrightarrow \\
(n \in \mathbb{N}) & & \langle\!\langle\langle Pide_n\rangle\rangle\!\rangle = \mathcal{P}_n \qquad \langle\!\langle\langle Fide_n\rangle\rangle\!\rangle = \mathcal{F}_n \qquad \langle\!\langle\langle Const\rangle\rangle\!\rangle = \mathcal{C} \qquad \langle\!\langle\langle Var\rangle\rangle\!\rangle = \mathcal{V}
\end{array}$$

Nella grammatica abbiamo sottolineato le produzioni nuove rispetto al calcolo proposizionale. L'ultima linea spiega il ruolo delle quattro componenti dell'alfabeto: per esempio, i simboli di predicato di arietà n possono essere usati dove compare la categoria sintattica $\langle Pide_n\rangle$, mentre i simboli di costante possono comparire al posto di $\langle Const\rangle$.

Un TERMINE, cioè un elemento del linguaggio $\langle\!\langle\langle Term\rangle\rangle\!\rangle$, può essere una variabile, un simbolo di costante, oppure un simbolo di funzione di arietà n (per $n \in \mathbb{N}^+$) applicato a esattamente n termini, come descritto dall'ultima produzione. Nel seguito chiameremo **Term** l'insieme $\langle\!\langle\langle Term\rangle\rangle\!\rangle$. Si noti che nei termini non possono comparire simboli di predicato, che invece servono a costruire le *formule*.

Una FORMULA ATOMICA, cioè un elemento del linguaggio $\langle\!\langle\langle Atom\rangle\rangle\!\rangle$, oltre che T, F, o una qualunque formula racchiusa tra parentesi, può essere un simbolo di predicato di arità zero (equivalente a un simbolo proposizionale), oppure un simbolo di predicato di arietà n applicato ad esattamente n termini, con $n \in \mathbb{N}^+$.

Una FORMULA PREDICATIVA (o semplicemente FORMULA), cioè una stringa generata da $\langle Pred\rangle$, può essere costruita in tutti i modi visti con il calcolo proposizionale e in più può essere una FORMULA QUANTIFICATA della forma $(\forall x\,.\,P)$ oppure $(\exists x\,.\,P)$, dove x è una variabile e P una formula.

Osservazione 8.5.3. *Si noti che tutte le formule del calcolo proposizionale sono anche formule predicative, considerando i simboli proposizionali come simboli di predicato di arietà zero, cioè elementi di* $\mathcal{P}_0$*.*

Naturalmente anche la grammatica appena presentata è ambigua, e quindi adotteremo le stesse convenzioni del calcolo proposizionale riassunte nell'Osservazione 8.2.1. Queste convenzioni sono sufficienti perché le nuove produzioni di questa logica (quelle che nella Definizione 8.5.2 sono sottolineate) non introducono ulteriori ambiguità:

- *Le formule quantificate sono sempre obbligatoriamente racchiuse tra parentesi,*[9] *quindi anche se sono annidate è sempre chiaro dove finisce il* campo di azione *di ogni quantificatore (Definizione 8.7.1).*

- *La categoria sintattica di una stringa della forma $p(t_1, \ldots, t_n)$, che potrebbe essere sia $\langle Atom \rangle$ (per la produzione $\langle Atom \rangle \rightsquigarrow \langle Pide_n \rangle(\langle Term \rangle, \ldots, \langle Term \rangle)$) che $\langle Term \rangle$ (per la produzione $\langle Term \rangle \rightsquigarrow \langle Fide_n \rangle(\langle Term \rangle, \ldots, \langle Term \rangle)$), è determinata dal simbolo p: sarà una formula atomica se $p \in \mathcal{P}_n$, e un termine se $p \in \mathcal{F}_n$.*

In realtà, nell'ultimo caso anche non sapendo se p è un simbolo di predicato o di funzione, il contesto in cui $p(t_1, \ldots, t_n)$ compare è sempre sufficiente a disambiguare, perché determina se la stringa deve essere una formula o un termine. Per esempio, se in una formula compare "$42+p(x, y)$" inferiamo che p è un simbolo di funzione perché $p(x, y)$ è argomento del simbolo di funzione $+$, e quindi deve essere un termine; invece se compare in $Q \Rightarrow p(x, y)$ inferiamo che p è un simbolo di predicato perché $p(x, y)$ deve essere una formula.

Vediamo alcuni esempi di alfabeti del primo ordine.

Esempio 8.5.4 (alcuni alfabeti del primo ordine)**.**

1. *L'alfabeto $\mathcal{A}_\mathcal{T} = (\mathcal{C}_\mathcal{T}, \mathcal{F}_\mathcal{T}, \mathcal{P}_\mathcal{T}, \mathcal{V}_\mathcal{T})$ (dove $\mathcal{T}$ sta per* toy*, perché lo useremo per esempi giocattolo), è costituito da:*

- *i simboli di costante $\mathcal{C}_\mathcal{T} = \{k\}$;*
- *i simboli di funzione $\mathcal{F}_\mathcal{T} = \{\mathcal{F}_{\mathcal{T},n}\}_{n \in \mathbb{N}^+}$, con $\mathcal{F}_{\mathcal{T},1} = \{f\}$ e $\mathcal{F}_{\mathcal{T},i} = \varnothing$ per tutti gli $i \in \mathbb{N}^+ \setminus \{1\}$;*
- *i simboli di predicato $\mathcal{P}_\mathcal{T} = \{\mathcal{P}_{\mathcal{T},n}\}_{n \in \mathbb{N}}$, con $\mathcal{P}_{\mathcal{T},1} = \{A\}$ e $\mathcal{P}_{\mathcal{T},1} = \{B\}$, e $\mathcal{P}_{\mathcal{T},i} = \varnothing$ per tutti gli $i \in \mathbb{N} \setminus \{1, 2\}$;*
- *le variabili $\mathcal{V}_\mathcal{T}\mathbb{N} = \{x, y, \ldots\}$.*

Attenzione: *Per alleggerire la notazione, nel seguito adottiamo la convenzione di considerare $\mathcal{F}$ e $\mathcal{P}$ come* insiemi *invece che come* famiglie di insiemi*, indicando per ogni simbolo l'arietà nel seguente modo: se g ha arietà n, allora lo scriveremo $g(\underbrace{\_, \ldots, \_}_{n})$. Per esempio, usando questa convenzione i simboli di funzione e di predicato dell'alfabeto $\mathcal{A}_\mathcal{T}$ appena visto possono essere definiti in modo equivalente ma più semplice come:*

- $\mathcal{F}_\mathcal{T} = \{f(\_)\}$
- $\mathcal{P}_\mathcal{T} = \{A(\_), B(\_, \_)\}$.

2. *L'ALFABETO DEI NATURALI $\mathcal{A}_\mathbb{N} = (\mathcal{C}_\mathbb{N}, \mathcal{F}_\mathbb{N}, \mathcal{P}_\mathbb{N}, \mathcal{V}_\mathbb{N})$ è costituito dai seguenti insiemi (i puntini sospensivi ci permetteranno eventualmente di aggiungere altri simboli nel seguito):*

- *i simboli di costante $\mathcal{C}_\mathbb{N} = \mathbb{N}$;*
- *i simboli di funzione $\mathcal{F}_\mathbb{N} = \{succ(\_), \_ + \_, \_ \times \_, \_/\_, \_\%\_, \ldots\}$;*
- *i simboli di predicato $\mathcal{P}_\mathbb{N} = \{\_ \leq \_, \_ \geq \_, \_ < \_, \_ > \_, \_ = \_, primo(\_), pari(\_), \ldots\}$;*
- *le variabili $\mathcal{V}_\mathbb{N} = \{x, y, \ldots, n, m, \ldots\}$.*

Si noti che abbiamo adottato la convenzione di scrivere $\_op\_$ invece di $op(\_, \_)$ per gli operatori binari per i quali normalmente si adotta la notazione infissa.

3. *L'ALFABETO DELLE PERSONE $\mathcal{A}_\mathcal{P} = (\mathcal{C}_\mathcal{P}, \mathcal{F}_\mathcal{P}, \mathcal{P}_\mathcal{P}, \mathcal{V}_\mathcal{P})$ è costituito dai seguenti insiemi:*

- *i simboli di costante $\mathcal{C}_\mathcal{P} = \{Aldo, Bruna, io, Joe\ Biden, \ldots\}$;*
- *i simboli di funzione $\mathcal{F}_\mathcal{P} = \{padre(\_), nonnoMaterno(\_), \ldots\}$;*
- *i simboli di predicato $\mathcal{P}_\mathcal{P} = \{padre(\_, \_), nonno(\_, \_), figlio(\_, \_), fratelli(\_, \_), \ldots\}$;*

[9] Questa è una convenzione adottata in queste dispense per semplificare la presentazione: nella notazione matematica usuale le parentesi non sono necessarie.

- *le variabili* $\mathcal{V}_\mathcal{P} = \{x, y, \ldots, p, q, \ldots\}$.

4 *L'alfabeto degli insiemi* $\mathcal{A}_\mathcal{S} = (\mathcal{C}_\mathcal{S}, \mathcal{F}_\mathcal{S}, \mathcal{P}_\mathcal{S}, \mathcal{V}_\mathcal{S})$ *è costituito dai seguenti insiemi:*

- *i simboli di costante* $\mathcal{C}_\mathcal{S} = \mathbb{N} = \{\emptyset, \mathbb{N}, \mathbb{Z}, \mathbb{N}^+, \ldots, 0, 1, 2 \ldots\}$*;*
- *i simboli di funzione* $\mathcal{F}_\mathcal{S} = \{\_ \cup \_, \_ \cap \_, \_ \setminus \_, \bar{\_} \ldots\}$*;*
- *i simboli di predicato* $\mathcal{P}_\mathcal{S} = \{isSet(\_), vuoto(\_), \_ \in \_, \_ \subset \_, \_ \subseteq \_, \_ \supset \_, \_ \supseteq \_, \_ = \_, \ldots\}$*;*
- *le variabili* $\mathcal{V}_\mathcal{S} = \{x, y, \ldots, X, Y, \ldots\}$*: useremo le variabili maiuscole per gli insiemi, le minuscole per gli elementi.*

Si noti che per gli alfabeti dei naturali, delle persone e degli insiemi abbiamo introdotto dei simboli che, avendo nomi che suggeriscono un preciso significato, ci renderanno le formule più facili da leggere. Tuttavia da un punto di vista formale è importante ricordare che tali simboli non hanno automaticamente un significato: questo gli viene attribuito successivamente con un'*interpretazione* (Definizione 8.7.3). Per esempio, del simbolo di funzione $\_ + \_$ possiamo dire che è binario e infisso, ma solo quando ne forniremo l'interpretazione potremo dire che esso rappresenta l'addizione tra naturali, come atteso, e non un'altra operazione.

Esempio 8.5.5 (sintassi di termini e formule)**.** *Si consideri l'alfabeto dei naturali* $\mathcal{A}_\mathbb{N}$*. Le seguenti stringhe sono sintatticamente corrette, ovvero possono essere generate dalla grammatica della Definizione 8.5.2:*

- 0*,* 5*,* 21934*,* x*,* n*:* *sono simboli di costante o variabili, quindi sono* termini*, cioè elementi di* $\langle\langle Term \rangle\rangle$*;*
- $3 + 5$*,* $succ(x + 3)$*,* $(3 + 5) \times 0$*:* *sono simboli di funzione applicati a un numero di termini pari alla propria arietà, quindi sono anche essi termini in* $\langle\langle Term \rangle\rangle$*;*
- $x \leq 6$*,* $pari(42)$*,* $primo(3 \times (5 + x))$*,* $4 + x = 4 - y$*:* *sono simboli di predicati applicati a un numero di termini pari alla propria arietà, quindi sono* formule atomiche*, elementi di* $\langle\langle Atom \rangle\rangle$*;*
- $(x \leq 6 \wedge pari(x))$*,* $(\text{primo(x)} \Rightarrow x > 0)$*,* $(\forall x \,.\, x \geq 0)$*,* $(x > 0 \Rightarrow (\exists y \,.\, y < x))$*:* *sono connettivi logici oppure quantificatori applicati a formule, quindi sono* formule*, elementi di* $\langle\langle Pred \rangle\rangle$*.*

Invece le seguenti espressioni non sono sintatticamente corrette, quindi non hanno alcun significato:

- $12+$*:* *"*$+$*" è un simbolo di funzione binario, non può essere applicato a un solo argomento;*
- $(x \leq y) \leq z$*:* *il secondo simbolo di predicato "*$\leq$*" è applicato a una formula atomica e a un termine invece che a due termini; questa espressione, sintatticamente non corretta, viene spesso usata come abbreviazione della formula* $(x \leq y) \wedge (y \leq z)$*;*
- $(\forall x \,.\, x + y)$*:* *l'espressione quantificata non è una formula ma un termine.*

8.6 Formalizzazione di frasi

Con la logica dei predicati siamo in grado di formalizzare molte frasi in maniera più espressiva di quanto sia possibile con il calcolo proposizionale. Vediamo alcuni esempi, basati sugli alfabeti degli interi e delle persone. In questi esempi ci basiamo su una comprensione intuitiva del significato dei quantificatori presentati alla fine della Sezione 8.5, perché presenteremo la loro semantica formale solo nella prossima sezione. Comunque assumiamo che il lettore conosca bene i connettivi logici.

1. *"Esiste un numero che è sia pari che primo"*
   Ovviamente rappresentiamo *"Esiste"* con una quantificazione esistenziale, e sfruttiamo i simboli di predicato $primo(\_)$ e $pari(\_)$. La formula risultante è

$$(\exists x \,.\, (pari(x) \wedge primo(x)))$$

2. *"Tutti gli uomini sono mortali"*
   Estendiamo l'alfabeto $\mathcal{A}_\mathcal{P}$ con il simbolo di predicato *mortale*(_) e il significato intuitivo che *mortale*(p) è vero se e solo se p è mortale. Inoltre rappresentiamo *"Tutti"* con una quantificazione universale. La formula risultante è:
   $$(\forall x \,.\, \mathit{mortale}(x))$$

3. *"Tutti i numeri naturali dispari sono maggiori di zero"*
   In questo caso la quantificazione universale non è su tutti i naturali, ma è ristretta a un sottoinsieme: i naturali dispari. Un modo naturale di formalizzarla è di riconoscere che c'è un'implicazione implicita: *"Per ogni naturale vale che se è dispari allora è maggiore di zero"*. La formula risultante è
   $$(\forall x \,.\, (\mathit{dispari}(x) \Rightarrow (x > 0))$$

4. *"Esiste un numero naturale dispari che è minore o uguale a zero"*
   Anche qui la quantificazione è ristretta ai numeri dispari, ma è una quantificazione esistenziale e possiamo leggerla come *"Esiste un numero che è dispari ed è minore o uguale a zero"*. Quindi può essere formalizzata con
   $$(\exists x \,.\, (\mathit{dispari}(x) \wedge (x \leq 0))) \tag{8.10}$$
   Si noti che l'uso di un'implicazione (come nel caso precedente) sarebbe errata: la formula
   $$(\exists x \,.\, (\mathit{dispari}(x) \Rightarrow (x \leq 0))) \tag{8.11}$$
   ha un significato diverso. Infatti la formula (8.10) è falsa nell'interpretazione standard sui naturali (così come è falso l'enunciato originale), mentre la formula (8.11) è vera, poiché il numero 42 (e qualunque altro numero pari) soddisfa ($\mathit{dispari}(42) \Rightarrow (42 \leq 0)$).

5. *"Ogni persona in questa classe ha un amico che parla inglese o francese"*
   Estendiamo l'alfabeto delle persone con i simboli di predicato unari *questaClasse*, *parlaENG* e *parlaFR* e con il simbolo di predicato binario *amico*, con il seguente significato intuitivo:
   - *questaClasse*(p) = `t` se e solo se la persona p appartiene a questa classe
   - *parlaENG*(p) = `t` se e solo se la persona p parla inglese
   - *parlaFR*(p) = `t` se e solo se la persona p parla francese
   - *amico*(p, q) = `t` se e solo se le persone p e q sono amiche.

   Quella che segue è una possibile formalizzazione della frase:
   $$(\forall x \,.\, (\mathit{questaClasse}(x) \;\Rightarrow\; (\exists y \,.\, (\mathit{amico}(x, y) \wedge (\mathit{parlaENG}(y) \vee \mathit{parlaFR}(y))))))$$
   Abbiamo usato di nuovo un'implicazione per restringere la quantificazione universale a un sottoinsieme delle persone. Che cosa succede se spostiamo la quantificazione esistenziale fuori dall'implicazione?
   $$(\forall x \,.\, (\exists y \,.\, (\mathit{questaClasse}(x) \;\Rightarrow\; (\mathit{amico}(x, y) \wedge (\mathit{parlaENG}(y) \vee \mathit{parlaFR}(y))))))$$
   Rileggendo la formula con attenzione si vede che il significato è rimasto invariato. Il prossimo esempio mostra che questo non è vero se si scambiano due quantificatori diversi.

6. *"Ogni persona ha un padre"*
   Usando il simbolo di predicato binario *padre*, formalizziamo questa semplice frase nel modo seguente:
   $$(\forall x \,.\, (\exists y \,.\, \mathit{padre}(y, x)))$$
   Se rileggiamo la formula abbiamo *"Per ogni persona ne esiste un'altra che è suo padre"*. Cosa succede se scambiamo i due quantificatori? La formula risultante è:
   $$(\exists y \,.\, (\forall x \,.\, \mathit{padre}(y, x)))$$
   che, riletta in italiano, dice *"Esiste una persona che è il padre di ogni altra persona"*. È evidente che il significato delle due formule è completamente diverso, e in particolare la prima è vera nell'interpretazione standard delle persone mentre la seconda è falsa.

7. *"Ogni persona felice ha un solo amico del cuore"*
   Estendiamo l'alfabeto delle persone con i simboli di predicato $felice(\_)$ e $amiciDelCuore(\_,\_)$ con l'ovvia interpretazione. Come primo tentativo, proviamo a formalizzare la frase così:

$$(\forall x\,.\,(\exists y\,.\,(felice(x) \;\Rightarrow\; amiciDelCuore(x,y))))$$

   Tuttavia questa formula non coglie il concetto di *unicità*! La seguente formula invece lo cattura in modo corretto:

$$(\forall x\,.\,(\exists y\,.\,(felice(x) \;\Rightarrow\; (amiciDelCuore(x,y) \wedge (\forall z\,.\,(amiciDelCuore(x,z) \Rightarrow y = z))))))$$

8.7 Interpretazioni e semantica delle formule predicative

Vediamo ora come determinare la semantica, cioè il valore di verità, di formule predicative su un certo alfabeto del primo ordine. La prima osservazione è che non tutte tali formule possono avere una semantica ben definita. Per esempio, interpretando in modo ovvio il simbolo di predicato $\leq$ come la relazione "minore o uguale" sui naturali, la formula $x \leq 0$ non ha un valore di verità ben determinato: essa vale `t` se sostituiamo x con 0, e `f` se sostituiamo x con un qualunque altro naturale. Invece la semantica delle formule $(\exists x\,.\,x \leq 0)$ e $(\forall x\,.\,x \leq 0)$ è ben definita, perché le quantificazioni $\forall x$ e $\exists x$ indicano come deve essere sostituita x per determinare il valore di verità: $(\exists x\,.\,x \leq 0)$ vale `t` perché c'è almeno un naturale minore o uguale a 0, mentre $(\forall x\,.\,x \leq 0)$ vale `f` perché non tutti i naturali sono minori o uguali a 0.

Definizione 8.7.1 (campo di azione, variabili legate e libere, formule aperte e chiuse)**.** *Il* CAMPO D'AZIONE *(o* PORTATA *o* SCOPE*) della quantificazione $\forall x$ nella formula $(\forall x\,.\,P)$ è l'intera formula P, e analogamente per la quantificazione esistenziale. Una specifica occorrenza di una variabile x in una formula P è detta* LEGATA *se compare nella portata di una quantificazione $\forall x$ o $\exists x$, altrimenti è detta* LIBERA*. Una formula P è* CHIUSA *se ogni occorrenza di variabile in essa è legata, altrimenti è aperta. Invece una formula come $x \leq 0$ viene detta* APERTA *perché contiene una variabile, la x, che compare nella formula senza essere legata a un corrispondente quantificatore. Una variabile che non è legata a un quantificatore viene detta* LIBERA*.*

Esempio 8.7.2 (campo d'azione, formule aperte e chiuse)**.** *Nella formula seguente le parentesi graffe mostrano il campo d'azione dei due quantificatori. La formula è chiusa perché ogni occorrenza di variabile (le due x e la y) compare nel campo d'azione di un corrispondente quantificatore.*

$$(\forall x\,.\,\overbrace{x \leq 0 \;\Rightarrow\; (\forall y\,.\,\underbrace{y \geq x}_{\forall y})}^{\forall x})$$

Invece la formula che segue è aperta, perché la seconda x non è nel campo di azione di un $\forall x$ o $\exists x$.

$$(\forall x\,.\,\underbrace{x > 0}_{\forall x}) \;\Rightarrow\; (\exists y\,.\,\underbrace{y < \overbrace{x}^{libera}}_{\exists y})$$

Data una formula su un certo alfabeto $\mathcal{A}$, per associarle un valore di verità dobbiamo fissare il significato dei simboli di costante, di funzione, di predicato e delle variabili che compaiono in essa, esattamente come per una formula proposizionale dovevamo fissare un valore di verità per ogni simbolo proposizionale. Per far questo fisseremo un *dominio di interpretazione*, cioè un insieme, e un'interpretazione dei simboli di costante come elementi del dominio, e dei simboli di funzione e di predicato come opportune funzioni o relazioni sul dominio. Le variabili verranno trattate diversamente, perchè il loro assegnamento a elementi del dominio può cambiare dinamicamente durante la valutazione della semantica di una formula, a differenza dell'interpretazione degli altri simboli che rimana fissa.

Quindi il concetto di interpretazione, già visto nel caso più semplice del calcolo proposizionale, viene arricchito nel modo seguente.

Definizione 8.7.3 (interpretazione per alfabeto del primo ordine)**.** *Un'*INTERPRETAZIONE *$\mathcal{I}$ per un alfabeto del primo ordine $\mathcal{A} = (\mathcal{C}, \mathcal{F}, \mathcal{P}, \mathcal{V})$ è una coppia $\mathcal{I} = (\mathcal{D}, \alpha)$ dove $\mathcal{D}$ è il* DOMINIO *(*DI INTERPRETAZIONE*) (un insieme di valori), mentre α è un'*ASSOCIAZIONE *costituita da tre componenti $\alpha = \langle \alpha_{\mathcal{C}}, \alpha_{\mathcal{F}}, \alpha_{\mathcal{P}} \rangle$, dove:*

- *$\alpha_{\mathcal{C}} : \mathcal{C} \to \mathcal{D}$ associa a ogni simbolo di costante in $\mathcal{C}$ un elemento del dominio;*
- *$\alpha_{\mathcal{F}} = \{\alpha_{\mathcal{F}_n}\}_{n \in \mathbb{N}^+}$ è una famiglia di funzioni, in cui $\alpha_{\mathcal{F}_n}$ associa a ogni simbolo di funzione $f \in \mathcal{F}_n$ una funzione $\alpha_{\mathcal{F}_n}(f) : \mathcal{D}^n \to \mathcal{D}$, cioè una funzione n-aria su $\mathcal{D}$.*
- *$\alpha_{\mathcal{P}} = \{\alpha_{\mathcal{P}_n}\}_{n \in \mathbb{N}}$ è una famiglia di funzioni, dove*
  - *$\alpha_{\mathcal{P}_0}$ associa a ogni simbolo di predicato di arietà zero un valore booleano, cioè $\alpha_{\mathcal{P}_0} : \mathcal{P}_0 \to Bool$.*
  - *per ogni $n \in \mathbb{N}^+$ $\alpha_{\mathcal{P}_n}$ associa a ogni simbolo di predicato $A \in \mathcal{P}_n$ un sottoinsieme $\alpha_{\mathcal{P}_n}(A) \subseteq \mathcal{D}^n$.*

Una volta fissata un'interpretazione per un alfabeto $\mathcal{A}$ possiamo utilizzarla per associare un valore di verità (la semantica) a ogni formula chiusa scritta con simboli in $\mathcal{A}$.

Ma prima di vedere come si fa, introduciamo le interpretazioni per gli alfabeti dell'Esempio 8.5.4, in modo da poter poi presentare degli esempi. Nel caso degli alfabeti dei naturali, delle persone e degli insiemi queste interpretazioni associano ai vari simboli il significato standard che ci aspettiamo, ma sono comuque necessarie per collegare le entità sintattiche (i simboli degli alfabeti) con la corrispondente semantica definita sul dominio di interpretazione.

Definizione 8.7.4 (esempi di interpretazioni)**.**

1. *Per l'alfabeto $\mathcal{A}_{\mathcal{T}} = (\mathcal{C}_{\mathcal{T}}, \mathcal{F}_{\mathcal{T}}, \mathcal{P}_{\mathcal{T}}, \mathcal{V}_{\mathcal{T}})$ dell'Esempio 8.5.4 introduciamo l'interpretazione $\mathcal{I}_{\mathcal{T}} = (\mathcal{D}_{\mathcal{T}}, \alpha^{\mathcal{T}})$ dove*

- *$\mathcal{D}_{\mathcal{T}} = \{a, b, c\}$, quindi il dominio è finito;*
- *$\alpha^{\mathcal{T}} = \langle \alpha^{\mathcal{T}}_{\mathcal{C}}, \alpha^{\mathcal{T}}_{\mathcal{F}}, \alpha^{\mathcal{T}}_{\mathcal{P}} \rangle$;*
- *$\alpha^{\mathcal{T}}_{\mathcal{C}}(k) = a$;*
- *$\alpha^{\mathcal{T}}_{\mathcal{F}}(f) = \{a \mapsto b, b \mapsto c, c \mapsto a\}$;*
- *$\alpha^{\mathcal{T}}_{\mathcal{P}}(A) = \{a, b\}$;*
- *$\alpha^{\mathcal{T}}_{\mathcal{P}}(B) = \{(a, a), (c, a), (c, b), (b, c)\}$.*

Si noti che coerentemente con la convenzione adottata nell'Esempio 8.5.4, consideriamo $\alpha^{\mathcal{T}}_{\mathcal{F}}$ e $\alpha^{\mathcal{T}}_{\mathcal{P}}$ come delle funzioni e non come famiglie di funzioni. La loro definizione per ogni simbolo è consistente con la corrispondente arietà, che non è scritta esplicitamente ma si può ricavare dalla definizione dell'alfabeto. Infatti $\alpha^{\mathcal{T}}_{\mathcal{F}}(f) : \mathcal{D}_{\mathcal{T}} \to \mathcal{D}_{\mathcal{T}}$ è una funzione unaria, $\alpha^{\mathcal{T}}_{\mathcal{F}}(A)$ è un sottoinsieme di $\mathcal{D}_{\mathcal{T}}$ e $\alpha^{\mathcal{T}}_{\mathcal{F}}(B)$ è un sottoinsieme di $\mathcal{D}^2_{\mathcal{T}}$.

2. *L'*interpretazione (standard) *per l'alfabeto dei naturali $\mathcal{A}_{\mathbb{N}} = (\mathcal{C}_{\mathbb{N}}, \mathcal{F}_{\mathbb{N}}, \mathcal{P}_{\mathbb{N}}, \mathcal{V}_{\mathbb{N}})$ è la coppia $\mathcal{I}_{\mathbb{N}} = (\mathcal{D}_{\mathbb{N}}, \alpha^{\mathbb{N}})$ dove il dominio $\mathcal{D}_{\mathbb{N}}$ è costituito dall'insieme dei naturali $\mathbb{N}$, mentre $\alpha^{\mathbb{N}} = \langle \alpha^{\mathbb{N}}_{\mathcal{C}}, \alpha^{\mathbb{N}}_{\mathcal{F}}, \alpha^{\mathbb{N}}_{\mathcal{P}} \rangle$ dove*

- *$\alpha^{\mathbb{N}}_{\mathcal{C}} : \mathcal{C}_{\mathbb{N}} \to \mathbb{N}$ associa a ogni simbolo di costante $c \in \mathcal{C}^{\mathbb{N}} (= \mathbb{N})$ se stessa;*
- *$\alpha^{\mathbb{N}}_{\mathcal{F}}$ associa a ogni simbolo di funzione in $\mathcal{F}_{\mathbb{N}}$ la corrispondente funzione sui naturali. Per esempio, $\alpha^{\mathbb{N}}_{\mathcal{F}}(succ)$ è la funzione* successore *che mappa ogni numero x su $x + 1$, quindi per esempio $\alpha^{\mathbb{N}}_{\mathcal{F}}(succ)(5) = 6$; invece $\alpha^{\mathbb{N}}_{\mathcal{F}}(+)$ è l'addizione su naturali, quindi per esempio $\alpha^{\mathbb{N}}_{\mathcal{F}}(+)(3, 5) = 8$, e così via.*
- *$\alpha^{\mathbb{N}}_{\mathcal{P}}$ associa a ogni simbolo di predicato in $\mathcal{P}_{\mathbb{N}}$ di arietà n la corrispondente proprietà sui naturali, cioè l'insieme di n-uple di naturali che la soddisfano. Per esempio, per tutti i naturali $n, m \in \mathbb{N}$, $(n, m) \in \alpha^{\mathbb{N}}_{\mathcal{P}}(\leq)$ se e solo se n è minore o uguale a m; mentre $n \in \alpha^{\mathbb{N}}_{\mathcal{P}}(primo)$ se e solo se n è un numero primo.*

2. *L'interpretazione (standard) $\mathcal{I}_\mathcal{P}$ dell'alfabeto delle persone $\mathcal{A}_\mathcal{P} = (\mathcal{C}_\mathcal{P}, \mathcal{F}_\mathcal{P}, \mathcal{P}_\mathcal{P}, \mathcal{V}_\mathcal{P})$ ha come dominio appunto l'insieme di tutte le persone. I simbili di costante, di funzione e di predicato sono associati nel modo ovvio a persone, a funzioni o a relazioni definite su persone. Per esempio, $\alpha^\mathcal{P}_\mathcal{C}(Aldo)$ è una determinata persona di nome Aldo; $\alpha^\mathcal{P}_\mathcal{C}(Joe\ Biden)$ è un politico americano, attualmente Presidente Eletto degli USA; $\alpha^\mathcal{P}_\mathcal{F}(padre)$ è la funzione che associa ogni persona al suo padre naturale; e $(p, q) \in \alpha^\mathcal{P}_\mathcal{P}(fratelli)$ se e solo se le persone p e q sono fratelli.*

3. *Infine l'interpretazione (standard) dell'alfabeto degli insiemi ha come dominio un insieme che ha come elementi sia gli insiemi di nostro interesse (come $\mathbb{N}$, $\mathbb{Z}$, ecc.) che i loro elementi. I simboli di costante, di funzione e di predicato elencati nell'Esempio 8.5.4 sono associati in modo ovvio al corrispodente significato rispettando la usuale notazione matematica.*

Semantica di termini e di formule predicative

Come nel caso del calcolo proposizionale, la semantica di una formula predicativa per una data interpretazione può essere definita in modo induttivo. Dobbiamo però considerare separatamente la categoria sintattica dei termini, visto che la loro semantica non è un booleano ma un elemento del dominio. Inoltre sia formule che termini possono in generale contenere variabili libere, per cui la semantica è parametrica rispetto a un *assegnamento* che associa a ogni variabile un elemento del dominio.

Definizione 8.7.5 (assegnamento)**.** *Dato un alfabeto $\mathcal{A}$ con insieme di variabili $\mathcal{V}$ e un'interpretazione $\mathcal{I} = \langle \mathcal{D}, \alpha \rangle$, un* ASSEGNAMENTO *è una funzione parziale $r : \mathcal{V} \to \mathcal{D}$.*[10]

Dati un assegnamento $r : \mathcal{V} \to \mathcal{D}$, una variable $v \in \mathcal{V}$ e un elemento $d \in D$, con $r[x \mapsto d]$ denotiamo l'assegnamento così definito:

$$r[x \mapsto d](y) = \begin{cases} d & \text{se } y = x \\ r(y) & \text{altrimenti.} \end{cases}$$

Se $\varnothing \colon \mathcal{V} \to \mathcal{D}$ è la relazione vuota, scriveremo semplicemente $[x \mapsto d]$ per $\varnothing[x \mapsto d]$

Definizione 8.7.6 (semantica dei termini)**.** *Data una interpretazione $\mathcal{I} = \langle \mathcal{D}, \alpha \rangle$ e un assegnamento $r : \mathcal{V} \to \mathcal{D}$, il* VALORE RISPETTO AD $\mathcal{I}$ ED r *di un termine è dato dalla funzione parziale*[11]

$$[\![\_]\!]^r_\mathcal{I} \colon \mathbf{Term} \to \mathcal{D}$$

definita per induzione strutturale come segue:

1. $[\![x]\!]^r_\mathcal{I} = r(x)$ *per ogni $x \in \mathcal{V}$ (il valore di una variable è determinato dalla sostituzione);*
2. $[\![c]\!]^r_\mathcal{I} = \alpha_\mathcal{C}(c)$ *per ogni $c \in \mathcal{C}$ (il valore di un simbolo di costante è determinato da $\alpha_\mathcal{C}$);*
3. *Se f è un simbolo di funzione di arietà n e $[\![t_1]\!]^r_\mathcal{I} = d_1, \ldots, [\![t_n]\!]^r_\mathcal{I} = d_n$, allora*

$$[\![f(t_1, \ldots, t_n)]\!]^r_\mathcal{I} = \alpha_\mathcal{F}(f)(d_1, \ldots, d_n)$$

(si valutano ricorsivamente i sottotermini e si applica la funzione $\alpha_\mathcal{F}(f)$).

Esempio 8.7.7 (da termini a elementi del dominio)**.**

1. *Consideriamo l'alfabeto $\mathcal{A}_\mathcal{T}$ dell'Esempio 8.5.4 e l'interpretazione $\mathcal{I}_\mathcal{T} = (\mathcal{D}_\mathcal{T}, \alpha^\mathcal{T})$ della Definizione 8.7.4. Esempi di termini sono la variabile x, la costante k, $f(k)$, e $f(f(x))$. Dato una assegnamento $r : \mathcal{V}_\mathcal{T} \to \mathcal{D}_\mathcal{T} = \{a, b, c\}$ tale che $r(x) = b$, i corrispondenti valori sono:*

- $[\![x]\!]^r_{\mathcal{I}_\mathcal{T}} = r(x) = b$ *(Definizione 8.7.6, clausola 1);*
- $[\![k]\!]^r_{\mathcal{I}_\mathcal{T}} = \alpha^\mathcal{T}_\mathcal{C}(k) = a$ *(Definizione 8.7.6, clausola 2);*

[10] Ricordiamo che diversamente da una funzione, una *funzione parziale* è una relazione univalente ma non necessariamente totale (Definizione 2.5.23).

[11] Poiché r è una funzione parziale, $[\![t]\!]^r_\mathcal{I}$ potrebbe non essere definito per un termine t. Per come useremo gli assegnamenti questo non può mai succedere, quindi per semplicità ignoriamo questa possibilità e consideriamo la funzione totale.

- $[\![f(k)]\!]^r_{\mathcal{I}_\mathcal{T}} = \alpha^\mathcal{T}_\mathcal{F}(f)([\![k]\!]^r_{\mathcal{I}_\mathcal{T}})$ *(Def. 8.7.6, cl. 3)*
  $= \alpha^\mathcal{T}_\mathcal{F}(f)(a)$ *(Def. 8.7.6, cl. 2)*
  $= b$ *(Def. 8.7.4, punto 1)*

- $[\![f(f(x))]\!]^r_{\mathcal{I}_\mathcal{T}} = \alpha^\mathcal{T}_\mathcal{F}(f)([\![f(x)]\!]^r_{\mathcal{I}_\mathcal{T}})$ *(Def. 8.7.6, cl. 3)*
  $= \alpha^\mathcal{T}_\mathcal{F}(f)(\alpha^\mathcal{T}_\mathcal{F}(f)([\![x]\!]^r_{\mathcal{I}_\mathcal{T}}))$ *(Def. 8.7.6, cl. 3)*
  $= \alpha^\mathcal{T}_\mathcal{F}(f)(\alpha^\mathcal{T}_\mathcal{F}(f)(b))$ *(Def. 8.7.6, cl. 2)*
  $= \alpha^\mathcal{T}_\mathcal{F}(f)(c)$ *(Def. 8.7.4, punto 1)*
  $= a$ *(Def. 8.7.4, punto 1)*

2. *Consideriamo ora l'alfabeto dei naturali* $\mathcal{A}_\mathbb{N}$. *Esempi di termini sono* 5, $7{+}x$, *e* $succ((y{\times}4)/5)$. *Dato un assegnamento* $r : \mathcal{V}_\mathbb{N} \to \mathbb{N}$ *tale che* $r(x) = 8$ *e* $r(y) = 3$, *i corrispondenti valori del dominio rispetto all'interpretazione standard* $\mathcal{I}_\mathbb{N}$ *e a* r *sono:*

- $[\![5]\!]^r_{\mathcal{I}_\mathbb{N}} = \alpha^\mathbb{N}_\mathcal{C}(5) = 5$;
- $[\![7 + x]\!]^r_{\mathcal{I}_\mathbb{N}} = \alpha^\mathbb{N}_\mathcal{F}(+)([\![7]\!]^r_{\mathcal{I}_\mathbb{N}}, [\![x]\!]^r_{\mathcal{I}_\mathbb{N}}) = \alpha^\mathbb{N}_\mathcal{F}(+)(7, r(x)) = 7 + 8 = 15$;

- $[\![succ((y \times 4)/5)]\!]^r_{\mathcal{I}_\mathbb{N}} = \alpha^\mathbb{N}_\mathcal{F}(succ)([\![(y \times 4)/5]\!]^r_{\mathcal{I}_\mathbb{N}})$ *(Def. 8.7.6, clausola 3)*
  $= \alpha^\mathbb{N}_\mathcal{F}(succ)(([\![y]\!]^r_{\mathcal{I}_\mathbb{N}} \times [\![4]\!]^r_{\mathcal{I}_\mathbb{N}})/[\![5]\!]^r_{\mathcal{I}_\mathbb{N}})$ *(Def. 8.7.6, cl. 3 e Def. 8.7.4, cl. 1)*
  $= \alpha^\mathbb{N}_\mathcal{F}(succ)((r(y) \times 4)/5)$ *(Def. 8.7.6, clausole 1 e 2)*
  $= \alpha^\mathbb{N}_\mathcal{F}(succ)((3 \times 4)/5)$ *($r(y) = 3$ per ipotesi)*
  $= \alpha^\mathbb{N}_\mathcal{F}(succ)(2)$ *(Calcolo ("/" è divisione intera))*
  $= 2 + 1 = 3.$ *(Def. 8.7.4, punto 2)*

La semantica delle formule è definita anche essa per induzione strutturale in modo parametrico rispetto a una interpretazione e a un assegnamento. Se una formula è aperta, essa può assumere valori di verità diversi nella stessa interpretazione a seconda del valore che l'assegnamento associa alle variabili libere (si pensi semplicemente a $primo(x)$ valutato nell'interpretazione $\mathcal{I}_\mathbb{N}$ con $r(x) = 3$ o $r(x) = 6$). Invece se una formula è chiusa, il suo valore in una interpretazione è univocamente determinato. Presentiamo prima la semantica delle formule generali e poi quella delle formule chiuse.

Definizione 8.7.8 (Semantica della logica dei predicati (anche formule aperte))**.** *Data una interpretazione* $\mathcal{I} = (\mathcal{D}, \alpha)$ *per un alfabeto* $\mathcal{A}$ *e una sostituzione* $r : \mathcal{V} \to \mathcal{D}$, *il* VALORE RISPETTO AD $\mathcal{I}$ ED r *delle formule predicative è dato dalla funzione*

$$[\![\_]\!]^r_\mathcal{I} : \mathbf{Pred} \to \mathit{Bool}$$

definita sulle formule per induzione strutturale come segue:

1. $[\![\mathsf{T}]\!]^r_\mathcal{I} = \mathtt{t}$ *e* $[\![\mathsf{F}]\!]^r_\mathcal{I} = \mathtt{f}$;
2. *Per ogni* $A \in \mathcal{P}_0$, $[\![A]\!]^r_\mathcal{I} = \alpha_{\mathcal{P}_0}(A)$.
3. *Per ogni* $A \in \mathcal{P}_n$ *di arietà* $n > 0$,

$$[\![A(t_1, \ldots, t_n)]\!]^r_\mathcal{I} = \begin{cases} \mathtt{t} & \text{se } ([\![t_1]\!]^r_\mathcal{I}, \ldots, [\![t_n]\!]^r_\mathcal{I}) \in \alpha_{\mathcal{P}_n}(A) \\ \mathtt{f} & \text{altrimenti.} \end{cases}$$

4. $[\![(P)]\!]^r_\mathcal{I} = ([\![P]\!]^r_\mathcal{I})$ *per ogni* $P \in \mathbf{Pred}$;
5. $[\![\neg P]\!]^r_\mathcal{I} = \neg[\![P]\!]^r_\mathcal{I}$ *per ogni formula atomica* P;
6. $[\![P \ op \ Q]\!]^r_\mathcal{I} = [\![P]\!]^r_\mathcal{I} \ op \ [\![Q]\!]^r_\mathcal{I}$ *per ogni connettivo* $op \in \{\wedge, \vee, \Rightarrow, \Leftarrow, \Leftrightarrow\}$ *e per ogni* $P, Q \in \mathbf{Pred}$.

7. $[\![(\forall x\,.\,P)]\!]^r_{\mathcal{I}} = \mathtt{t}$ *se e solo se* $[\![P]\!]^{r[x\mapsto d]}_{\mathcal{I}} = \mathtt{t}$ $\boxed{\textit{per ogni}}$ $d \in \mathcal{D}$.

8. $[\![(\exists x\,.\,P)]\!]^r_{\mathcal{I}} = \mathtt{t}$ *se e solo se* $[\![P]\!]^{r[x\mapsto d]}_{\mathcal{I}} = \mathtt{t}$ $\boxed{\textit{per almeno un}}$ $d \in \mathcal{D}$.

Definizione 8.7.9 (Semantica della logica dei predicati (formule chiuse))**.** *Data una interpretazione* $\mathcal{I} = (\mathcal{D}, \alpha)$ *per un alfabeto* $\mathcal{A} = (\mathcal{C}, \mathcal{F}, \mathcal{P}, \mathcal{V})$ *e una formula* chiusa P*, usando la Definizione 8.7.8 il valore di* P *rispetto ad* $\mathcal{I}$ *è definito come*

$$[\![P]\!]_{\mathcal{I}} = [\![P]\!]^{\varnothing}_{\mathcal{I}}$$

dove $\varnothing : \mathcal{V} \to \mathcal{D}$ *è la relazione vuota.*

Vediamo qualche esempio di valutazione della semantica di formule chiuse senza e con quantificatori.

Esempio 8.7.10 (semantica di formule chiuse senza quantificatori)**.** *Consideriamo l'interpretazione standard dei naturali* $\mathcal{I}_{\mathbb{N}}$*, e vediamo la semantica di alcune formule chiuse senza quantificatori (e quindi senza variabili).*

- *Mostriamo che* $[\![0 \leq 1]\!]_{\mathcal{I}_{\mathbb{N}}} = \mathtt{t}$*:*

$$\begin{array}{lcll} [\![0 \leq 1]\!]_{\mathcal{I}_{\mathbb{N}}} & = & [\![0 \leq 1]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}} & \textit{(Def. 8.7.9)} \\ & = & ([\![0]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}}, [\![1]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}}) \in \alpha^{\mathcal{I}_{\mathbb{N}}}_{\mathcal{P}}(\leq) & \textit{(Def. 8.7.8, clausola 3)} \\ & = & (0,1) \in \alpha^{\mathcal{I}_{\mathbb{N}}}_{\mathcal{P}}(\leq) & \textit{(Def. 8.7.6, clausola 2)} \\ & = & 0 \leq 1 & \textit{(Def. 8.7.4, punto 2)} \\ & = & \mathtt{t} & \end{array}$$

- *Mostriamo che* $[\![pari(2) \Rightarrow primo(2)]\!]_{\mathcal{I}_{\mathbb{N}}} = \mathtt{t}$*:*

$$\begin{array}{lcll} & & [\![pari(2) \Rightarrow primo(2)]\!]_{\mathcal{I}_{\mathbb{N}}} & \\ & = & [\![pari(2) \Rightarrow primo(2)]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}} & \textit{(Def. 8.7.9)} \\ & = & [\![pari(2)]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}} \Rightarrow [\![primo(2)]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}} & \textit{(Def. 8.7.8, clausola 6)} \\ & = & ([\![2]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}} \in \alpha^{\mathcal{I}_{\mathbb{N}}}_{\mathcal{P}}(pari)) \Rightarrow ([\![2]\!]^{\varnothing}_{\mathcal{I}_{\mathbb{N}}} \in \alpha^{\mathcal{I}_{\mathbb{N}}}_{\mathcal{P}}(primo)) & \textit{(Def. 8.7.8, clausola 3)} \\ & = & (2 \in \alpha^{\mathcal{I}_{\mathbb{N}}}_{\mathcal{P}}(pari)) \Rightarrow (2 \in \alpha^{\mathcal{I}_{\mathbb{N}}}_{\mathcal{P}}(primo)) & \textit{(Def. 8.7.6, clausola 2)} \\ & = & \mathtt{t} \Rightarrow \mathtt{t} & \textit{(Def. 8.1.6)} \\ & = & \mathtt{t} & \end{array}$$

Esempio 8.7.11 (semantica di formule chiuse con quantificatori)**.** *Consideriamo l'alfabeto* $\mathcal{A}_{\mathcal{T}}$ *dell'Esempio 8.5.4 e l'interpretazione* $\mathcal{I}_{\mathcal{T}} = (\mathcal{D}_{\mathcal{T}}, \alpha^{\mathcal{T}})$ *della Definizione 8.7.4. Valutiamo la semantica della formula*

$$\Phi = (\forall z\,.\,A(f(z))) \vee (\forall y\,.\,(\exists x\,.\,B(x,y) \wedge A(x)))$$

In questo caso particolare, poichè il dominio $\mathcal{D}_{\mathcal{T}} = \{a, b, c\}$ *è finito, possiamo gestire i quantificatori molto facilmente, valutando il quantificatore universale come una congiunzione e quello esistenziale come una disgiunzione. Mostriamo che la formula è falsa nell'interpretazione data.*

Per prima cosa valutiamo $[\![A(f(z))]\!]^{[z\mapsto b]}_{\mathcal{I}_{\mathcal{T}}}$*. Abbiamo:*

$$\begin{array}{lcll} & & [\![A(f(z))]\!]^{[z\mapsto b]}_{\mathcal{I}_{\mathcal{T}}} & (8.12) \\ & = & [\![f(z)]\!]^{[z\mapsto b]}_{\mathcal{I}_{\mathcal{T}}} \in \alpha^{\mathcal{T}}_{\mathcal{P}}(A) & \textit{(Def. 8.7.8, clausola 3)} \\ & = & \alpha^{\mathcal{T}}_{\mathcal{F}}(f)([\![z]\!]^{[z\mapsto b]}_{\mathcal{I}_{\mathcal{T}}}) \in \alpha^{\mathcal{T}}_{\mathcal{P}}(A) & \textit{(Def. 8.7.6, clausola 3)} \\ & = & \alpha^{\mathcal{T}}_{\mathcal{F}}(f)(b) \in \alpha^{\mathcal{T}}_{\mathcal{P}}(A) & \textit{(Def. 8.7.6, clausola 1)} \\ & = & c \in \{a, b\} \quad = \quad \mathtt{f} & \textit{(Def. 8.7.4, punto 1)} \end{array}$$

Valutiamo ora la formula Φ*:*

$$
\begin{aligned}
&[\![(\forall z\,.\,A(f(z))) \vee (\forall y\,.\,(\exists x\,.\,B(x,y) \wedge A(x)))]\!]_{\mathcal{I}_\mathcal{T}} \\
&\quad = [\![(\forall z\,.\,A(f(z))) \vee (\forall y\,.\,(\exists x\,.\,B(x,y) \wedge A(x)))]\!]^{\varnothing}_{\mathcal{I}_\mathcal{T}} && \textit{(Def. 8.7.9)} \\
&\quad = [\![(\forall z\,.\,A(f(z)))]\!]^{\varnothing}_{\mathcal{I}_\mathcal{T}} \vee [\![(\forall y\,.\,(\exists x\,.\,B(x,y) \wedge A(x)))]\!]^{\varnothing}_{\mathcal{I}_\mathcal{T}} && \textit{(Def. 8.7.8, clausola 6)} \\
&\quad = \Big([\![A(f(z))]\!]^{[z\mapsto a]}_{\mathcal{I}_\mathcal{T}} \wedge \underline{[\![A(f(z))]\!]^{[z\mapsto b]}_{\mathcal{I}_\mathcal{T}}} \wedge [\![A(f(z))]\!]^{[z\mapsto c]}_{\mathcal{I}_\mathcal{T}}\Big) && \textit{(Def. 8.7.8, cl. 7, dom. finito)} \\
&\qquad \vee [\![(\forall y\,.\,(\exists x\,.\,B(x,y) \wedge A(x)))]\!]^{\varnothing}_{\mathcal{I}_\mathcal{T}} && \textit{(Formula sottolineata } \mathtt{f} \textit{ per } (8.12)\textit{)} \\
&\quad = [\![(\forall y\,.\,(\exists x\,.\,B(x,y) \wedge A(x)))]\!]^{\varnothing}_{\mathcal{I}_\mathcal{T}} && \textit{((assorbimento), (unità))} \\
&\quad = [\![(\exists x\,.\,B(x,y) \wedge A(x))]\!]^{[y\mapsto a]}_{\mathcal{I}_\mathcal{T}} \wedge \underline{[\![(\exists x\,.\,B(x,y) \wedge A(x))]\!]^{[y\mapsto b]}_{\mathcal{I}_\mathcal{T}}} && \textit{(Def. 8.7.8, cl. 7, dom. finito)} \\
&\qquad \wedge [\![(\exists x\,.\,B(x,y) \wedge A(x))]\!]^{[y\mapsto c]}_{\mathcal{I}_\mathcal{T}}
\end{aligned}
$$

Concludiamo mostrando che l'ultima formula sottolineata è anche essa $\mathtt{f}$ *nell'interpretazione data, e quindi lo è anche la formula iniziale* Φ*.*

$$
\begin{aligned}
&[\![(\exists x\,.\,B(x,y) \wedge A(x))]\!]^{[y\mapsto b]}_{\mathcal{I}_\mathcal{T}} \quad = \quad \textit{(Def. 8.7.8, cl. 8, dominio finito)} \\
&\quad = [\![B(x,y) \wedge A(x)]\!]^{[y\mapsto b][x\mapsto a]}_{\mathcal{I}_\mathcal{T}} \;\vee\; [\![B(x,y) \wedge A(x)]\!]^{[y\mapsto b][x\mapsto b]}_{\mathcal{I}_\mathcal{T}} \;\vee\; [\![B(x,y) \wedge A(x)]\!]^{[y\mapsto b][x\mapsto c]}_{\mathcal{I}_\mathcal{T}} \\
&\quad = \big((a,b) \in \alpha^{\mathcal{T}}_{\mathcal{P}}(B) \wedge a \in \alpha^{\mathcal{T}}_{\mathcal{P}}(A)\big) \;\vee\; \big((b,b) \in \alpha^{\mathcal{T}}_{\mathcal{P}}(B) \wedge b \in \alpha^{\mathcal{T}}_{\mathcal{P}}(A)\big) \;\vee\; \big((c,b) \in \alpha^{\mathcal{T}}_{\mathcal{P}}(B) \wedge \\
&\qquad \wedge c \in \alpha^{\mathcal{T}}_{\mathcal{P}}(A)\big) \qquad \textit{(Def. 8.7.4: } \alpha^{\mathcal{T}}_{\mathcal{P}}(A) = \{a,b\},\ \alpha^{\mathcal{T}}_{\mathcal{P}}(B) = \{(a,a),(c,a),(c,b),(b,c)\}\textit{)} \\
&\quad = \big(\mathtt{f} \wedge \mathtt{t}\big) \;\vee\; \big(\mathtt{f} \wedge \mathtt{t}\big) \;\vee\; \big(\mathtt{t} \wedge \mathtt{f}\big) \quad = \quad \mathtt{f}
\end{aligned}
$$

Consideriamo ora l'interpretazione dei naturali, e valutiamo la semantica di alcune formule chiuse con quantificatori.

1. $[\![(\forall x\,.\,(0 \leq 1))]\!]_{\mathcal{I}_\mathbb{N}}$

$$
\begin{aligned}
&= [\![(\forall x\,.\,(0 \leq 1))]\!]^{\varnothing}_{\mathcal{I}_\mathbb{N}} && \textit{(Def. 8.7.9)} \\
&= [\![0 \leq 1]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}},\ \textit{per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.8, cl. 7)} \\
&= ([\![0]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}}, [\![1]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}}) \in \alpha^{\mathcal{I}_\mathbb{N}}_{\mathcal{P}}(\leq),\ \textit{per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.8, cl. 3)} \\
&= (0,1) \in \alpha^{\mathcal{I}_\mathbb{N}}_{\mathcal{P}}(\leq),\ \textit{per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.6, cl. 2)} \\
&= \mathtt{t} \qquad \textit{Non dipende da } d \textit{ perché } x \textit{ non occorre nella formula}
\end{aligned}
$$

2. $[\![(\forall x\,.\,(x \leq 1))]\!]_{\mathcal{I}_\mathbb{N}}$

$$
\begin{aligned}
&= [\![(\forall x\,.\,(x \leq 1))]\!]^{\varnothing}_{\mathcal{I}_\mathbb{N}} && \textit{(Def. 8.7.9)} \\
&= [\![x \leq 1]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}},\ \textit{per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.8, cl. 7)} \\
&= ([\![x]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}}, [\![1]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}}) \in \alpha^{\mathcal{I}_\mathbb{N}}_{\mathcal{P}}(\leq),\ \textit{per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.8, cl. 3)} \\
&= (d,1) \in \alpha^{\mathcal{I}_\mathbb{N}}_{\mathcal{P}}(\leq),\ \textit{per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.6, cl. 1 e 2)} \\
&= d \leq 1 \textit{ per ogni } d \in \mathbb{N} && \textit{(Def. 8.7.4, punto 2)}
\end{aligned}
$$

Quindi la formula è falsa, scegliendo per esempio $d = 2$*.*

I prossimi esempi li presentiamo in modo più informale, ma comunque rigoroso.

3. $[\![(\exists x\,.\,(x \leq 1))]\!]_{\mathcal{I}_\mathbb{N}}$*: per la clausola 8 della Def. 8.7.8, è vera se e solo se* $[\![(x \leq 1)]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}} = \mathtt{t}$ *per almeno un* $d \in \mathbb{N}$*. Quindi è vera, perché scegliendo per esempio* $d = 0$ *abbiamo* $[\![(x \leq 1)]\!]^{[x\mapsto 0]}_{\mathcal{I}_\mathbb{N}} = ([\![x]\!]^{[x\mapsto 0]}_{\mathcal{I}_\mathbb{N}} \leq 1) = (0 \leq 1) = \mathtt{t}$*.*

4. $[\![(\forall x\,.\,(\exists y\,.\,x \leq y))]\!]_{\mathcal{I}_\mathbb{N}}$*: per la clausola 7 della Def. 8.7.8, è vera se e solo se* $[\![(\exists y\,.\,x \leq y)]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}} = \mathtt{t}$ *per ogni* $d \in \mathbb{N}$*. A sua volta,* $[\![(\exists y\,.\,x \leq y)]\!]^{[x\mapsto d]}_{\mathcal{I}_\mathbb{N}}$ *è vera se e solo se (clausola 8)* $[\![(x \leq y)]\!]^{[x\mapsto d][y\mapsto d']}_{\mathcal{I}_\mathbb{N}} = \mathtt{t}$ *per almeno un* $d' \in \mathbb{N}$*. Qualunque sia il valore* d*, possiamo sicuramente trovare un* d' *tale che* $d \leq d'$*, per esempio* $d' := d + 1$*. Quindi la formula è vera.*

5. $[\![(\exists y\,.\,(\forall x\,.\,x \leq y))]\!]_{\mathcal{I}_\mathbb{N}}$: *per la clausola 8 della Def. 8.7.8, è vera se e solo se* $[\![(\forall x\,.\,x \leq y)]\!]_{\mathcal{I}_\mathbb{N}}^{[y\mapsto d]} = \mathtt{t}$ *per almeno un elemento* $d \in \mathbb{N}$. *A sua volta,* $[\![(\forall x\,.\,x \leq y)]\!]_{\mathcal{I}_\mathbb{N}}^{[y\mapsto d]}$ *è vera se e solo se (clausola 7)* $[\![(x \leq y)]\!]_{\mathcal{I}_\mathbb{N}}^{[y\mapsto d][x\mapsto d']} = \mathtt{t}$ *per ogni* $d' \in \mathbb{N}$. *Ora, supponendo di aver fissato un valore per* d, *non è possibile che per ogni* $d' \in \mathbb{N}$ *valga* $d' \leq d$ *(basta prendere* $d' := d + 1$*). Quindi la formula è falsa. Gli esempi 4 e 5 mostrano che in generale scambiando le posizioni di un quantificatore esistenziale e di uno universale il significato di una formula può cambiare, come avevamo già visto nel sesto esempio della Sezione 8.6.*

Per concludere questa sezione, ricordiamo che tutti i concetti introdotti per il calcolo proposizionale nella Definizione 8.1.10 si applicano anche alla logica dei predicati, considerando come interpretazioni quelle introdotte nella Definizione 8.7.3, e le formule predicative chiuse invece di quelle proposizionali. Riportiamo qui le definizioni con gli aggiustamenti necessari.

Definizione 8.7.12 (modello, equivalenza, conseguenza logica)**.** *Data una formula predicativa chiusa* P *e una interpretazione* $\mathcal{I}$, *diciamo che* $\mathcal{I}$ *è un* MODELLO *di* P *se* P *è vera in* $\mathcal{I}$, *cioè se* $[\![P]\!]_\mathcal{I} = \mathtt{t}$, *e in questo caso scriviamo:*

$$\mathcal{I} \models P \qquad\qquad (\mathcal{I} \text{ è modello di } P)$$

Se invece $[\![P]\!]_\mathcal{I} = \mathtt{f}$ *scriviamo* $\mathcal{I} \not\models P$. *La notazione si estende nel modo ovvio a un insieme di formule* Γ *(letto "Gamma"): scriviamo* $\mathcal{I} \models \Gamma$ *se* $\mathcal{I} \models P$ *per ogni* $P \in \Gamma$, *mentre scriviamo* $\mathcal{I} \not\models \Gamma$ *se c'è almeno una formula* $P \in \Gamma$ *tale che* $\mathcal{I} \not\models P$.

Due formule chiuse P *e* B *sono* LOGICAMENTE EQUIVALENTI *se hanno gli stessi modelli, cioè assumono lo stesso valore di verità per qualunque interpretazione. In questo caso scriviamo:*

$$P \equiv B \qquad\qquad (P \text{ e } B \text{ sono logicamente equivalenti})$$

Data una formula chiusa P *e un insieme di formule* Γ, *diciamo che* P è una CONSEGUENZA LOGICA di Γ *se* P *è vera in ogni interpretazione che rende vere tutte le formule di* Γ, *oppure, equivalentemente, se ogni modello di* Γ *è anche un modello di* P. *In questo caso scriviamo:*

$$\Gamma \models P \qquad\qquad (P \text{ è conseguenza logica di } \Gamma)$$

8.8 Dimostrazioni per sostituzione di formule valide

La Definizione 8.7.9 stabilisce come, fissata un'interpretazione $\mathcal{I}$ per un alfabeto del primo ordine $\mathcal{A}$, si può assegnare un valore di verità a ogni formula predicativa chiusa. Questo ci permette di introdurre per la logica dei predicati una classificazione delle formule simile a quanto visto con la Definizione 8.2.5 per il calcolo proposizionale.

Definizione 8.8.1 (formule valide, soddisfacibili e insoddisfacibili)**.** *Una* FORMULA VALIDA *è una formula predicativa chiusa che è sempre vera, per qualunque interpretazione. Se* P *è valida, scriveremo anche* $\models P$. *Una* FORMULA INSODDISFACIBILE *è una formula predicativa chiusa che è sempre falsa, per qualunque interpretazione. Una formula predicativa chiusa è* SODDISFACIBILE *se esiste almeno una interpretazione per la quale è vera.*

Esercizio 8.8.2. *Si dimostri che ogni tautologia* P, *vista come formula della logica dei predicati, è valida. Si dimostri che, analogamente, ogni contraddizione* P *è insoddisfacibile.*

Rispetto al calcolo proposizionale il problema principale è che le interpretazioni possibili da considerare generalmente sono infinite. Inoltre, anche se fissiamo un'interpretazione, il dominio dei valori da considerare potrebbe essere infinito. Questo rende il problema di dimostrare l'equivalenza o la conseguenza logica estremamente più complicati. Infatti non esiste per la logica dei predicati una tecnica analoga a quella delle tavole di verità.

Fortunatamente, il principio di sostituzione e tutte le leggi relative ai connettivi logici che abbiamo visto per il calcolo proposizionale continuano a valere. Più precisamente, il sistema di dimostrazioni introdotto nella Definizione 8.3.12 è corretto anche per la logica dei predicati. Quindi possiamo dimostrare la validità di certe formule con delle dimostrazioni per sostituzione, sfruttando le leggi o altre formule valide già note. Alle leggi già presentate per il calcolo proposizionale aggiungiamone alcune che riguardano le formule quantificate.

Teorema 8.8.3 (alcune leggi sui quantificatori)**.** *Le seguenti doppie implicazioni su formule quantificate chiuse sono valide:*

commutatività	$(\exists x\,.\,(\exists y\,.\,P)) \Leftrightarrow (\exists y\,.\,(\exists x\,.\,P))$	$(\forall x\,.\,(\forall y\,.\,P)) \Leftrightarrow (\forall y\,.\,(\forall x\,.\,P))$
distributività	$(\exists x\,.\,(P \vee B)) \Leftrightarrow ((\exists x\,.\,P) \vee (\exists x\,.\,B))$	$(\forall x\,.\,(P \wedge B)) \Leftrightarrow ((\forall x\,.\,P) \wedge (\forall x\,.\,B))$
De Morgan	$\neg(\exists x\,.\,P) \Leftrightarrow (\forall x\,.\,\neg P)$	$\neg(\forall x\,.\,P) \Leftrightarrow (\exists x\,.\,\neg P)$

Tabella 8.4: Leggi per quantificatori

La dimostrazione di queste leggi va al di là degli obiettivi di questa dispensa, ma usando le dimostrazioni per sostituzione possiamo mostrare che in ogni coppia di leggi sulla stessa riga della tavola l'una è conseguenza logica dall'altra.

Esempio 8.8.4 (dimostrazioni per sostituzione: $\neg(\forall x\,.\,P) \Leftrightarrow (\exists x\,.\,\neg P)$)**.** *Vediamo per esempio come la seconda legge di De Morgan può essere dimostrata usando la prima:*

$\neg(\forall x\,.\,\underline{A})$
$\Leftrightarrow$ *(doppia negazione)*
$\neg(\underline{\forall x\,.\,\neg(\neg A)})$
$\Leftrightarrow$ *(prima legge (De Morgan), applicata da destra verso sinistra sostituendo A con $\neg A$)*
$\neg(\neg(\exists x\,.\,\neg A))$
$\Leftrightarrow$ *(doppia negazione)*
$(\exists x\,.\,\neg A)$

Esempio 8.8.5 (dimostrazioni per sostituzione: $(\forall x\,.\,(A \wedge B)) \Leftrightarrow ((\forall x\,.\,A) \wedge (\forall x\,.\,B))$)**.** *In modo analogo al caso precedente, possiamo dimostrare la seconda legge di distributività usando la prima:*

$(\forall x\,.\,(A \wedge B))$
$\Leftrightarrow$ *(doppia negazione)*
$\neg(\underline{\neg(\forall x\,.\,(A \wedge B))})$
$\Leftrightarrow$ *(De Morgan) per $\forall$*
$\neg(\exists x\,.\,\underline{\neg(A \wedge B)})$
$\Leftrightarrow$ *(De Morgan) per $\wedge$*
$\neg(\underline{\exists x\,.\,(\neg A \vee \neg B)})$
$\Leftrightarrow$ *(distributività) per $\exists$*
$\neg(\underline{(\exists x\,.\,\neg A)} \vee \underline{(\exists x\,.\,\neg B)})$
$\Leftrightarrow$ *(De Morgan) per $\forall$, due volte*
$\neg(\underline{\neg(\forall x\,.\,A) \vee \neg(\forall x\,.\,B)})$
$\Leftrightarrow$ *(De Morgan)*
$\neg(\neg((\forall x\,.\,A) \wedge (\forall x\,.\,B)))$
$\Leftrightarrow$ *(doppia negazione)*
$((\forall x\,.\,A) \wedge (\forall x\,.\,B))$

Relativamente alle leggi di distributività per i quantificatori, è bene riflettere sul fatto che il quantificatore esistenziale distribuisce *solo* rispetto alla disgiunzione, mentre quello universale *solo* rispetto alla congiunzione. Infatti, per esempio, la formula $(\forall x\,.\,(A \vee B)) \Leftrightarrow ((\forall x\,.\,A) \vee (\forall x\,.\,B))$ non è valida, anche se è soddisfacibile.

Esempio 8.8.6 (controesempio per $(\forall x\,.\,(A \vee B)) \Leftrightarrow ((\forall x\,.\,A) \vee (\forall x\,.\,B))$)**.** *Come controesempio consideriamo l'alfabeto dei naturali con la corrispondente interpretazione $\mathcal{I}_{\mathbb{N}}$, e le due formule*

$$(\forall x\,.\,(\mathit{pari}(x) \vee \mathit{dispari}(x))) \qquad ((\forall x\,.\,\mathit{pari}(x)) \vee (\forall x\,.\,\mathit{dispari}(x)))$$

Chiaramente $\mathcal{I}_{\mathbb{N}} \models (\forall x\,.\,(\mathit{pari}(x) \vee \mathit{dispari}(x)))$, perché è vero che ogni numero naturale o è pari o è dispari. Però abbiamo $\mathcal{I}_{\mathbb{N}} \not\models (\forall x\,.\,\mathit{pari}(x))$, perché non è vero che tutti i naturali sono pari, e analogamente $\mathcal{I}_{\mathbb{N}} \not\models (\forall x\,.\,\mathit{dispari}(x))$. Quindi abbiamo $\mathcal{I}_{\mathbb{N}} \not\models ((\forall x\,.\,\mathit{pari}(x)) \vee (\forall x\,.\,\mathit{dispari}(x)))$.

Analogamente, la formula $(\exists x \,.\, (A \wedge B)) \Leftrightarrow ((\exists x \,.\, A) \wedge (\exists x \,.\, B))$ non è valida, cioè la quantificazione esistenziale non distribuisce in generale rispetto alla congiunzione. Il lettore è invitato a trovare un controesempio a questa equivalenza, magari sfruttando quello visto precedentemente.

Concludiamo quest'analisi delle leggi per i quantificatori osservando che la legge di commutatività vale solo per coppie di quantificatori dello stesso tipo. Infatti la formula $(\exists y \,.\, (\forall x \,.\, A))) \Leftrightarrow (\forall x \,.\, (\exists y \,.\, A))$ non è valida, anche se è soddisfacibile.

Intuitivamente, la formula $(\exists y \,.\, (\forall x \,.\, A))$ asserisce che esiste (almeno) un particolare valore di y che rende valida A indipendentemente dall'x considerato. La formula $(\forall x \,.\, (\exists y \,.\, A))$ invece asserisce che per ciascun valore x possiamo trovare un valore y che rende vera A, ma per valori di x diversi potrebbero servire valori di y diversi.

Esempio 8.8.7 (controesempio per $(\exists y \,.\, (\forall x \,.\, A))) \Leftrightarrow (\forall x \,.\, (\exists y \,.\, A))$)**.** *Un esempio che rende falsa questa formula si ottiene facilmente dalle formule dei punti 4 e 5 dell'Esempio 8.7.11. Infatti abbiamo visto che* $\mathcal{I}_{\mathbb{N}} \models (\forall x \,.\, (\exists y \,.\, x \leq y))$ *perché la formula asserisce che per ogni naturale possiamo trovarne uno maggiore, cosa che è ovviamente vera. Ma* $\mathcal{I}_{\mathbb{N}} \not\models (\exists y \,.\, (\forall x \,.\, x \leq y))$*: infatti questa formula asserisce che esiste un naturale più grande di tutti gli altri, cosa che è chiaramente falsa. Quindi possiamo concludere che*

$$\mathcal{I}_{\mathbb{N}} \not\models (\exists y \,.\, (\forall x \,.\, x \leq y)) \Leftrightarrow (\forall x \,.\, (\exists y \,.\, x \leq y))$$

e quindi la formula non è valida.

Formalizzazione di enunciati sulla teoria degli insiemi

Nel Capitolo 1 abbiamo introdotto le principali relazioni e operazioni tra insiemi in modo preciso ma discorsivo, usando il linguaggio naturale. Un utile esercizio di formalizzazione consiste nel riformulare quelle definizioni usando formule predicative. La formalizzazione risulta particolarmente utile quando tali definizioni devono essere manipolate in qualche modo, come vedremo nell'Esempio 8.8.10. Nel resto di questa sezione useremo l'alfabeto per gli insiemi $\mathcal{A}_{\mathcal{S}}$ introdotto nell'Esempio 8.5.4.

Esempio 8.8.8 (formalizzazione di relazioni tra insiemi)**.** *Con formule predicative basate sull'alfabeto degli insiemi* $\mathcal{A}_{\mathcal{S}}$ *possiamo definire in modo rigoroso alcune delle relazioni tra insiemi introdotte nelle Definizioni 1.2.1 e 1.2.5. Siano* A *e* B *due generici insiemi:*

Inclusione: "A è un **sottoinsieme** di B, scritto $A \subseteq B$, se e solo se ogni elemento di A è anche elemento di B"

$$(A \subseteq B \quad \Leftrightarrow \quad (\forall x \,.\, (x \in A \Rightarrow x \in B))) \tag{8.13}$$

Disuguaglianza: "A e B sono **diversi**, scritto $A \neq B$, se e solo se esiste almeno un elemento che è contenuto in uno dei due insiemi ma non nell'altro"

$$(A \neq B \quad \Leftrightarrow \quad (\exists x \,.\, ((x \in A \ \wedge \ x \notin B) \vee (x \notin A \ \wedge \ x \in B)))) \tag{8.14}$$

Inclusione stretta: "A è un **sottoinsieme stretto** di B, scritto $A \subset B$, se e solo se $A \subseteq B$ e $A \neq B$"

$$(A \subset B \quad \Leftrightarrow \quad (A \subseteq B \ \wedge \ A \neq B)) \tag{8.15}$$

Naturalmente possiamo anche formalizzare le classiche operazioni su insiemi.

Esempio 8.8.9 (formalizzazione di operazioni su insiemi)**.** *Definiamo con formule predicative basate sull'alfabeto degli insiemi le operazioni su insiemi, introdotte nella Sezione 1.3, e in particolare la relazione di appartenenza (*$\in$*) per insiemi ottenuti come risultato di queste operazioni. Siano* A *e* B *due generici insiemi:*

Intersezione:	$(x \in A \cap B$	$\Leftrightarrow$	$(x \in A \ \wedge \ x \in B))$
Unione:	$(x \in A \cup B$	$\Leftrightarrow$	$(x \in A \ \vee \ x \in B))$
Complemento (rispetto a $\mathcal{U}$)	$(x \in \overline{A}$	$\Leftrightarrow$	$(x \in \mathcal{U} \ \wedge \ x \notin A))$
Differenza:	$(x \in A \setminus B$	$\Leftrightarrow$	$(x \in A \ \wedge \ x \notin B))$

Esempio 8.8.10 (dimostrazioni su insiemi)**.** *Nell'Esempio 8.8.8 abbiamo formalizzato con la seguente formula la relazione di* inclusione stretta *tra due insiemi:*

$$(A \subset B \quad \Leftrightarrow \quad (A \subseteq B \ \wedge \ A \neq B))$$

D'altra parte dopo la Definizione 1.2.5 avevamo affermato che "per dimostrare che $A \subset B$ si può mostrare che ogni elemento di A appartiene a B, ma che esiste un elemento di B che non appartiene ad A"*. Formalizzando questa frase in logica dei predicati otteniamo la seguente formula:*

$$(((\forall x\,.\,(x \in A \Rightarrow x \in B) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B))) \quad \Rightarrow \quad A \subset B)$$

Vediamo che questo è vero, mostrando che premessa e conseguenza sono anzi logicamente equivalenti:

$A \subset B$
$\Leftrightarrow$ *(formula (8.15) dell'Esempio 8.8.8)*
$(\underline{A \subseteq B} \ \wedge \ A \neq B)$
$\Leftrightarrow$ *(formula (8.13) dell'Esempio 8.8.8)*
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ \underline{A \neq B})$
$\Leftrightarrow$ *(formula (8.14) dell'Esempio 8.8.8)*
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ \underline{(\exists x\,.\,((x \in A \ \wedge \ x \notin B) \vee (x \notin A \ \wedge \ x \in B)))})$
$\Leftrightarrow$ *((distributività) di $\exists$ su $\vee$)*
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \wedge \ ((\exists x\,.\,(x \in A \ \wedge \ x \notin B)) \vee (\exists x\,.\,(x \notin A \ \wedge \ x \in B))))$
$\Leftrightarrow$ *((distributività) di $\wedge$ su $\vee$)*
$(((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ \underline{(\exists x\,.\,(x \in A \ \wedge \ x \notin B))})\vee$
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B))))$
$\Leftrightarrow$ *((doppia negazione) e (De Morgan))*
$(((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ \neg(\forall x\,.\,\underline{\neg(x \in A \ \wedge \ x \notin B)}))\vee$
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B))))$
$\Leftrightarrow$ *((De Morgan) e (eliminazione implicazione) al contrario)*
$(\underline{((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ \neg(\forall x\,.\,(x \in A \Rightarrow x \in B)))}\vee$
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B))))$
$\Leftrightarrow$ *(contraddizione)*
$(\mathsf{F} \vee ((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B))))$
$\Leftrightarrow$ *(unità)*
$((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B)))$

Cerchiamo ora di esplicitare come si può dimostrare che $A \not\subset B$. Abbiamo:

$A \not\subset B$
$\Leftrightarrow$ *(per la dimostrazione appena fatta)*
$\neg((\forall x\,.\,(x \in A \Rightarrow x \in B)) \ \wedge \ (\exists x\,.\,(x \notin A \ \wedge \ x \in B)))$
$\Leftrightarrow$ *((De Morgan) per $\wedge$)*
$(\underline{\neg(\forall x\,.\,(x \in A \Rightarrow x \in B))} \ \vee \ \underline{\neg(\exists x\,.\,(x \notin A \ \wedge \ x \in B))})$
$\Leftrightarrow$ *((De Morgan), due volte)*
$((\exists x\,.\,\underline{\neg(x \in A \Rightarrow x \in B)}) \ \vee \ (\forall x\,.\,\underline{\neg(x \notin A \ \wedge \ x \in B)}))$
$\Leftrightarrow$ *((eliminazione implicazione) e (De Morgan))*
$((\exists x\,.\,\underline{\neg(\neg(x \in A) \ \vee \ x \in B)}) \ \vee \ (\forall x\,.\,\underline{(x \in A \ \vee \ \neg(x \in B))}))$
$\Leftrightarrow$ *((De Morgan), (doppia negazione) e (eliminazione implicazione) al contrario)*
$((\exists x\,.\,(x \in A \ \wedge \ x \notin B)) \ \vee \ (\forall x\,.\,(x \in B \ \Rightarrow x \in A)))$

Rileggendo questa formula in italiano possiamo concludere che *"per dimostrare che $A \not\subset B$ si può mostrare che esiste un elemento di A che non appartiene a B, oppure che ogni elemento di B appartiene ad A".*