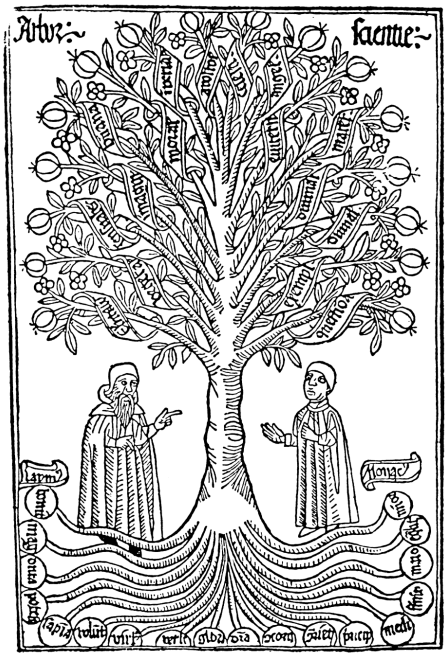
LOGICA MATEMATICA

LUCA SPADA



per il corso di laurea in Informatica

Dicembre 2012 – versione 1.4

Via via che la logica si perfeziona, diminuisce il numero delle cose in cui si crede.

— Bertrand Russell

PANORAMICA DEL CORSO

Questo corso è pensato per compiere i primi passi attraverso lo studio del linguaggio e del ragionamento formale. Gli obiettivi finali del corso sono:

1. il raggiungimento di una consapevolezza nel riconoscimento e l'uso delle regole logiche;
2. capacità di formalizzazione di informazioni vaghe in un linguaggio formale;
3. sviluppo e verifica di ragionamenti corretti;

Gli argomenti principali del corso sono:

1. La sintassi della logica proposizionale,
2. La semantica della logica proposizionale,
3. Algoritmi per riconoscere le tautologie proposizionali,
4. La completezza della logica proposizionale,
5. La sintassi della logica del primo ordine,
6. La semantica della logica del primo ordine,
7. Algoritmi per riconoscere le tautologie del primo ordine,
8. La completezza della logica del primo ordine.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth

RINGRAZIAMENTI

Queste note sono state sviluppate partendo dalle note del prof. Francesco Bottacin disponibili su

http://www.math.unipd.it/~bottacin/books/logica.pdf

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*". `classicthesis` is available for both LaTeX and LyX:

http://code.google.com/p/classicthesis/

INDICE

1 INTRODUZIONE

Nella vita siamo abituati a gestire quotidianamente una notevole quantità di informazione: memorizziamo, rielaboriamo, verifichiamo, confutiamo, facciamo ipotesi e traiamo conclusioni. Seppure questi processi si rilevino la maggior parte delle volte corretti, succede che in alcuni casi commettiamo degli *errori di valutazione* o sbagliamo a trarre delle conclusioni. Ciò nondimeno Il nostro cervello è una potentissima macchina tutt'ora inimitabile da qualsiasi computer.

Nel nostro ragionamento utilizziamo diverse tecniche che trovano base in diverse esperienze riscontrabili nella vita. Ad esempio escludiamo automaticamente ipotesi che non siano abbastanza verosimili: raramente uno accetterebbe come ipotesi che il "sole non sorga domani mattina", eppure questo evento, seppure con una probabilità bassissima, potrebbe in qualche modo avvenire; quindi una parte del nostro ragionamento è di tipo **probabilistico**. Allo stesso modo siamo capaci, osservando un evento, di risalirne alle cause: sentendo delle gocce d'acqua in testa, penseremo che sta piovendo, seppure non sia questa l'unica ipotesi plausibile; un tale ragionamento si chiama **abduttivo**. Ancora, siamo capaci di estrarre delle regole *universali* da casi *particolari*: vedendo una fila di uomini ed una di donne davanti a dei bagni ci metteremo in coda secondo una regola che in effetti abbiamo creato noi al momento e che potrebbe anche essere sbagliata; un tale ragionamento si chiama **induttivo**.

Infine possediamo una tecnica di ragionamento che, se usata correttamente, non ci conduce *mai* in errore, si chiama ragionamento **deduttivo**. Tale tecnica combina un certo numero di **ipotesi** (che crediamo siano vere) e produce una **tesi**, la cui verità è strettamente collegata a quella delle ipotesi: in ogni caso le ipotesi siano vere anche la tesi lo sarà. Uno degli esempi più antichi di tale ragionamento risale agli antichi greci:

$$\begin{array}{rl} \text{Ipotesi} & \begin{cases} \text{Ogni animale è mortale,} \\ \text{Ogni uomo è un animale,} \end{cases} \\ \hline \text{Tesi} & \left\{ \text{Ogni uomo è mortale.} \right. \end{array}$$

Notiamo innanzitutto che la correttezza di tale ragionamento non dipende dalla veridicità delle frasi: il seguente ragionamento, che segue lo stesso schema di quello qui sopra, rimane corretto:

$$\begin{array}{c} \text{Ogni animale è buono,} \\ \text{Ogni uomo è un animale,} \\ \hline \text{Ogni uomo è buono.} \end{array}$$

Il ragionamento è corretto: se accettiamo che le frasi 'ògni animale è buono" e 'ògni uomo è un animale" allora dobbiamo *necessariamente*

accettare anche 'ògni uomo è buono". Se tale tesi non rispecchia la realtà vorrà dire che c'è qualche problema con le nostre ipotesi, non con il nostro ragionamento.

Quindi la validità di un ragionamento non dipende dalla validità delle singole frasi, ma dalla *struttura* del ragionamento. Possiamo quindi definire un ragionamento corretto, come una regola che ci permette di arrivare ad una **conclusione vera ogni volta che le ipotesi sono vere**.

Un secondo fatto che andrebbe notato a proposito dei ragionamenti qui sopra è che essi ci forniscono *nuovo informazione*. Consideriamo il seguente esempio.

> Su un tavolo ci sono 6 scrigni; su ognuno di essi c'è un'iscrizione.
>
> 1. Sul primo scrigno l'iscrizione dice: «nessuno di questi 6 scrigni contiene un tesoro».
> 2. Sul secondo scrigno l'iscrizione dice: «1 solo di questi 6 scrigni contiene un tesoro».
> 3. Sul terzo scrigno l'iscrizione dice: «2 soli di questi 6 scrigni contengono un tesoro».
> 4. Sul quarto scrigno l'iscrizione dice: «3 soli di questi 6 scrigni contengono un tesoro».
> 5. Sul quinto scrigno l'iscrizione dice: «4 soli di questi 6 scrigni contengono un tesoro».
> 6. Sul sesto scrigno l'iscrizione dice: «5 soli di questi 6 scrigni contengono un tesoro».

Sapendo che ciascuno scrigno contiene un tesoro *se e solo* se l'iscrizione su di esso dice il vero, possiamo stabilire in quali scrigni c'è un tesoro?

Cominciamo a ragionare, innanzitutto le frasi non possono essere tutte vere, perché si contraddicono tra di loro: se è vero che «2 soli di questi 6 scrigni contengono un tesoro» allora non può anche essere vero che «4 soli di questi 6 scrigni contengono un tesoro». Quindi al più una può essere vera. E se fossero tutte false? Neanche questo può succedere, perché se fossero tutte false non ci sarebbe nessuno scrigno con un tesoro, ma questo è proprio quello che c'è scritto sul primo scrigno, quindi esso dovrebbe contenere un tesoro ed abbiamo quindi una contraddizione. Allora abbiamo stabilito che esattamente una delle frasi deve essere vera. Ovviamente la frase vera non può essere l'ultima, perché dice che ci sono cinque scrigni contenenti un tesoro, e se questo fosse vero cinque scritte dovrebbero essere vere. Per lo stesso motivo non può essere vera quella sul penultimo scrigno

e così via. L'unica frase che può essere vera è quella che dice «1 solo di questi 6 scrigni contiene un tesoro», per cui ora sappiamo che l'unico scrigno che contiene un tesoro è il secondo.

Il ragionamento eseguito qui sopra è logicamente corretto, esso rimane valido a prescindere che le informazioni che ci hanno dato sugli scrigni e sulle scritte siano vere o false. Ciò che conta è che **se** le informazioni che ci hanno dato sono corrette, allora la nostra tesi deve per forza valere.

Ricapitolando, la nostra mente è capace di manipolare informazioni in vari modi, alcuni di questi funzionano nella maggior parte dei casi, ma non sempre. Una delle nostre tecniche di ragionamento invece ci permette di avere estrema certezza che, se le informazioni che stiamo manipolando sono corrette, allora lo sarà anche la nostra conclusione. Chiameremo tale tecnica **ragionamento logico** ed esso sarà l'oggetto del nostro studio.

La prima cosa da fare per studiare le regole che governano il ragionamento logico è cercare di minimizzare le ambiguità contenute nel nostro linguaggio. Quando per esempio si dice «aiuteremo gli anziani o i bambini» si intende dire che verrà aiutata l'una o l'atra classe di persone ma non entrambe? Oppure potrebbero anche essere aiutate entrambe? Quando si dice «Se viene Marco io non vengo e Antonio viene» si intende dire che Antonio viene se viene Marco? Oppure Antonio viene comunque?

Per eliminare questo tipo di ambiguità costruiremo un linguaggio formale in cui definiremo **univocamente** i significati di ogni simbolo, in maniera che si comportino come le parti del discorso che vogliamo studiare. Cominceremo a costruire prima un linguaggio più semplice, che chiameremo **linguaggio proposizionale**, per poi passare ad uno un po' più complesso (che però possiede più potere espressivo) che chiameremo **linguaggio del primo ordine**.

Parte I

LOGICA PROPOSIZIONALE

2 SINTASSI E SEMANTICA

2.1 PROPOSIZIONI.

Il primo passo da compiere è definire quali sono le parti del discorso che ci interessano in questo studio. Una **proposizione** è un'affermazione che possiede un valore di verità, cioè una affermazione che è VERA oppure FALSA. Ad esempio:

- "Marco è alto 1,75m"
- "5 è un numero dispari"
- "Roma è la capitale della Francia"

sono proposizioni. Al contrario, l'affermazione 'àlzati' non è una proposizione (esprime un ordine e non un fatto che può essere vero o falso). Le proposizioni possono essere combinate tra loro per costruire delle proposizioni più complesse utilizzando dei connettivi, quali " e", " o", "non", "se... allora ...", ecc. Chiameremo **atomiche** quelle proposizioni che non sono ottenute da proposizioni più semplici mediante l'uso di connettivi. Ad esempio:

1. "4 è un numero pari" è una proposizione atomica,
2. "Se c'è il sole allora vado al mare" è una proposizione composta (le cui componenti sono "C'è il sole" e "vado al mare").

2.2 IL LINGUAGGI FORMALE

Il linguaggio naturale è troppo complesso e ambiguo. Perciò il primo passo da fare per poter studiare le regole logiche è costruire un nuovo linguaggio in cui formalizzare i ragionamenti. Lo scopo è quello di fornire un linguaggio abbastanza espressivo da poterci tradurre i ragionamenti di tutti i giorni, ma costituito da una rigida sintassi e un significato univoco per ogni suo oggetto, in maniera da poterlo studiare matematicamente. Chiameremo questo ambiente il **linguaggio formale**. Per costruire un linguaggio formale bisogna fissare un **alfabeto**, cioè un insieme di simboli che ci serviranno a costruire delle "frasi" (che, in questo contesto, chiameremo **formule** o **proposizioni**). Le "frasi" non sono altro che delle sequenze finite (**stringhe**) di simboli che appartengono all'alfabeto che abbiamo fissato. Servirà poi una sintassi, cioè un insieme di regole per stabilire quali sequenze di simboli sono accettabili nel nostro linguaggio e quali no. Attenzione: la sintassi si occupa solo della forma delle frasi e non del loro contenuto.

Esempio 2.2.1. Nel linguaggio dell'Aritmetica, consideriamo le seguenti "frasi" (cioè formule):

- $2\Delta(3+1) = 8$, formula sintatticamente corretta,
- $2 + (4(- ==)$, formula sintatticamente non corretta
- $2 + 1 = 6$, formula sintatticamente corretta.

Si noti che la prima e la terza formula sono entrambe sintatticamente corrette (anche se la prima è VERA e la terza è FALSA). In particolare si noti che, nel caso di una formula non sintatticamente corretta, non ha nessun senso chiederci se è VERA o falsa; essa è semplicemente una sequenza di simboli priva di senso. Le formule sintatticamente corrette saranno chiamate Formule Ben Formate (FBF). Riassumendo: il compito della sintassi è quello di fornire un insieme di regole per costruire le FBF. Solo quando una formula è sintatticamente corretta si può poi parlare del suo significato. Questo è il compito della semantica: assegnare un significato a tutte le frasi sintatticamente corrette (cioè a tutte le FBF).

2.3 IL LINGUAGGIO DEL CALCOLO PROPOSIZIONALE

Come abbiamo visto, per capire se un certo ragionamento è corretto oppure no, non è importante sapere se le particolare proposizioni usate sono vere o false. Quello che ci interessa è che ragionamento *preservi la verità delle ipotesi nella tesi*. Possiamo quindi del tutto abbandonare le frasi concrete e sostituire con dei simboli. Nel Calcolo Proposizionale manipoleremo delle proposizioni e le indicheremo con delle lettere maiuscole $A, B, C, \ldots, P, Q, \ldots$ (eventualmente con delle lettere con indici, come $A_1, A_2, A_3, \ldots$). Le proposizioni si possono poi combinare tra loro mediante l'uso dei connettivi. Pertanto il nostro alfabeto dovrà contenere:

- Simboli per indicare le proposizioni: $A_1, B_1, C_1, A_2, B_2, C_2, \ldots$,
- Simboli per indicare i connettivi: $\neg, \wedge, \vee, \rightarrow$
- Simboli accessori, come le parentesi: (e).

Per comodità, introduciamo anche il simbolo $\perp$ che servirà ad indicare la falsità, l'assurdo. Ora stabiliamo le regole della sintassi, cioè spieghiamo come si costruiscono le FBF.

Definizione 2.3.1. Le regole per la formazione di **FBF** (**formule ben formate**) sono le seguenti:

- $A, B, C, \ldots$ e $\perp$ sono FBF;
- le altre FBF si ottengono combinando delle FBF già costruite mediante l'uso dei connettivi, nei modi seguenti: Se X e Y sono FBF, allora anche $(\neg X)$, $(X \wedge Y)$, $(X \vee Y)$ e $(X \rightarrow Y)$ sono FBF.
- le istruzioni qui sopra sono gli unici metodi permessi per ottenere FBF.

Esempio 2.3.2. Le seguenti sono FBF:

- $((A \vee B) \to (C \wedge D))$,
- $((A \to B) \to (\neg C))$,
- $(((\neg A) \vee \bot) \wedge (\neg C))$.

Al contrario, la seguente sequenza di simboli non è una FBF: $\wedge \to A \to ($

Osservazione 2.3.3. Si noti che (per ora) i simboli $\neg, \wedge, \vee, \to$ non hanno nessun significato; essi sono solo dei simboli che devono essere manipolati in modo puramente formale, secondo le regole della sintassi enunciate prima.

Osservazione 2.3.4. Le regole che abbiamo stabilito conducono ad un uso eccessivo delle parentesi. Per semplificare l'aspetto delle formule conviene stabilire delle priorità tra i vari simboli:

priorità più alta	$\neg$
$\vdots$	$\wedge$
$\vdots$	$\vee$
$\vdots$	$\leftrightarrow$
priorità più bassa	$\to$

Pertanto la formula

$$A \wedge \neg B \to C$$

deve essere interpretata come segue

$$((A \wedge (\neg B)) \to C)$$

mentre la formula

$$\neg A \wedge B \vee C \to \neg D$$

significa

$$((((\neg A) \wedge B) \vee C) \to (\neg D)) \ .$$

Invece, nella formula

$$(A \to B) \vee C$$

non si possono togliere le parentesi, perché la formula

$$A \to B \vee C$$

verrebbe interpretata come

$$(A \to (B \vee C)) \ .$$

2.4 LA SEMANTICA DEL CALCOLO PROPOSIZIONALE

Bisogna ora attribuire un significato (VERO o FALSO) a tutte le formule sintatticamente corrette (le FBF). Indichiamo i valori di verità VERO con 1 e FALSO con 0. Si tratta quindi di definire una funzione $v\colon FBF \to \{0,1\}$, che associa ad ogni formula ben formata X il suo valore di verità $v(X)$. Dato che ogni FBF si ottiene combinando tra loro delle proposizioni atomiche mediante l'uso dei connettivi (secondo le regole della sintassi), per definire una tale funzione v sull'insieme di tutte le FBF basta definirla per le proposizioni atomiche e poi analizzare il comportamento dei vari connettivi. Poniamo quindi

$$v(X) = \begin{cases} 1 & \text{se } X \text{ è vera,} \\ 0 & \text{se } X \text{ è falsa} \end{cases},$$

per ogni proposizione atomica X. Poniamo inoltre $v(\bot) = 0$, dato che il simbolo $\bot$ rappresenta la falsità.

Analizziamo ora il comportamento dei vari connettivi. Per fare ciò scriviamo le tavole di verità: Tavola della verità della **negazione** ($\neg$ = NOT)

X	$\neg X$
0	1
1	0

Tavola della verità della **congiunzione** ($\wedge$ = AND)

X	Y	$X \wedge Y$
0	0	0
0	1	0
1	0	0
1	1	1

Tavola della verità della **disgiunzione** ($\vee$ = OR)

X	Y	$X \vee Y$
0	0	0
0	1	1
1	0	1
1	1	1

Tavola della verità della **implicazione** ($\to$)

X	Y	$X \to Y$
0	0	1
0	1	1
1	0	0
1	1	1

Esercizio 2.4.1. Verificare che $X \to Y$ ha la stessa tavola della verità di $\neg X \vee Y$. Diremo che queste due formule sono semanticamente equivalenti.

Osservazione 2.4.2. I simboli usati per denotare i vari connettivi non sono standard. In contesti diversi si usano spesso simboli diversi da quelli che noi abbiamo introdotto. Ad esempio, nel linguaggio di programmazione C, il simbolo per il connettivo logico AND è &&, mentre quello per il connettivo logico OR è ||. (Sempre nel linguaggio C, si faccia molta attenzione a non confondere gli operatori logici && e || con i cosiddetti "bitwise operators" (operatori bit-a-bit) & e |).

Definizione 2.4.3. Una funzione $v : FBF \to \{0,1\}$ che soddisfa tutte le proprietà precedentemente elencate è detta una **interpretazione**.

Osservazione 2.4.4. Da quanto detto risulta evidente che una interpretazione è univocamente determinata dai valori che essa assume sulle proposizioni atomiche, poiché ogni altra FBF si ottiene combinando delle proposizioni atomiche mediante l'uso di connettivi.

Definizione 2.4.5. Sia X una FBF e v una interpretazione. Se $v(X) = 1$, diremo che X è **soddisfatta** nell'interpretazione v, oppure che v è un **modello** per X. In tal caso si scrive $v \models X$ (X è VERA nell'interpretazione v).

Definizione 2.4.6. Una formula ben formata X è **soddisfacibile** se ha almeno un modello, cioè se esiste almeno una interpretazione in cui X è soddisfatta. In caso contrario X è **insoddisfacibile** (si dice anche che X è una contraddizione).

Esempio 2.4.7. La seguente formula è soddisfacibile: $(A \wedge \neg B) \vee (B \to A)$. Infatti, se consideriamo una interpretazione v tale che $v(A) = 1$ e $v(B) = 0$, si ha $v(\neg B) = 1$, quindi $v(A \wedge \neg B) = 1$, da cui segue che

$$v((A \wedge \neg B) \vee (B \to A)) = 1 \, .$$

Un esempio di formula insoddisfacibile (contraddizione) è dato dalla formula seguente: $A \wedge \neg A$. Infatti, dato che per una qualunque interpretazione v, $v(A)$ può solo essere 0 o 1, la tavola di verità della formula precedente è:

A	$\neg A$	$A \wedge \neg A$
0	1	0
1	0	0

il che significa che la proposizione $A \wedge \neg A$ è sempre FALSA, indipendentemente dal fatto che A sia VERA o falsa.

Definizione 2.4.8. Una formula ben formata X è una **tautologia** se ogni interpretazione v è un modello per X, cioè se X risulta VERA in ogni interpretazione. In tal caso si scriverà $\models X$.

Esempio 2.4.9. La formula

$$A \to A \vee B$$

è una tautologia. Infatti, dato che per una qualunque interpretazione v, $v(A)$ e $v(B)$ possono solo essere 0 o 1, la tavola di verità della formula precedente è:

A	B	$A \vee B$	$A \to A \vee B$
0	0	0	1
0	1	1	1
1	0	1	1
1	1	1	1

il che significa che la proposizione $A \to A \vee B$ è sempre VERA, indipendentemente dal fatto che A o B siano vere o false.

Esempio 2.4.10. Ci chiediamo se la formula $A \to \neg A$ sia soddisfacibile. Scriviamo la tavola di verità:

A	$\neg A$	$A \to \neg A$
0	1	1
1	0	0

Ciò significa che la proposizione $A \to \neg A$ è soddisfacibile e l'interpretazione che la soddisfa è quella che assegna ad A il valore 0, cioè $v(A) = 0$ (in altre parole, la proposizione $A \to \neg A$ è VERA solo quando A è FALSA).

Propozione 2.4.11. *Una formula ben formata X è una tautologia se e solo se $\neg X$ è insoddisfacibile.*

Dimostrazione. La dimostrazione è lasciata per esercizio. □

Osservazione 2.4.12. La proposizione precedente afferma una cosa piuttosto ovvia e cioè che una proposizione X è "sempre VERA" se e solo se la sua negazione $\neg X$ è "sempre falsa."

Definizione 2.4.13. Sia Γ un insieme di formule ben formate. Γ è un **insieme soddisfacibile** se esiste un'interpretazione v tale che $v(X) = 1$, per ogni $X \in \Gamma$ (cioè un'interpretazione che renda vere tutte le proposizioni di Γ).

Γ è insoddisfacibile se, per ogni interpretazione v, esiste almeno una proposizione $X \in \Gamma$ tale che $v(X) = 0$.

Definizione 2.4.14. Sia Γ un insieme di formule ben formate. Diremo che una proposizione YY è **conseguenza semantica** di Γ, e scriveremo $\Gamma \models Y$, se per ogni interpretazione v tale che $v(X) = 1$ per ogni $X \in \Gamma$, si ha anche $v(Y) = 1$.

In altre parole, $\Gamma \models Y$ significa che Y è VERA in tutte le interpretazioni che sono dei modelli per Γ. In caso contrario, scriveremo $\Gamma \not\models Y$. A titolo di esempio dimostriamo i seguenti risultati:

Propozione 2.4.15. *Si ha $\Gamma \models Y$ se e solo se $\Gamma \cup \{\neg Y\}$ è insoddisfacibile.*

Dimostrazione. Ricordando la definizione, si ha $\Gamma \models Y$ se e solo se $v(Y) = 1$ per tutte le interpretazioni v che rendono vere tutte le formule di Γ. Ciò equivale a dire che, per ogni interpretazione v, o esiste qualche $X \in \Gamma$ tale che $v(X) = 0$, oppure si ha $v(Y) = 1$. Ma sYe $v(Y) = 1$,allora $v(\neg Y) = 0$, quindi per ogni interpretazione v, esiste almeno una formula $Z \in \Gamma \cup \{\neg Y\}$ tale che $v(Z) = 0$. Ciò significa che $\Gamma \cup \{\neg Y\}$ è insoddisfacibile. □

Propozione 2.4.16. *Si ha $X \models Y$ se e solo se $\models X \to Y$.*

Dimostrazione. Supponiamo che $X \models Y$. Allora, per ogni interpretazione v tale che $v(X) = 1$ si deve avere anche $v(Y) = 1$. Quindi vale anche $v(X \to Y) = 1$. Se invece l'interpretazione v è tale che $v(X) = 0$, si ha ancora $v(X \to Y) = 1$ (si veda la tavola di verità di $\to$). Pertanto, per ogni interpretazione v, si ha sempre $v(X \to Y) = 1$, il che equivale a dire che $\models X \to Y$.

Viceversa, supponiamo che valga $\models X \to Y$. Allora, per ogni interpretazione v, si ha $v(X \to Y) = 1$. In particolare, se consideriamo un'interpretazione v tale che $v(X) = 1$, dal fatto che $v(X \to Y) = 1$ si deduce che anche $v(Y) = 1$ (si veda la tavola di verità di $\to$). Questo significa che $X \models Y$. □

Usando quest'ultimo risultato si può dimostrare il seguente teorema:

Teorema 2.4.17 (Deduzione semantica)**.** *Per ogni intero $n \geq 1$, si ha*

$$X_1, X_2, \ldots, X_n \models Y$$

se e solo se

$$X_1, X_2, \ldots, X_{n-1} \models X_n \to Y\ .$$

Dimostrazione. La dimostrazione procede per induzione su n. □

Enunciamo ora, senza dimostrarli, alcuni risultati di particolare importanza teorica.

Teorema 2.4.18 (Teorema di Compattezza)**.** *Un insieme Γ di formule ben formate è soddisfacibile se e solo se lo è ogni suo sottoinsieme finito.*

Una formulazione equivalente del Teorema di Compattezza è la seguente:

Teorema 2.4.19. *Un insieme Γ di formule ben formate è insoddisfacibile se e solo se esiste un sottoinsieme finito di Γ che è insoddisfacibile.*

Corollario 2.4.20. *Sia Γ un insieme di formule ben formate e X una proposizione. $\Gamma \models X$ se e solo se esiste un sottoinsieme finito Δ di Γ tale che $\Delta \models X$.*

2.5 EQUIVALENZA SEMANTICA

Due formule si dicono semanticamente equivalenti quando i loro valori di verità coincidono per ogni interpretazione, cioè quando hanno la stessa tavola di verità.

Diamo quindi la seguente definizione:

Definizione 2.5.1. Due formule ben formate X e Y si dicono **semanticamente equivalenti** se, per ogni interpretazione v, si ha $v(X) = v(Y)$. In tal caso scriveremo $X \leftrightarrow Y$.

Teorema 2.5.2. *Si hanno le seguenti equivalenze:*

idempotenza	$X \vee X \leftrightarrow X$
	$X \wedge X \leftrightarrow X$
commutatività	$X \vee Y \leftrightarrow Y \vee X$
	$X \wedge Y \leftrightarrow Y \wedge X$
associatività	$(X \vee Y) \vee Z \leftrightarrow X \vee (Y \vee Z)$
	$(X \wedge Y) \wedge Z \leftrightarrow X \wedge (Y \wedge Z)$
assorbimento	$X \vee (X \wedge Y) \leftrightarrow X$
	$X \wedge (X \vee Y) \leftrightarrow X$
distributività	$X \vee (Y \wedge Z) \leftrightarrow (X \vee Y) \wedge (X \vee Z)$
	$X \wedge (Y \vee Z) \leftrightarrow (X \wedge Y) \vee (X \wedge Z)$
leggi di De Morgan	$\neg(X \vee Y) \leftrightarrow \neg X \wedge \neg Y$
	$\neg(X \wedge Y) \leftrightarrow \neg X \vee \neg Y$
doppia negazione	$\neg\neg X \leftrightarrow X$

Dimostrazione. Basta scrivere le rispettive tavole di verità e controllare che siano uguali (farlo per esercizio). □

2.6 COMPLETEZZA FUNZIONALE

Definizione 2.6.1. Sia X una FBF contenente n proposizioni atomiche distinte $A_1, A_2, \ldots, A_n$. La funzione $f_X \colon \{0,1\}^n \to \{0,1\}$ tale che, per ogni $(x_1, x_2, \ldots, x_n) \in \{0,1\}^n$, si ha $f_X(x_1, x_2, \ldots, x_n) = v(X)$, dove v è una interpretazione tale che $v(A_i) = x_i$, per ogni $i = 1, \ldots, n$, è detta la **funzione di verità** di X.

La funzione di verità di una proposizione X è equivalente alla tavola di verità di X.

Esercizio 2.6.2. Quante sono le possibili tavole di verità per una proposizione X contenente esattamente n proposizioni atomiche distinte?

Anche i connettivi logici definiscono delle funzioni di verità, descritte dalle loro tavole di verità. Ogni funzione $f \colon \{0,1\}^n \to \{0,1\}$ definisce un qualche connettivo n-ario. Ad esempio, se $n = 2$, vi sono $2^{(2^2)} = 16$ funzioni da $\{0,1\}^2$ in $\{0,1\}$, quindi esistono 16 connettivi binari differenti (ricordiamo che noi ne abbiamo definiti solo tre: $\wedge$, $\vee$ e $\to$), i quali corrispondono a tutte le possibili tavole di verità del tipo

X	Y	$X * Y$
0	0	?
0	1	?
1	0	?
1	1	?

Definizione 2.6.3. Dato un insieme di connettivi logici C e un connettivo $c \notin C$, c si dice (semanticamente) derivabile da C se esiste una formula proposizionale X costruita con i soli connettivi di C tale che $f_X = f_c$.

In altre parole, un connettivo c è derivabile dall'insieme di connettivi C se è possibile esprimerlo mediante connettivi di C.

Esempio 2.6.4. Dalle leggi di De Morgan (e dalla doppia negazione) si deduce che

$$X \vee Y \quad \equiv \quad \neg(\neg X \wedge \neg Y) \, ,$$

quindi il connettivo $\vee$ è derivabile dall'insieme di connettivi $\{\neg, \wedge\}$. Analogamente, si ha che $X \wedge Y \quad \equiv \quad \neg(\neg X \vee \neg Y)$, da cui segue che il connettivo $\wedge$ è derivabile dall'insieme di connettivi $\{\neg, \vee\}$.

Esempio 2.6.5. Definiamo il connettivo $\oplus$ mediante la seguente tavola di verità:

X	Y	$X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

Questo connettivo è detto "o esclusivo" (eXclusive OR, XOR). Si noti che esso corrisponde alla somma in $\mathbb{Z}/2\mathbb{Z}$. Analizzando la tavola di verità, si verifica facilmente che si ha $A \oplus B \quad \equiv \quad (A \wedge \neg B) \vee (\neg A \wedge B)$, quindi il connettivo $\oplus$ è derivabile dall'insieme di connettivi $\{\neg, \vee, \wedge\}$.

Esercizio 2.6.6. Poiché abbiamo già osservato che il connettivo $\wedge$ è derivabile dall'insieme di connettivi $\{\neg, \vee\}$, si deduce che il connettivo $\oplus$ è, in effetti, derivabile anch'esso dall'insieme di connettivi $\{\neg, \vee\}$. Si scriva dunque una formula per esprimere $A \oplus B$ usando solo i due connettivi $\neg$ e $\vee$.

Esempio 2.6.7. Definiamo il connettivo $\leftrightarrow$ mediante la seguente tavola di verità:

X	Y	$X \leftrightarrow Y$
0	0	1
0	1	0
1	0	0
1	1	1

Analizzando la tavola di verità, si verifica facilmente che si ha $A \leftrightarrow B \quad \equiv \quad (A \rightarrow B) \wedge (B \rightarrow A)$, quindi il connettivo $\leftrightarrow$ è derivabile dall'insieme di connettivi $\{\wedge, \rightarrow\}$.

Esercizio 2.6.8. Si dimostrino le seguenti equivalenze:

$A \rightarrow B$	è equivalente a	$\neg A \vee B$
$A \vee B$	è equivalente a	$\neg A \rightarrow B$
$A \vee B$	è equivalente a	$\neg(\neg A \wedge \neg B)$
$A \wedge B$	è equivalente a	$\neg(\neg A \vee \neg B)$
$A \wedge B$	è equivalente a	$(((A \rightarrow \bot) \rightarrow \bot) \rightarrow (B \rightarrow \bot)) \rightarrow \bot$
$\neg A$	è equivalente a	$A \rightarrow \bot$
$\bot$	è equivalente a	$A \wedge \neg A$
$A \leftrightarrow B$	è equivalente a	$(A \rightarrow B) \wedge (B \rightarrow A)$
$A \oplus B$	è equivalente a	$(A \wedge \neg B) \vee (\neg A \wedge B)$.

Esercizio 2.6.9. Usando il fatto che il connettivo $\rightarrow$ è derivabile dall'insieme di connettivi $\{\neg, \vee\}$ (vedi Esercizio 2.6.8), si scriva una formula per esprimere $A \leftrightarrow B$ usando solo i due connettivi $\neg$ e $\vee$.

Definizione 2.6.10. Un insieme C di connettivi logici si dice funzionalmente completo se, per ogni $n \geq 1$ e per ogni funzione $f\colon \{0,1\}^n \rightarrow \{0,1\}$, esiste una formula ben formata X, costruita utilizzando solo i connettivi di C, tale che $f = f_X$.

In altri termini, un insieme di connettivi è funzionalmente completo se ogni altro connettivo possibile (che corrisponde ad ogni possibile tavola di verità) è derivabile da esso.

2.7 FORME NORMALI

Definizione 2.7.1. Un **letterale** è una proposizione atomica o la sua negazione.

Definizione 2.7.2. Una congiunzione di formule ben formate $X_1, X_2, \ldots, X_n$ è una formula del tipo $X_1 \wedge X_2 \wedge \ldots \wedge X_n$. La disgiunzione delle formule $X_1, X_2, \ldots, X_n$ è invece la formula $X_1 \vee X_2 \vee \ldots \vee X_n$.

Definizione 2.7.3. Una formula ben formata X è detta in **forma normale congiuntiva** (CNF) se X è della forma $X_1 \wedge X_2 \wedge \ldots \wedge X_n$ (per qualche $n \geq 1$), dove ciascuna X_i è una disgiunzione di letterali.

Esempio 2.7.4. Le due formule seguenti sono in forma normale congiuntiva:

$$A \wedge \neg B \wedge (A \vee C)(\neg A \vee B \vee C) \wedge (\neg C \vee A)$$

Osservazione 2.7.5. Un caso particolare di forma normale congiuntiva si ha quando $n = 1$. La formula $X_1 \wedge X_2 \wedge \ldots \wedge X_n$ si riduce allora

alla sola X_1,la quale è una disgiunzione di letterali. Ciò significa che una formula come la seguente

$$\neg A \vee B \vee \neg C \ ,$$

dove A, B e C sono proposizioni atomiche, è già nella sua forma normale congiuntiva!

Definizione 2.7.6. Una formula ben formata X è detta in **forma normale disgiuntiva** (DNF) se X è della forma $X_1 \vee X_2 \vee \ldots \vee X_n$ (per qualche $n \geq 1$), dove ciascuna X_i è una congiunzione di letterali.

Esempio 2.7.7. Le due formule seguenti sono in forma normale disgiuntiva:

$$A \vee (\neg B \wedge C \wedge \neg A)$$

$$(A \wedge B) \vee (C \wedge \neg A) \vee C$$

Osservazione 2.7.8. Un caso particolare di forma normale disgiuntiva si ha quando $n = 1$. La formula $X_1 \vee X_2 \vee \ldots \vee X_n$ si riduce allora alla sola X_1,la quale è una congiunzione di letterali. Ciò significa che una formula come la seguente

$$A \wedge \neg B \wedge \neg C \ ,$$

dove A, B e C sono proposizioni atomiche, è già nella sua forma normale disgiuntiva ! In particolare, ciò significa che, se $A_1, A_2, \ldots, A_n$ sono dei letterali, una formula del tipo

$$A_1 \wedge A_2 \wedge \ldots \wedge A_n$$

può essere interpretata sia come una forma normale congiuntiva, sia come una forma normale disgiuntiva. Lo stesso vale per una formula del tipo

$$A_1 \vee A_2 \vee \ldots \vee A_n \ .$$

L'importanza di tali forme normali è dovuta al seguente risultato:

Teorema 2.7.9. *Per ogni formula ben formata X esistono una formula in forma normale congiuntiva X' e una formula in forma normale disgiuntiva X'' tali che X è logicamente equivalente a X' e X è logicamente equivalente a X''.*

Dimostrazione. La dimostrazione (e la costruzione delle formule CNF e DNF a partire da X) procede nel modo seguente:

1. Si eliminano dalla formula X tutti i connettivi diversi da $\wedge, \vee$ e $\neg$ utilizzando le formule dimostrate nell'Esercizio **??** (chi non lo avesse ancora svolto lo faccia ora!).

2. Si utilizzano le leggi di De Morgan e la legge della doppia negazione per portare i simboli di negazione immediatamente davanti alle proposizioni atomiche.

3. Si utilizzano le proprietà distributive per convertire la formula così ottenuta nella forma CNF oppure DNF.

□

Esempio 2.7.10. Consideriamo la formula $X = (A \vee \neg B) \to C$. Vogliamo determinare una sua CNF:

$$\begin{aligned}(A \vee \neg B) \to C &\equiv \neg(A \vee \neg B) \vee C \\ &\equiv (\neg A \wedge \neg\neg B) \vee C \\ &\equiv (\neg A \wedge B) \vee C \\ &\equiv (\neg A \vee C) \wedge (B \vee C). \quad \text{CNF.}\end{aligned}$$

Cerchiamo ora una sua forma normale disgiuntiva DNF:

$$\begin{aligned}(A \vee \neg B) \to C &\equiv \neg(A \vee \neg B) \vee C \\ &\equiv (\neg A \wedge \neg\neg B) \vee C \\ &\equiv (\neg A \wedge B) \vee C. \quad \text{DNF.}\end{aligned}$$

Un metodo pratico per determinare una DNF di una formula ben formata X contenente le proposizioni atomiche $A_1, A_2, \ldots, A_n$, è il seguente:

1. Si costruisce la tavola di verità di X .
2. Per ogni riga in cui X ha valore di verità 1 si scrive una congiunzione, i cui letterali sono determinati come segue: se nell'interpretazione v che corrisponde alla riga in questione risulta $v(A_i) = 1$ allora viene inserito A_i come letterale, altrimenti si inserisce $\neg A_i$.
3. Tutte le formule così ottenute vanno concatenate tra loro, separandole con il connettivo $\vee$.

Un esempio dovrebbe essere sufficiente a chiarire quanto sopra detto.

Esempio 2.7.11. Supponiamo che X sia una formula la cui tavola di verità è la seguente:

A	B	C	...	X
0	0	0	...	1
0	0	1	...	0
0	1	0	...	0
0	1	1	...	0
1	0	0	...	1
1	0	1	...	1
1	1	0	...	0
1	1	1	...	0

Le righe in cui X ha valore di verità 1 sono la prima, la quinta e la sesta. La forma normale sarà data quindi dalla disgiunzione di tre formule

$$(\ldots) \vee (\ldots) \vee (\ldots) \text{ ,}$$

una per ciascuna delle tre righe menzionate. La formula corrispondente alla prima riga si ottiene nel modo seguente: dato che nella prima riga i valori di verità di A, B e C sono tutti e tre 0, dovremo scrivere una congiunzione di $\neg A, \neg B$ e $\neg C$. Si ha quindi

$$(\neg A \wedge \neg B \wedge \neg C) \vee (\dots) \vee (\dots) .$$

Passiamo ora alla quinta riga: il valore di verità di A è 1, mentre quelli di B e C sono 0. Dovremo quindi scrivere una congiunzione di $A, \neg B$ e $\neg C$. Si ha pertanto

$$(\neg A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg B \wedge \neg C) \vee (\dots) .$$

Infine, consideriamo la sesta riga. Il valore di verità di A è 1, quello di B è 0 e quello di C è 1. Dovremo quindi scrivere una congiunzione di $A, \neg B$ e C. Arriviamo così alla formula seguente:

$$(\neg A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg B \wedge \neg C) \vee (A \wedge \neg B \wedge C) .$$

Esiste un algoritmo analogo per determinare una CNF di una formula ben formata X. Tale algoritmo si ottiene semplicemente scambiando, nel procedimento descritto in precedenza, i ruoli di 0 e 1 e quelli di $\wedge$ e $\vee$. Più precisamente, si procede come segue:

1. Si costruisce la tavola di verità di X.
2. Per ogni riga in cui X ha valore di verità 0 si scrive una disgiunzione, i cui letterali sono determinati come segue: se nell'interpretazione v che corrisponde alla riga in questione risulta $v(A_i) = 0$ allora viene inserito A_i come letterale, altrimenti si inserisce $\neg A_i$.
3. Tutte le formule così ottenute vanno concatenate tra loro, separandole con il connettivo $\wedge$.

Applichiamo questo algoritmo alla formula dell'esempio precedente:

Esempio 2.7.12. Supponiamo che X sia una formula la cui tavola di verità è la seguente:

A	B	C	$\dots$	X
0	0	0	$\dots$	1
0	0	1	$\dots$	0
0	1	0	$\dots$	0
0	1	1	$\dots$	0
1	0	0	$\dots$	1
1	0	1	$\dots$	1
1	1	0	$\dots$	0
1	1	1	$\dots$	0

Le righe in cui X ha valore di verità 0 sono la seconda, la terza, la quarta, la settima e l'ottava. La sua CNF sarà quindi una congiunzione di cinque formule

$$(\dots) \wedge (\dots) \wedge (\dots) \wedge (\dots) \wedge (\dots) ,$$

una per ciascuna delle cinque righe menzionate. La formula corrispondente alla seconda riga si ottiene nel modo seguente: dato che nella seconda riga i valori di verità di A e B sono 0 mentre quello di C è 1, dovremo scrivere una disgiunzione di A, B e $\neg C$. Si ha quindi

$$(A \vee B \vee \neg C) \wedge (\dots) \wedge (\dots) \wedge (\dots) \wedge (\dots) .$$

Passiamo ora alla terza riga: il valore di verità di A è 0, quello di B è 1 e quello di C è 0. Dovremo quindi scrivere una disgiunzione di A, $\neg B$ e C. Si ha pertanto

$$(A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (\dots) \wedge (\dots) \wedge (\dots) .$$

Consideriamo la quarta riga. Il valore di verità di A è 0, quello di B è 1 e quello di C è 1. Dovremo quindi scrivere una disgiunzione di A, $\neg B$ e $\neg C$. Si ha pertanto

$$(A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge (\dots) \wedge (\dots) .$$

Consideriamo la settima riga. Il valore di verità di A è 1, quello di B è 1 e quello di C è 0. Dovremo quindi scrivere una disgiunzione di $\neg A$, $\neg B$ e C. Si ha pertanto

$$(A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge (\neg A \vee \neg B \vee C) \wedge (\dots) .$$

Infine, consideriamo l'ottava riga. Il valore di verità di A è 1, quello di B è 1 e quello di C è 1. Dovremo quindi scrivere una disgiunzione di $\neg A$, $\neg B e \neg C$. Si arriva così alla seguente formula finale:

$$\begin{aligned}(A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge \\ \wedge (\neg A \vee \neg B \vee C) \wedge (\neg A \vee \neg B \vee \neg C) .\end{aligned}$$

Osservazione 2.7.13. Le formule CNF e DNF ottenute con i due metodi appena descritti non sono necessariamente le più corte possibili.

A volte è possibile semplificarle, come nel seguente esempio. Vogliamo determinare una forma normale disgiuntiva DNF della formula X la cui tavola di verità è

A	B	C	$\dots$	X
0	0	0	$\dots$	0
0	0	1	$\dots$	0
0	1	0	$\dots$	1
0	1	1	$\dots$	0
1	0	0	$\dots$	0
1	0	1	$\dots$	0
1	1	0	$\dots$	1
1	1	1	$\dots$	0

Le righe in cui X ha valore di verità 1 sono la terza e la settima. Applichiamo il metodo descritto in precedenza. La DNF sarà dunque una disgiunzione di due formule

$$(\dots) \vee (\dots) .$$

La formula corrispondente alla terza riga si ottiene nel modo seguente: dato che nella terza riga il valore di verità di A è 0, quello di B è 1 e quello di C è 0, dovremo scrivere una congiunzione di $\neg A, B$ e $\neg C$. Si ha quindi

$$(\neg A \wedge B \wedge \neg C) \vee (\dots) \ .$$

Passiamo ora alla settima riga: il valore di verità di A è 1, quello di B è 1 e quello di C è 0. Dovremo allora scrivere una congiunzione di A, B e $\neg C$. Si ha quindi

$$(\neg A \wedge B \wedge \neg C) \vee (A \wedge B \wedge \neg C) \ .$$

Utilizzando le proprietà dei connettivi (in special modo la proprietà distributiva), questa formula può essere semplificata come segue:

$$\begin{aligned}
(\neg A \wedge B \wedge \neg C) \vee (A \wedge B \wedge \neg C) \quad &\equiv \\
&\equiv \quad (\neg A \wedge (B \wedge \neg C)) \vee \\
&\qquad\qquad\quad (A \wedge (B \wedge \neg C)) \\
&\equiv \quad (\neg A \vee A) \wedge (B \wedge \neg C) \\
&\equiv \quad B \wedge \neg C \ ,
\end{aligned}$$

dove abbiamo usato il fatto che $\neg A \vee A$ è una tautologia. Da quanto visto finora discende immediatamente il seguente risultato:

Teorema 2.7.14. *L'insieme di connettivi $\{\neg, \wedge, \vee\}$ è funzionalmente completo.*

Corollario 2.7.15. *Gli insiemi di connettivi $\{\neg, \vee\}$ e $\{\neg, \wedge\}$ sono funzionalmente completi.*

Dimostrazione. Basta osservare che $\wedge$ è derivabile da $\{\neg, \vee\}$, dato che si ha $X \wedge Y \equiv \neg(\neg X \vee \neg Y)$ e, analogamente, $\vee$ è derivabile da $\{\neg, \wedge\}$, dato che si ha $X \vee Y \equiv \neg(\neg X \wedge \neg Y)$. □

Esercizio 2.7.16. È possibile definire un connettivo binario (chiamiamolo $*$) tale che l'insieme costituito da questo unico connettivo sia funzionalmente completo?

3

CALCOLO SEMANTICO

Verificare che una formula proposizionale ben formata sia una tautologia è un compito abbastanza semplice, poiché è necessaria solo un po' di pazienza per scrivere la sua tavola di verità e verificare che nell'ultima colonna siano presenti solo 1. Tuttavia la pazienza necessaria potrebbe essere davvero tanta poiché i casi considerati nella tavola di verità sono 2^n dove n è il numero di variabili preposizionali distinte presenti nella data formula.

In questo breve capitolo presenteremo un metodo più efficace per la verifica di tautologie. L'algoritmo, che chiameremo `Tab` costruisce, a partire da una formula, un albero di possibili valutazioni per quella formula. Partendo dalla negazione della formula data, se un ramo dell'albero dovesse arrivare a conclusione positivamente, otterremmo una valutazione che falsifica la formula data; questa non sarebbe dunque una tautologia. Se viceversa l'algoritmo dovesse chiudere tutti i rami negativamente (cioè con qualche contraddizione all'interno di ogni ramo) allora vorrebbe dire che è impossibile falsificare la formula data o, equivalentemente, che essa è una tautologia.

Il primo passo, data la formula X è porla falsa, scriveremo dunque:

$$F(X)$$

I successivi passi da effettuare nell'algoritmo `Tab` dipendono, come sempre, dalla *forma* della formula data.

Se $X = X_1 \wedge X_2$ allora l'albero si prolunga come segue:

$$\begin{array}{ccc} & F(X_1 \wedge X_2) & \\ & \swarrow \quad \searrow & \\ F(X_1) & & F(X_2) \end{array}$$

Intuitivamente, se vogliamo falsificare $X_1 \wedge X_2$ basta falsificare o X_1 o X_2.

Se $X = X_1 \vee X_2$ allora l'albero si prolunga come segue:

$$\begin{array}{c} F(X_1 \vee X_2) \\ | \\ F(X_1) \\ | \\ F(X_2) \end{array}$$

Intuitivamente, se vogliamo falsificare $X_1 \vee X_2$ bisogna falsificare contemporaneamente (cioè sullo stesso ramo) sia X_1 che X_2.

Se $X = X_1 \to X_2$ allora l'albero si prolunga come segue:

$$\begin{array}{c} F(X_1 \to X_2) \\ | \\ V(X_1) \\ | \\ F(X_2) \end{array}$$

Intuitivamente, se vogliamo falsificare $X_1 \to X_2$ bisogna contemporaneamente verificare X_1 e falsificare X_2.

Se $X = \neg X_1$ allora l'albero si prolunga come segue:

$$\begin{array}{c} F(\neg X_1) \\ | \\ V(X_1) \end{array}$$

Intuitivamente, se vogliamo falsificare $\neg X_1$ bisogna verificare X_1.

Abbiamo così visto tutti quattro casi possibile per prolungare un la foglia $F(X)$. Notiamo però che facendo questo, si creano dei nodi dove al post di F c'è V. Dobbiamo quindi descrivere come procede l'algoritmo nel caso il nodo da prolungare si del tipo $V(X)$. Abbiamo di nuovo quattro casi possibili.

Se $X = X_1 \wedge X_2$ allora l'albero si prolunga come segue:

$$\begin{array}{c} V(X_1 \wedge X_2) \\ | \\ V(X_1) \\ | \\ V(X_2) \end{array}$$

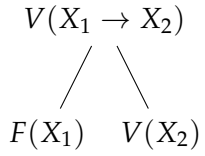
Intuitivamente, se vogliamo verificare $X_1 \wedge X_2$ dobbiamo verificare contemporaneamente X_1 e X_2.

Se $X = X_1 \vee X_2$ allora l'albero si prolunga come segue:

$$\begin{array}{ccc} & V(X_1 \vee X_2) & \\ \swarrow & & \searrow \\ V(X_1) & & V(X_2) \end{array}$$

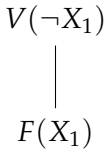
Intuitivamente, se vogliamo verificare $X_1 \vee X_2$ bisogna verificare almeno uno tra X_1 e X_2.

Se $X = X_1 \to X_2$ allora l'albero si prolunga come segue:



Intuitivamente, se vogliamo verificare $X_1 \to X_2$ possiamo o falsificare l'ipotesi, cioè X_1 o verificare la tesi, cioè X_2.

Se $X = \neg X_1$ allora l'albero si prolunga come segue:



Intuitivamente, se vogliamo verificare $\neg X_1$ bisogna falsificare X_1.

Parte II

LOGICA DEL PRIMO ORDINE

4

SINTASSI E SEMANTICA

Il calcolo proposizionale che abbiamo presentato è poco espressivo. Manca la possibilità di esprimere che una certa proprietà P vale per tutti gli elementi di un certo insieme, oppure che esiste almeno un elemento che gode della proprietà P. Se x e y sono elementi di un dato insieme, scriveremo $P(x)$ per indicare che la proprietà P vale per l'elemento x o $R(x, y)$ per indicare che l'elemento x è in relazione R con l'elemento y. Ad esempio, $P(x)$ potrebbe essere l'asserzione seguente: "x è rosso," dove x denota un qualsiasi oggetto, oppure la relazione $R(x, y)$ potrebbe indicare che x è più vecchio di y, dove x e y sono elementi nell'insieme degli esseri umani. Grazie a questo linguaggio più espressivo, possiamo ora studiare ragionamenti come il seguente, che non sono approcciabili con il metodo della logica proposizionale.

$$\frac{\begin{array}{c}\text{Anna è moglie di Nicola,}\\ \\ \text{Se una qualsiasi persona è moglie di un qualsiasi altro,}\\ \text{allora quell'altro è marito di questa persona,}\end{array}}{\text{Nicola è marito di Anna.}}$$

Con la nostra nuova notazione potremmo indicare con $R(x, y)$ che x è marito di y e con $Q(x, y)$ che x è moglie di y, dunque avremmo:

$$\frac{\begin{array}{c}Q(\text{Anna, Nicola}),\\ Q(x, y) \to R(y, x)\end{array}}{R(\text{Nicola, Anna}).}$$

La struttura logica del ragionamento comincia a diventare più evidente. Quello che ancora manca è il poter esprimere la parola *qualsiasi* del ragionamento iniziale. Introdurremo due nuovi simboli, che indicheremo con $\forall$ e $\exists$, a cui attribuiremo i seguenti significati:

1. $\forall x P(x)$ servirà ad indicare che la proprietà P vale per ogni elemento x (di un qualche insieme prefissato),

2. $\exists x P(x)$ servirà ad indicare che la proprietà P vale per qualche x, cioè che esiste almeno un elemento x per cui $P(x)$ è vera.

I due simboli $\forall$ e $\exists$ si chiamano, rispettivamente, **quantificatore universale** e **quantificatore esistenziale**. Si tratterà quindi di estendere il linguaggio del calcolo proposizionale aggiungendo questi due nuovi simboli, assieme a tutta una serie di altri "oggetti" di cui avremo bisogno.

4.1 IL LINGUAGGIO DEL CALCOLO DEI PREDICATI

Definiamo ora la sintassi che ci servirà a manipolare espressioni logiche contenenti i due quantificatori $\forall$ e $\exists$.

Definizione 4.1.1. Un **linguaggio del primo ordine** è un insieme $\mathcal{L}$ composto da:

1. un insieme (anche vuoto) di simboli di costante: $a, b, c, \ldots$ (oppure $a_1, a_2, a_3, \ldots$);
2. un insieme infinito di simboli di variabile: $x, y, z, \ldots$ (oppure $x_1, x_2, x_3, \ldots$);
3. un insieme (anche vuoto) di simboli di funzione: $f, g, h, \ldots$ (oppure $f_1, f_2, f_3, \ldots$);
4. un insieme (anche vuoto) di simboli di predicato: $A, B, C, \ldots$ (oppure $A_1, A_2, A_3, \ldots$);
5. i connettivi proposizionali: $\neg, \wedge, \vee, \rightarrow$;
6. il simbolo $\bot$;
7. i simboli per i quantificatori: $\forall, \exists$;
8. i simboli ausiliari: parentesi e virgole;

Ad ogni simbolo di funzione f deve essere associato un numero naturale n (detto **arietà** di f, che indicheremo all'apice $f^{(n)}$. Analogamente, ad ogni simbolo di predicato A deve essere associato un numero naturale n (detto anch'esso arietà del predicato); ancora una volta useremo il simbolo $A^{(n)}$.

Di solito si richiede che i simboli funzione e di predicato siano in numero finito, ma molti dei risultati che stabiliremo valgono anche se questi insiemi sono infiniti. L'insieme dei simboli di costante può avere cardinalità arbitraria. L'arietà dei simboli di funzione e di predicato sta a indicare il corretto numero di argomenti da associare f. Essa ci permetterà più avanti di definire i termini e le formule ben formate. Inoltre nella interpretazione semantica, assoceremo al simbolo $f^{(n)}$ una funzione a n variabili e assoceremo al simbolo $A^{(m)}$ un predicato m-ario.

Fissato un linguaggio $\mathcal{L}$ possiamo definire termini e formule in quel linguaggio.

Definizione 4.1.2. Definiamo ora l'insieme *Term* dei **termini** stabilendo che una stringa di sibili t nel linguaggio $\mathcal{L}$:

1. se $t = c$ per qualche costante, allora t è un termine;
2. se $t = x$ per qualche variabile x, allora t è un termine;
3. Se $t = f^{(n)}(t_1, t_2, \ldots, t_n)$ se $t_1, t_2, \ldots, t_n$ sono dei termini e $f^{(n)}$ è un simbolo di funzione allora t è un termine.

4. Un oggetto è un termine, solo se è costruito mediante applicazioni ripetute delle tre regole sopra.

Possiamo ora definire le formule atomiche.

Definizione 4.1.3. Una stringa X di simboli in un linguaggio del primo ordine $\mathcal{L}$ è detta **formula atomica** se e solo se vale una delle seguenti regole

1. se $X = \bot$ allora X è una formula atomica;
2. se $X = A(n)(t_1, t_2, \ldots, t_n)$, $t_1, t_2, \ldots, t_n$ sono dei termini e $A^{(n)}$ è un simbolo di predicato, allora X è una formula atomica.

Anche in questo caso le formule atomiche sono tutte e sole quelle formule che si possono costruire mediante l'applicazione delle regole precedenti.

Siamo finalmente in grado di definire le formule ben formate.

Definizione 4.1.4. Una stringa di simboli X in un linguaggio del primo ordine $\mathcal{L}$ è detta **formula ben formata** (**FBF**) se vale una delle seguenti

1. X è una formula atomica,
2. $X = (\neg P)$ e P è una FBF;
3. $X = (P \wedge Q)$ e P e Q sono FBF;
4. $X = (P \vee Q)$ e P e Q sono FBF;
5. $X = (P \to Q)$ e P e Q sono FBF;
6. $X = (\forall x P)$ e P è una FBF;
7. $X = (\exists x P)$ e P è una FBF.

Esempio 4.1.5. Le seguenti sono FBF:

$$(\exists x((P(x) \wedge Q) \to (\forall y R(y))))$$

$$(\forall x(\exists y(P(x, f(c, y)) \vee Q(y, c))))$$

Al contrario, la seguente sequenza di simboli non è una FBF:

$$\forall x \exists \to (\wedge P \forall$$

Osservazione 4.1.6. Le regole che abbiamo stabilito conducono ad un uso eccessivo delle parentesi. Come nel calcolo proposizionale, conviene dunque stabilire delle priorità tra i vari simboli:

priorità più alta	$\forall, \exists, \neg,$
	$\wedge,$
	$\vee,$
priorità più bassa	$\to$.

Si noti che $\forall, \exists$ e $\neg$ hanno la stessa priorità.

La formula

$$(\forall x(P(x) \to ((\exists y Q(x,y)) \vee (\neg P(x)))))$$

si potrà dunque scrivere come segue:

$$\forall x(P(x) \to \exists y Q(x,y) \vee \neg P(x)) \ .$$

Si noti che l'ultima coppia di parentesi rimasta non si può togliere. Infatti, la formula

$$\forall x P(x) \to \exists y Q(x,y) \vee \neg P(x)$$

verrebbe interpretata come

$$((\forall x P(x)) \to ((\exists y Q(x,y)) \vee (\neg P(x)))) \ .$$

4.2 VARIABILI LIBERE E VARIABILI LEGATE.

In una formula del tipo $\forall x P$ (oppure $\exists x P$) la variabile x si dice legata. Ogni quantificatore introduce un legame con le variabili presenti nel suo campo d'azione, ove per campo d'azione di un quantificatore si intende la formula ben formata a cui "si applica" il quantificatore stesso durante la costruzione della formula . Ad esempio, nella formula

$$\exists x(P \to Q) \to R \wedge Q$$

il campo d'azione di $\exists x$ è la formula $P \to Q$. In tale formula la variabile x è dunque legata. Una variabile si dice libera se non è legata. Ad esempio, nella formula $\forall x A(x,y) \vee \neg B(y,z)$, la variabile x è legata, mentre le variabili y e z sono libere. Indicheremo con $FV(P)$ l'insieme delle variabili libere (free variables) della formula ben formata P. Esso può essere definito in modo rigoroso come segue:

Definizione 4.2.1. Se t è un termine, l'insieme $FV(t)$ delle **variabili libere** di t è definito induttivamente mediante le seguenti regole:

1. $FV(c) = \emptyset$, per ogni costante c;
2. $FV(x) = \{x\}$, per ogni variabile x;
3. $FV(f^{(n)}(t_1, \ldots, t_n)) = FV(t_1) \cup \ldots \cup FV(t_n)$, per ogni funzione n-aria $f^{(n)}$ (e, naturalmente, per ogni n).

Definizione 4.2.2. Se X è una FBF, l'insieme $FV(X)$ è definito induttivamente mediante le seguenti regole:

1. $FV(\bot) = \emptyset$;
2. $FV(A^{(n)}(t_1, \ldots, t_n)) = FV(t1) \cup \Delta\Delta\Delta \cup FV(t_n)$, per ogni predicato n-ario $A^{(n)}$ (e, naturalmente, per ogni n);
3. $FV(\neg X) = FV(X)$, per ogni formula ben formata X;
4. $FV(X \wedge Y) = FV(X) \cup FV(Y)$, per ogni X e Y;

5. $FV(X \vee Y) = FV(X) \cup FV(Y)$, per ogni X e Y;
6. $FV(X \to Y) = FV(X) \cup FV(Y)$, per ogni X e Y;
7. $FV(\forall x X) = FV(X) \setminus \{x\}$, per ogni X;
8. $FV(\exists x X) = FV(X) \setminus \{x\}$, per ogni X.

Definizione 4.2.3. Una formula ben formata X è detta **chiusa** se essa non contiene variabili libere, cioè se $FV(X) = \emptyset$. In caso contrario essa è detta aperta.

Indicheremo con $BV(X)$ l'insieme delle variabili legate (bound variables) della formula X. Facciamo subito notare che, per una formula ben formata X, è possibile avere

$$FV(X) \cap BV(X) \neq \emptyset \,,$$

cioè una variabile può comparire all'interno di una formula X sia come variabile libera che come variabile legata. Ad esempio, se indichiamo con X la seguente formula

$$\forall x(Q(x,y) \to R(x)) \wedge \forall y(\neg Q(x,y) \to \forall z R(z)) \,,$$

si ha $FV(P) = \{y, x\}$ e $BV(P) = \{x, y, z\}$. Tuttavia, ogni occorrenza di una data variabile all'interno di una formula ben formata è libera oppure legata.

Osservazione 4.2.4. Può accadere che una variabile compaia nel campo d'azione di più quantificatori, come, ad esempio, nella formula

$$\forall x(A(x) \to \forall x B(x)) \,,$$

dove la seconda occorrenza di x (cioè quella in $B(x)$) si trova nel campo d'azione di entrambi i quantificatori. In questo caso si assume che il legame corretto sia con il quantificatore più interno.

4.3 SOSTITUZIONI.

Anche se non abbiamo ancora parlato dell'interpretazione di una FBF contenente dei quantificatori, ricordiamo che il simbolo $\forall$ è stato introdotto allo scopo di poter esprimere il fatto che una determinata proprietà valga per tutti gli elementi di un dato insieme: $\forall x P(x)$ dovrà dunque significare che la proprietà P vale per tutti gli elementi di un prefissato insieme. Da ciò segue che il fatto di aver indicato la variabile quantificata con la lettera x è del tutto irrilevante; avremmo anche potuto scrivere $\forall y P(y)$, oppure $\forall z P(z)$, ecc. Un discorso del tutto analogo vale anche per il quantificatore esistenziale $\exists$. E' dunque naturale assumere che nel nostro linguaggio valga la seguente regola:

> *ogni variabile che compare nel campo d'azione di un quantificatore può essere sostituita con un'altra, a patto che quest'ultima non compaia come variabile libera all'interno della stessa formula.*

Cerchiamo di chiarire con un esempio quest'ultima richiesta. Se le variabili x e y rappresentano dei numeri naturali, la formula

$$\exists x(x > y)$$

esprime il fatto che, dato un numero intero y, esiste un qualche numero intero x che sia maggiore di y (il che, in effetti, è vero). In base a quanto detto, noi possiamo sostituire ogni occorrenza della variabile quantificata x con un'altra variabile, ad esempio z, ottenendo la formula

$$\exists z(z > y)$$

che esprime esattamente la stessa affermazione. Non possiamo invece sostituire la variabile quantificata x con la variabile y, dato che y compare come variabile libera nella formula $x > y$. Se lo facessimo otterremmo infatti la formula

$$\exists y(y > y)$$

la quale affermerebbe che esiste un qualche numero intero che è maggiore di sé stesso (il che è falso).

Introduciamo ora una notazione adeguata per indicare la sostituzione all'interno di una formula.

Definizione 4.3.1. Siano R e P due formule ben formate e sia X una formula atomica. Indicheremo con $R[P/X]$ la formula ben formata ottenuta rimpiazzando tutte le occorrenze di X in R con la formula P.

Esempio 4.3.2. Se $R = \neg A \wedge B \to B \vee A$, $P = \neg B \vee C$ e $X = A$, allora si ha

$$R[P/A] = \neg(\neg B \vee C) \wedge B \to B \vee (\neg B \vee C)\ .$$

Definiamo ora, in modo formale, la sostituzione all'interno dei termini.

Definizione 4.3.3. Siano s e t due termini e x una variabile. Il termine $s[t/x]$ è definito mediante le seguenti regole:

1. se $s = c$ per qualche costante c, poniamo $c[t/x] = c$;
2. se $s = y$ per qualche variabile y, allora si pone

$$y[t/x] = \begin{cases} y & \text{se } x \neq y \\ t & \text{se } x = y \end{cases}$$

3. se $t_1, t_2, \ldots, t_n$ sono termini e $f^{(n)}$ è una funzione, allora poniamo $f^{(n)}(t_1, \ldots, t_n)[t/x] = f^{(n)}(t_1[t/x], \ldots, t_n[t/x])$.

Passiamo ora a definire la sostituzione di termini all'interno delle formule ben formate. Come già accennato in precedenza, in questo

caso si presentano dei problemi. Infatti, se P è una formula ben formata in cui compaiono una variabile libera x e una variabile legata y, mentre t è un termine in cui la y compare come variabile libera, operando direttamente la sostituzione di tutte le x per t in P, la variabile y che compariva in t come variabile libera diventerebbe legata.

Possiamo illustrare questo fatto con il seguente esempio. Consideriamo la formula $P = \exists y(x < y)$, in cui x compare come variabile libera e y come variabile legata. Se consideriamo il termine $t = z$, in cui non compare la variabile y, non incontriamo nessun problema: la formula $P[z/x]$ è $\exists y(z < y)$, che esprime la stessa asserzione della formula P. Al contrario, se consideriamo il termine $t = y$, in cui compare proprio la y come variabile libera, si ottiene la formula $P[y/x] = \exists y(y < y)$, la quale ha un "significato logico" diverso. Questo problema può tuttavia essere risolto ridenominando opportunamente la variabile legata y all'interno della formula P prima di operare la sostituzione del termine t al posto della variabile x. In questo modo si ha, infatti,

$$\begin{aligned} P[z/y][y/x] =& (\exists y(x < y))[z/y][y/x] = \\ =& (\exists z(x < z))[y/x] = \\ =& (\exists z(y < z)) \,, \end{aligned}$$

la quale ha, intuitivamente, lo stesso significato della formula P. Chiarito questo aspetto, diamo ora la definizione formale:

Definizione 4.3.4. Siano P una formula ben formata, t un termine e x una variabile. La formula $P[t/x]$ è definita mediante le seguenti regole:

1. $\bot[t/x] = \bot$;
2. se $t_1, t_2, \ldots, t_n$ sono dei termini e $A^{(n)}$ è un simbolo di predicato, allora $A^{(n)}(t_1, \ldots, t_n)[t/x] = A^{(n)}(t_1[t/x], \ldots, t_n[t/x])$;
3. se P e Q sono FBF, allora $(\neg P)[t/x] = \neg(P[t/x])$, $(P \wedge Q)[t/x] = (P[t/x] \wedge Q[t/x])$, $(P \vee Q)[t/x] = (P[t/x] \vee Q[t/x])$ e $(P \to Q)[t/x] = (P[t/x] \to Q[t/x])$;
4. se P è una FBF, allora

$$\forall y(P)[t/x] = \begin{cases} \forall y(P[t/x]) & \text{se } x \neq y \text{ e } y \notin FV(t), \\ \forall w(P[w/y][t/x]) & \text{se } x \neq y \text{ e } y \in FV(t), \\ \forall y(P) & \text{se } x = y, \end{cases}$$

   dove w è una variabile *nuova*, cioè che non compare né in P né in t.

5. se P è una FBF, allora

$$\exists y(P)[t/x] = \begin{cases} \exists y(P[t/x]) & \text{se } x \neq y \text{ e } y \notin FV(t), \\ \exists w(P[w/y][t/x]) & \text{se } x \neq y \text{ e } y \in FV(t), \\ \exists y(P) & \text{se } x = y, \end{cases}$$

dove w è una variabile *nuova*, cioè che non compare né in P né in t.

Osservazione 4.3.5. Se $t_1, t_2, \ldots, t_n$ sono dei termini e $x_1, x_2, \ldots, x_n$ sono delle variabili, indicheremo con il simbolo $P[t_1, \ldots, t_n / x_1, \ldots, x_n]$ la sostituzione simultanea all'interno della formula P . In generale il suo effetto è diverso dalle corrispondenti sostituzioni iterate. Si ha infatti

$$P(x,y)[y,x/x,y] = P(y,x)$$

mentre

$$P(x,y)[y,x/x,y] = P(y,x), P(x,y)[y/x][x/y] = P(y,y)[x/y] = P(x,x)\,.$$

4.4 SEMANTICA

Ci poniamo ora il problema di attribuire un significato a tutte le formule ben formate del nostro linguaggio. Se nel caso del Calcolo Proposizionale ciò era piuttosto facile (ricordiamo che si trattava solo di stabilire le tavole di verità dei connettivi proposizionali), nel caso del Calcolo dei Predicati l'attribuzione di un valore di verità ad una formula ben formata risulta complicata a causa della presenza di costanti, variabili, funzioni, ecc. Innanzitutto dovremo fissare un dominio, cioè un insieme D, nel quale assumeranno i loro valori i simboli di costante e i simboli di variabile. Poi dovremo far corrispondere ad ogni simbolo di funzione $f^{(n)}$ una funzione vera e propria da D^n in D (cioè una funzione di n variabili a valori nel dominio D). Infine, ad ogni simbolo di predicato $A^{(n)}$ dovremo far corrispondere una relazione n-aria sull'insieme D, cioè una funzione definita su D^n a valori nell'insieme $\{0,1\}$ (VERO o FALSO). Cominciamo quindi col dare la seguente definizione.

Definizione 4.4.1. Una **dominio di interpretazione** (detto anche **struttura**) $\mathfrak{A}$ è dato da un insieme non vuoto D, detto dominio e da un assegnamento che associa:

1. a ogni simbolo di costante c un elemento $c^{\mathfrak{A}} \in D$;
2. a ogni simbolo di funzione $f^{(n)}$ una funzione $f^{\mathfrak{A}} \colon D^n \to D$;
3. a ogni simbolo di predicato $B^{(n)}$ una relazione n-aria $B^{\mathfrak{A}} \colon D^n \to \{0,1\}$ o in altre parole, un sottoinsieme $B^{\mathfrak{A}} \subseteq D^n$. [1]

Notiamo che, in generale, non è possibile assegnare un valore di verità ad una formula che contenga delle variabili libere; sarà prima necessario assegnare a tali variabili degli elementi del dominio. Di

1 Si noti infatti che in generale i sottoinsiemi di un insieme A sono in corrispondenza biunivoca con le funzioni da A in $\{0,1\}$. Infatti a un insieme $S \subseteq A$ possiamo associare una funzione $\chi_S \colon A \to \{0,1\}$ tale che $\chi_S(x) = 1$ se e soltanto se $x \in S$. Viceversa, a ogni funzione $f \colon A \to \{0,1\}$ possiamo associare il sottoinsieme di A definito come $\{x \in A \mid f(x) = 1$. La funzione χ_S è detta la **funzione caratteristica** di S.

conseguenza il valore di verità di una formula dipenderà, in generale, dallo specifico assegnamento scelto. Diamo dunque la seguente definizione:

Definizione 4.4.2. Data una struttura $\mathfrak{A}$ con dominio D, chiameremo **interpretazione delle variabili** (o a volte anche **ambiente** dall'inglese *environment*) una funzione $\xi\colon Var \to D$, quindi definita sull'insieme Var delle variabili e con valori in D (una tale funzione assegna dunque ad ogni variabile un elemento del dominio). Indicheremo con

$$ENV_D = \{\xi \mid \xi\colon Var \to D\}$$

l'insieme di tutti gli ambienti per la struttura $\mathfrak{A}$. Se ξ è un ambiente per una struttura $\mathfrak{A}$ e $a \in D$, indicheremo con $\xi[a/x]$ l'interpretazione delle varibili ξ modificata in modo da associare alla variabile x l'elemento a. Più precisamente, si ha:

$$\xi[a/x](y) = \begin{cases} \xi(y) & \text{se } y \neq x \\ a & \text{se } y = x \end{cases}$$

Definizione 4.4.3. Una **interpretazione** $\mathfrak{I} = (\mathfrak{A}, \xi)$ è data da una struttura $\mathfrak{A}$ e di un ambiente ξ per tale struttura. Se a è un elemento di D, scriveremo $\mathfrak{I}[a/x]$ per indicare l'interpretazione data dalla struttura $\mathfrak{A}$ e dall'interpretazione delle variabili $\xi[a/x]$.

Definiamo ora il valore di un termine in una data interpretazione:

Definizione 4.4.4. Si consideri un'interpretazione $\mathfrak{I} = (\mathfrak{A}, \xi)$. Estendiamo l'interpretazione delle variabili ξ a una funzione $\hat{\xi}$ su tutto l'insieme dei termini *Term* come segue. Dato un qualsiasi termine t definiamo:

1. se $t = c$ per qualche costante c, si ha $\hat{\xi}(t) = c^{\mathfrak{A}}$;
2. se $t = x$ per qualche variabile x, si ha $\hat{\xi}(t) = \xi(x)$;
3. se $t = f^{(n)}(t_1, \ldots, t_n)$, ove $t_1, \ldots, t_n$ sono termini, si ha $\hat{\xi}(t) = f^{\mathfrak{A}}(\hat{\xi}(t_1), \ldots, \hat{\xi}(t_n))$.

Possiamo ora definire formalmente il valore di verità di una formula ben formata P nell'interpretazione $\mathfrak{I} = (\mathfrak{A}, \xi)$, il quale verrà indicato con $v^{\mathfrak{I}}(P)$.

Definizione 4.4.5. La funzione di valutazione $v^{\mathfrak{I}}\colon FBF \to \{0,1\}$ è definita induttivamente come segue:

1. $v^{\mathfrak{I}}(\bot) = 0$;
2. $v^{\mathfrak{I}}(P(t_1, \ldots, t_n)) = P^A(\xi(t_1), \ldots, \xi(t_n))$;
3. $v^{\mathfrak{I}}(\neg P) = 1 - v^{\mathfrak{I}}(P)$;
4. $v^{\mathfrak{I}}(P \wedge Q) = \min v^{\mathfrak{I}}(P), v^{\mathfrak{I}}(Q)$;
5. $v^{\mathfrak{I}}(P \vee Q) = \max v^{\mathfrak{I}}(P), v^{\mathfrak{I}}(Q)$;

6. $v^{\mathfrak{I}}(P \to Q) = \max 1 - v^{\mathfrak{I}}(P), v^{\mathfrak{I}}(Q)$;
7. $v^{\mathfrak{I}}(\forall x P) = \min\{v^{\mathfrak{I}}[a/x](P) \mid a \in D\}$;
8. $v^{\mathfrak{I}}(\exists x P) = \max\{v^{\mathfrak{I}}[a/x](P) \mid a \in D\}$.

Osservazione 4.4.6. Si noti che le definizioni date ai punti 3,4,5 e 6 dei valori di verità delle formule costruite mediante l'uso dei connettivi $\neg, \wedge, \vee e \to$, sono equivalenti a quelle date nel Calcolo Proposizionale mediante l'uso delle tavole di verità.

Osservazione 4.4.7. Si noti che il quantificatore universale $\forall$ può essere pensato come una "congiunzione iterata". $\forall x P(x)$ ha lo stesso significato di $\bigwedge_{a \in D} P(a)$. Analogamente, il quantificatore esistenziale $\exists$ può essere pensato come una "disgiunzione iterata". $\exists x P(x)$ ha lo stesso significato di $\bigvee_{a \in D} P(a)$.

Osservazione 4.4.8. Al contrario di quanto avviene nel Calcolo Proposizionale, le definizioni che abbiamo dato non consentono, in generale, di determinare in modo effettivo il valore di verità di una formula ben formata P in una data interpretazione $\mathfrak{I}$. Infatti, se il dominio D è un insieme infinito e se la formula P contiene dei quantificatori, per determinare il valore di verità $v^{\mathfrak{I}}(P)$ sarebbe necessario calcolare il valore di verità delle infinite formule che si ottengono da P sostituendo alle variabili quantificate gli infiniti elementi di D.

Possiamo, infine, notare che il valore di verità di una formula ben formata P in una data interpretazione $\mathfrak{I} = (\mathfrak{A}, \xi)$ dipende solo dalla restrizione di ξ all'insieme delle variabili libere di P. Si ha infatti:

Teorema 4.4.9. *Siano P una formula ben formata e $\mathfrak{A}$ una struttura. Sia $FV(P) = \{y_1, y_2, \ldots, y_n\}$ l'insieme delle variabili libere di P . Allora, per tutti gli ambienti ξ_1 e ξ_2 per A tali che $\xi_1(y_i) = \xi_2(y_i)$, per ogni $i = 1, \ldots, n$, si ha $v^{(\mathfrak{A},\xi_1)}(P) = v^{(\mathfrak{A},\xi_2)}(P)$.*

Dimostrazione. La dimostrazione procede per induzione sulla struttura della formula ben formata P. □

4.5 SODDISFACIBILITÀ, VALIDITÀ E MODELLI

Definizione 4.5.1. Sia P una formula ben formata. Diremo che:

1. P è **soddisfatta** in una struttura $\mathfrak{A}$ rispetto all'ambiente ξ se $v^{(\mathfrak{A},\xi)}(P) = 1$. In tal caso scriveremo $(\mathfrak{A}, \xi) \models P$.
2. P è **soddisfacibile** in una struttura $\mathfrak{A}$ se esiste un ambiente ξ tale che $v^{(\mathfrak{A},\xi)}(P) = 1$.
3. P è **vera** in una struttura $\mathfrak{A}$ se per ogni ambiente ξ si ha $v^{(\mathfrak{A},\xi)}(P) = 1$. In tal caso diremo che $\mathfrak{A}$ è un modello per P, e scriveremo $\mathfrak{A} \models P$.
4. P è **soddisfacibile** se esistono una struttura $\mathfrak{A}$ ed un ambiente ξ tale che $v(\mathfrak{A}, \xi)(P) = 1$.

5. P è **valida** se essa è vera in ogni struttura. In tal caso scriveremo $\models P$ e diremo che P è una **tautologia** .

Osservazione 4.5.2. Il concetto di tautologia del primo ordine e tautologia proposizionale sono molto simili. Infatti nella logica proposizionale abbiamo definito tautologia una qualsiasi formula che avesse solo 1 nell'ultima colonna della sua tavola di verità, in altre parole una formula che fosse vera secondo ogni possibile interpretazione; esattamente come stiamo facendo qui per il primo ordine.

Definizione 4.5.3. Sia Γ un insieme di formule ben formate. Diremo che:

1. Γ è **soddisfacibile** se esistono una struttura $\mathfrak{A}$ ed un ambiente ξ tali che, per ogni formula $P \in \Gamma$, si abbia $v^{(\mathfrak{A},\xi)}(P) = 1$.

2. Una struttura $\mathfrak{A}$ è un **modello** per Γ se, per ogni $P \in \Gamma$, si ha $\mathfrak{A} \models P$. In tal caso scriveremo $\mathfrak{A} \models \Gamma$.

3. Γ è **valido** se ogni struttura è un modello per Γ . In tal caso scriveremo $\models \Gamma$.

Definizione 4.5.4. Sia P una formula ben formata. Diremo che

1. P è **falsa** in una struttura $\mathfrak{A}$ se essa non è soddisfacibile in $\mathfrak{A}$, cioè se non esiste alcun ambiente ξ tale che $v^{(\mathfrak{A},\xi)}(P) = 1$. In tal caso scriveremo $A \not\models P$.

2. P è **insoddisfacibile** (o **contraddittoria**) se non è soddisfacibile, cioè se essa è falsa in ogni struttura.

Diamo ora la generalizzazione della nozione di conseguenza semantica.

Definizione 4.5.5. Dato un insieme di formule ben formate Γ ed una formula Q, diremo che Q è conseguenza semantica di Γ, e scriveremo $\Gamma \models Q$, se per ogni struttura $\mathfrak{A}$ ed ogni ambiente ξ per i quali si abbia $v^{(\mathfrak{A},\xi)}(P) = 1$ per ogni $P \in \Gamma$, risulta anche $v^{(\mathfrak{A},\xi)}(Q) = 1$.

Dalle definizioni precedenti segue immediatamente il seguente risultato:

Teorema 4.5.6. *Siano Γ un insieme di FBF e P una formula ben formata. Allora:*

1. *P è valida se e solo se $\neg P$ è insoddisfacibile.*

2. *P è soddisfacibile se e solo se $\neg P$ non è valida.*

3. *$\Gamma \models P$ se e solo se $\Gamma \cup \neg P$ è insoddisfacibile.*

Osservazione 4.5.7. Notiamo che affermare che una formula P non è valida non implica che essa sia contraddittoria (cioè che $\neg P$ sia valida), ma solo che $\neg P$ è soddisfacibile.

Studiamo ora alcune proprietà della relazione di soddisfacibilità.

Definizione 4.5.8. Sia P una formula ben formata e $FV(P) = \{x_1, ..., x_n\}$ l'insieme delle variabili libere di P. La **chiusura universale** di P è la formula $Cl(P) = \forall x_1, \ldots, x_n P$, mentre la **chiusura esistenziale** di P è la formula $Ex(P) = \exists x_1, \ldots, x_n P$.

Osservazione 4.5.9. Un'espressione del tipo $\forall x_1, ..., x_n P$ è solo una forma abbreviata per indicare la formula

$$\forall x_1 (\ldots (\forall x_{n-1} (\forall x_n P)) \ldots) .$$

Una considerazione del tutto analoga vale anche per il quantificatore $\exists$.

Valgono i seguenti risultati, di cui omettiamo le dimostrazioni:

Teorema 4.5.10. *Sia P una formula ben formata. Allora P è valida se e solo se $Cl(P)$ lo è.*

Teorema 4.5.11. *Sia P una formula ben formata. Allora P è soddisfacibile se e solo se $Ex(P)$ lo è.*

Osservazione 4.5.12. In generale, verificare la validità di una formula in modo semantico è complicato. Si tratta infatti di provare che tale formula vale per ogni possibile interpretazione. Al contrario, la semantica è molto utile per dimostrare la non validità di una formula P; basta infatti esibire una interpretazione che non la soddisfa.

4.6 EQUIVALENZA SEMANTICA

Definizione 4.6.1. Due formule ben formate P e Q sono **logicamente equivalenti** (o **semanticamente equivalenti** o più semplicemente **equivalenti** se, per tutte le interpretazioni $\mathfrak{I} = (\mathfrak{A}, \xi)$ si ha $v^{\mathfrak{I}}(P) = v^{\mathfrak{I}}(Q)$. Per indicare che P e Q sono semanticamente equivalenti scriveremo $P \equiv Q$.

Osservazione 4.6.2. Possiamo notare che due formule P e Q sono semanticamente equivalenti se e solo se $\models P \leftrightarrow Q$.

È possibile dimostrare che ridenominando una variabile quantificata all'interno del campo d'azione di un quantificatore si ottiene una nuova formula semanticamente equivalente a quella data (naturalmente a condizione che il nuovo simbolo di variabile non compaia già all'interno della formula in questione). Più precisamente, si ha:

Teorema 4.6.3. *Siano P una formula ben formata e z un simbolo di variabile che non compare in P. Allora si hanno le seguenti equivalenze:*

1. $\exists x P \equiv \exists z (P[z/x])$,
2. $\forall x P \equiv \forall z (P[z/x])$.

Elenchiamo adesso, senza dimostrazione, una serie di utili risultati:

Teorema 4.6.4. *Sia P una formula ben formata. Si hanno le seguenti equivalenze:*

1. $\neg\forall xP \equiv \exists x\neg P$;
2. $\neg\exists xP \equiv \forall x\neg P$;
3. $\forall xP \equiv \neg\exists x\neg P$;
4. $\exists xP \equiv \neg\forall x\neg P$.
5. $\forall x\forall yP \equiv \forall y\forall xP$;
6. $\exists x\exists yP \equiv \exists y\exists xP$;
7. $\forall xP \equiv P$, *se* $x \notin FV(P)$;
8. $\exists xP \equiv P$, *se* $x \notin FV(P)$.

Osservazione 4.6.5. Si noti però che, in generale,

$$\forall x\exists yP \not\equiv \exists y\forall xP \ .$$

Teorema 4.6.6. *Siano P_1 e P_2 due formule ben formate. Si hanno le seguenti equivalenze:*

1. $\forall x(P_1 \wedge P_2) \equiv \forall xP_1 \wedge \forall xP_2$;
2. $\exists x(P_1 \vee P_2) \equiv \exists xP_1 \vee \exists xP_2$;
3. $\forall x(P_1 \vee P_2) \equiv \forall xP_1 \vee P_2$, *se* $x \notin FV(P_2)$;
4. $\exists x(P_1 \wedge P_2) \equiv \exists xP_1 \wedge P_2$, *se* $x \notin FV(P2)$.

Osservazione 4.6.7. Si faccia tuttavia molta attenzione perché, in generale, si ha:

1. $\forall x(P_1 \vee P_2) \not\equiv \forall xP_1 \vee \forall xP_2$;
2. $\exists x(P_1 \wedge P_2 \not\equiv \exists xP_1 \wedge \exists xP_2$.

In generale, se vogliamo spostare i quantificatori attraverso i connettivi $\wedge$ e $\vee$, sarà necessario rinominare in modo opportuno le variabili:

Teorema 4.6.8. *Siano P_1 e P_2 due formule ben formate. Indichiamo con q_1 e q_2 due quantificatori. Allora si ha:*

$q_1xP_1 \vee q_2xP_2 \equiv q1xq_2z(P_1 \vee P_2[z/x])$,

$q_1xP_1 \wedge q_2xP_2 \equiv q_1xq_2z(P_1 \wedge P_2[z/x])$, *ove* $q_1, q_2 \in \{\forall, \exists\}$ *e* $x \notin FV(P_1) \cup FV(P_2)$. *Infine, vale anche il seguente risultato:*

Teorema 4.6.9. *Siano P e Q due formule ben formate. Se $x \notin FV(Q)$, si ha:*

1. $\forall xP \to Q \equiv \exists x(P \to Q)$;
2. $\exists xP \to Q \equiv \forall x(P \to Q)$;
3. $Q \to \exists xP \equiv \exists x(Q \to P)$;
4. $Q \to \forall xP \equiv \forall x(Q \to P)$.

Dimostrazione. Dimostriamo la prima equivalenza:

$$\forall x P \to Q \equiv (\neg \forall x P) \vee Q \equiv (\exists x \neg P) \vee Q \equiv \exists x (\neg P \vee Q) \equiv \exists x (P \to Q) \, .$$

La seconda si dimostra in modo analogo:

$$\exists x P \to Q \equiv (\neg \exists x P) \vee Q \equiv (\forall x \neg P) \vee Q \equiv \forall x (\neg P \vee Q) \equiv \forall x (P \to Q) \, .$$

Dimostriamo ora la terza equivalenza:

$$Q \to \exists x P \equiv \neg Q \vee (\exists x P) \equiv \exists x (\neg Q \vee P) \equiv \exists x (Q \to P) \, .$$

Per quanto riguarda la quarta equivalenza, si ha:

$$Q \to \forall x P \equiv \neg Q \vee (\forall x P) \equiv \forall x (\neg Q \vee P) \equiv \forall x (Q \to P) \, .$$

□

Osservazione 4.6.10. Si noti che, nel teorema precedente, si può sempre fare in modo che l'ipotesi $x \notin FV(Q)$ sia verificata. A tal fine basta infatti ridenominare opportunamente le variabili legate.

4.7 FORME NORMALI.

In questa sezione mostreremo come sia possibile trasformare una qualunque formula ben formata in una particolare forma normale, detta forma normale prenessa.

Definizione 4.7.1. Una formula ben formata P è in **forma normale prenessa** se ha la seguente forma:

$$P = q_1 x_1 q_2 x 2 \dots q_n x_n P_1 \, ,$$

per qualche $n \geq 0$, dove $q_1, \dots, q_n \in \{\exists, \forall\}$ e dove la formula P_1 non contiene alcun quantificatore. L'espressione $q_1 x_1 q_2 x_2 ... q_n x_n$ viene detta **prefisso**, mentre P_1 è detta la **matrice** della formula P.

Dimostriamo ora il seguente risultato:

Teorema 4.7.2. *Per ogni formula ben formata esiste una formula in forma normale prenessa ad essa equivalente.*

Dimostrazione. La dimostrazione procede per induzione sulla struttura della formula P.

1. Se P è una formula atomica, essa è già in forma normale prenessa.

2. Supponiamo che sia $P = \neg P_1$, per qualche formula ben formata P_1. Per ipotesi induttiva esiste una formula P_1' in forma normale prenessa equivalente a P_1. Allora, per determinare la forma normale prenessa di P è sufficiente scambiare di posto il simbolo di negazione $\neg$ con i quantificatori che compaiono nella formula P_1'.

3. Supponiamo che sia $P = P_1 \wedge P_2$ (risp. $P = P_1 \vee P_2$ o $P = P_1 \to P_2$), ove P_1 e P_2 sono formule ben formate. Per ipotesi induttiva esistono due formule in forma normale prenessa P_1' e P_2' rispettivamente equivalenti a P_1 e P_2. Allora, per determinare la forma normale prenessa di P è sufficiente scambiare di posto il connettivo $\wedge$ (risp. $\vee$ o $\to$) con i quantificatori che compaiono nelle formule P_1' e P_2'.

4. Infine, supponiamo che sia $P = qxP_1$, per qualche formula ben formata P_1, ove $q \in \{\exists, \forall\}$. Per ipotesi induttiva esiste una formula P_1' in forma normale prenessa equivalente a P_1. Allora, la formula $P' = qxP_1'$ è in forma normale prenessa ed è equivalente a P.

□

Illustriamo il contenuto del precedente teorema con alcuni esempi.

Esempio 4.7.3. Consideriamo la formula

$$P = \forall x A(x) \to \forall y B(y)$$

e cerchiamo una sua forma normale prenessa. Utilizzando le equivalenze dimostrate nella sezione precedente, si ha:

$$\forall x A(x) \to \forall y B(y) \equiv \exists x (A(x) \to \forall y B(y)) \equiv \exists x (\forall y (A(x) \to B(y)))$$

$$\equiv \exists x \forall y (A(x) \to B(y)) \, .$$

Tuttavia si ha anche:

$$\forall x A(x) \to \forall y B(y) \equiv \forall y (\forall x A(x) \to B(y)) \equiv \forall y (\exists x (A(x) \to B(y)))$$

$$\equiv \forall y \exists x (A(x) \to B(y)) \, .$$

Da ciò si deduce che le due formule in forma normale prenessa $\exists x \forall y (A(x) \to B(y))$ e $\forall y \exists x (A(x) \to B(y))$ sono tra loro equivalenti (essendo entrambe equivalenti alla formula P).

Esempio 4.7.4. Vogliamo ora determinare la forma normale prenessa della formula

$$P = \forall x A(x) \to \neg \forall y B(y) \, .$$

Utilizzando le equivalenze dimostrate nella sezione precedente, si ha:

$$\begin{aligned} \forall x A(x) \to \neg \forall y B(y) &\equiv \forall x A(x) \to \exists y \neg B(y) \\ &\equiv \exists y (\forall x A(x) \to \neg B(y)) \\ &\equiv \exists y \exists x (A(x) \to \neg B(y)) \, . \end{aligned}$$

Quest'ultima formula è in forma normale prenessa.

Esempio 4.7.5. Consideriamo ora una formula un po' più complessa:

$$P = \neg \forall x A(x) \wedge \exists y B(y) \to \exists x C(x, y) \, .$$

Osserviamo che la variabile legata x compare sia nella sottoformula $\forall x A(x)$ che nella sottoformula $\exists x C(x, y)$, inoltre la variabile y compare sia come variabile libera nella sottoformula $\exists x C(x, y)$ che come variabile legata nella sottoformula $\exists y B(y)$. Come già precedentemente osservato, problemi di questo tipo possono essere facilmente risolti ridenominando opportunamente le variabili:

$$\forall x A(x) \equiv \forall x_1 A(x_1), \exists y B(y) \equiv \exists x 2 B(x2), \exists x C(x, y) \equiv \exists x 3 C(x3, y) \, .$$

Si ha dunque:

$$\begin{aligned}
P \equiv & \neg \forall x_1 A(x_1) \wedge \exists x_2 B(x_2) \rightarrow \exists x_3 C(x_3, y) \\
\equiv & \exists x_1 \neg A(x_1) \wedge \exists x_2 B(x_2) \rightarrow \exists x_3 C(x_3, y) \\
\equiv & \exists x_1 (\neg A(x_1) \wedge \exists x_2 B(x_2)) \rightarrow \exists x_3 C(x_3, y) \\
\equiv & \exists x_1 \exists x_2 (\neg A(x_1) \wedge B(x_2)) \rightarrow \exists x_3 C(x_3, y) \\
\equiv & \forall x_1 (\exists x_2 (\neg A(x_1) \wedge B(x_2)) \rightarrow \exists x_3 C(x_3, y)) \\
\equiv & \forall x_1 \forall x_2 (\neg A(x_1) \wedge B(x_2) \rightarrow \exists x_3 C(x_3, y)) \\
\equiv & \forall x_1 \forall x_2 \exists x_3 (\neg A(x_1) \wedge B(x_2) \rightarrow C(x_3, y)) \, .
\end{aligned}$$

5

CALCOLO SEMANTICO

A differenza della logica proposizionale, verificare che una formula del primo ordine ben formata sia una tautologia è un compito abbastanza complicato, infatti essa è una tautologia se è valida in tutti i possibili domini di interpretazione e per tutte le possibili interpretazioni delle variabile. Una tale classe è infinita, è perciò impossibile controllare caso per caso che la proprietà valga (cosa che invece facciamo in logica proposizionale quando calcoliamo riga per riga la tavola di verità).

In questo breve capitolo presenteremo un metodo più efficace per la verifica di tautologie del primo ordine. L'algoritmo, che chiameremo Tab$\forall$ costruisce, a partire da una formula, un albero di possibili valutazioni per quella formula. Partendo dalla negazione della formula data, se un ramo dell'albero dovesse arrivare a conclusione positivamente, otterremmo una valutazione che falsifica la formula data; questa non sarebbe dunque una tautologia. Se viceversa l'algoritmo dovesse chiudere tutti i rami negativamente (cioè con qualche contraddizione all'interno di ogni ramo) allora vorrebbe dire che è impossibile falsificare la formula data o, equivalentemente, che essa è una tautologia.

L'algoritmo Tab$\forall$ è un'estensione di quello dato nel Capitolo **??**, quindi le regole per gestire i connettivi preposizionali sono le stesse trovate in precedenza.

Descriviamo quindi solo come trattare i quantificatori.

I passi per i quantificatori da effettuare nell'algoritmo Tab$\forall$ si dividono in due tipi: del I tipo e del II tipo.

5.1 REGOLE DEL I TIPO.

Se $X = \exists x X_1$ allora l'albero si prolunga come segue:

$$\begin{array}{c} V(\exists x X_1) \\ | \\ V(X_1[c/x]) \end{array}$$

qui il simbolo di costante c è nuovo, nel senso che esso deve essere scelto tra i simboli di costante che non appaiono nell'albero (non è necessario che c appartenga al linguaggio). Intuitivamente, se vogliamo verificare $X = \exists x X_1$ bisogna verificare che $X_1[c/x]$ valga per qualche costante c.

Se $X = \forall x X_1$ allora l'albero si prolunga come segue:

$$F(\forall x X_1)$$
$$\vert$$
$$F(X_1[c/x])$$

anche qui il simbolo di costante c è nuovo, nel senso che esso deve essere scelto tra i simboli di costante che non appaiono nell'albero (non è necessario che c appartenga al linguaggio). Intuitivamente, se vogliamo falsificare $X = \forall x X_1$ bisogna falsificare $X_1[c/x]$ per qualche costante c.

Come al solito, nelle regole del I tipo, il nodo a cui è stata applicata la regola viene segnato poiché esso non verrà più considerato. Al contrario i nodi a cui si applicano regole del II tipo non verranno segnati perché essi vanno riconsiderati ogni volta che un nuovo simbolo di costante appare nell'albero.

5.2 REGOLE DEL II TIPO.

Se $X = \exists x X_1$ allora l'albero si prolunga come segue:

$$F(\exists x X_1)$$
$$\vert$$
$$F(X_1[c_1/x])$$
$$\vert$$
$$\vdots$$
$$\vert$$
$$F(X_1[c_n/x])$$

qui $c_1, \ldots, c_n$ sono tutte i simboli di costante già apparsi nel ramo che stiamo prolungando. Se non dovessero essere apparsi simboli di constatante precedentemente, se ne introduce uno nuovo. Intuitivamente, se vogliamo falsificare $X = \exists x X_1$ dobbiamo falsificare ogni $X_1[c_i/x]$ per ogni possibile costante c_i presa in considerazione.

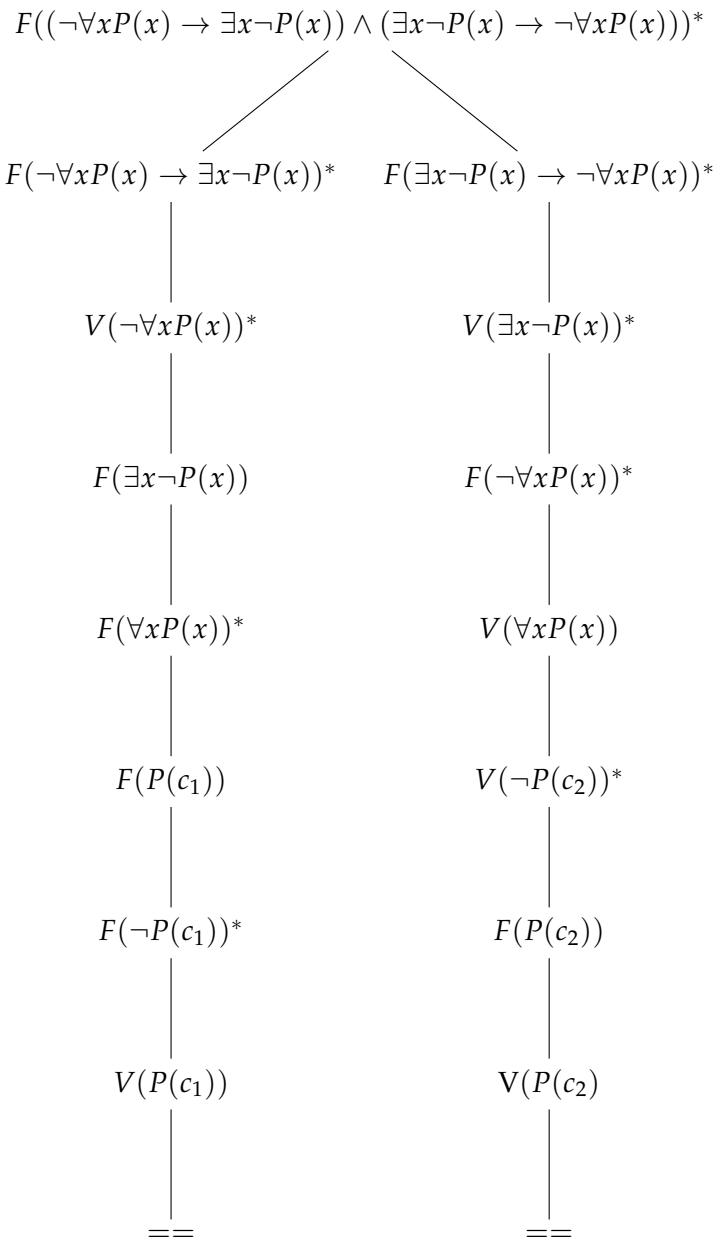
Se $X = \forall x X_1$ allora l'albero si prolunga come segue:

$$V(\forall x X_1)$$
$$\vert$$
$$V(X_1[c_1/x])$$
$$\vert$$
$$\vdots$$
$$\vert$$
$$V(X_1[c_n/x])$$

qui $c_1, \ldots, c_n$ sono tutte i simboli di costante già apparsi nel ramo che stiamo prolungando. Se non dovessero essere apparsi simboli di constatante precedentemente, se ne introduce uno nuovo. Intuitivamente, se vogliamo verificare $X = \forall x X_1$ dobbiamo verificare $X_1[c/x]$ per ogni costante c_i già presa in considerazione.

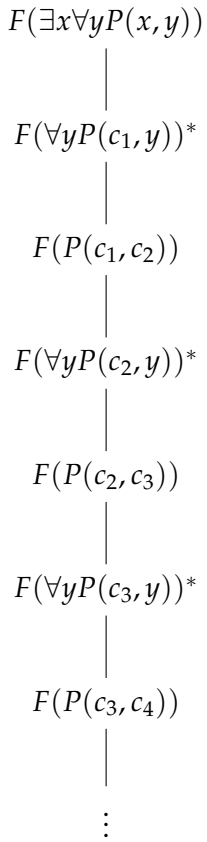
Le regole del secondo tipo non vengono applicate una sola volta, ma si *riattivano* ogni volta che appare un nuovo simbolo di costante nel ramo a cui esse appartengono. Questo fa sì che, al contrario di quello che succede nell'algoritmo in logica proposizionale, questo algoritmo possa andare avanti all'infinito (si veda l'Esempio 5.2.2).

Esempio 5.2.1. Dimostriamo che $\neg\forall x P(x) \leftrightarrow \exists x \neg P(x)$ è una tautologia del primo ordine.



Poiché tutti i rami portano ad una contraddizione la formula è una tautologia

Esempio 5.2.2. Consideriamo ora la formula $\forall x \exists y P(x, y)$.



Come si vede, l'algoritmo entra in un ciclo dal quale non uscirà mai, generando sempre nuove costanti.

BIBLIOGRAFIA

[1] Andrea Asperti, Agata Ciabattoni. *Logica a informatica.* McGraw-Hill. 1997. 204 pp.

[2] Marcello Frixione. *Come ragioniamo* Laterza. 2007. 167+VI pp.

[3] Giangiacomo Gerla. *Linguaggio e verità.* ilmiolibro.it. 2011. 274 pp. (also available in reduced form, for free, at http://www.dipmat.unisa.it/people/gerla/www/Down/Light%20logica.pdf)

[4] Piergiorgio Odifreddi. *Il diavolo in cattedra. La logica da Aristotele a Godel* Einaudi. 2004. 299 pp.

[5] Dario Palladino. *Corso di logica. Introduzione elementare al calcolo dei predicati* Carocci. 2010. 412 pp.