Logica

Dispense per il corso di Logica per L'Informatica
a.a. 2013/2014

(estratte dal libro: M. Cialdea Mayer. *Logica. Linguaggio, ragionamento, calcolo.*
Esculapio, 2002, con alcune integrazioni e correzioni)

Marta Cialdea Mayer

Indice

Capitolo 1

Logica e Informatica

La logica è lo studio delle forme del ragionamento, con l'obiettivo principale di riconoscere e individuare le forme dei ragionamenti corretti, e distinguerle da quelli scorretti. Per questo motivo essa viene a volte chiamata logica "formale", o anche "simbolica", perché le "forme" del ragionamento vengono espresse mediante l'uso di simboli, che consentono di astrarre dal contenuto dei ragionamenti stessi. Astrarre significa semplificare: sbarazzandosi dei dettagli, si guardano oggetti diversi da uno stesso punto di vista, e gli si può dare un trattamento uniforme.

Si considerino ad esempio i due ragionamenti seguenti:

a) Se nevica, la temperatura è di $0^o C$.
   Nevica.
   Quindi la temperatura è di $0^o C$.

b) Se la matematica è un'opinione, allora 1 + 1 = 0.
   La matematica è un'opinione.
   Quindi 1 + 1 = 0.

I due ragionamenti hanno evidentemente la stessa forma, anche se essi riguardano concetti diversi e diverso è il contenuto di verità delle proposizioni in essi coinvolte. Possiamo identificare la loro forma comune utilizzando simboli, A e B, al posto delle proposizioni:

Se A allora B.
A.
Quindi B.

Lo studio della logica, nato nell'antica Grecia con Aristotele e i logici megarici e stoici, ha conosciuto un grande sviluppo a partire dalla fine del secolo scorso, soprattutto come studio del tipo di ragionamento normalmente condotto nel dimostrare teoremi matematici (logica matematica). Per rappresentare proposizioni e ragionamenti, è stato definito un linguaggio formale che, attraverso una serie di evoluzioni, ha raggiunto una forma standard ormai universalmente accettata.

In realtà non è corretto parlare di "logica", al singolare, in quanto sono state studiate diverse "logiche", che trovano applicazione in diversi domini. Ciascuna di esse è caratterizzata da un suo linguaggio formale (con una *sintassi*, che definisce l'insieme delle espressioni corrette del linguaggio, ed una *semantica* che ne definisce il significato) e uno o più metodi per "ragionare" con tali espressioni. La logica più antica e più nota, che costituisce la base di molte altre logiche, è la *logica classica* (proposizionale e dei predicati): un linguaggio formale per la rappresentazione di conoscenza di tipo *dichiarativo*. Altre forme di di conoscenza non possono però essere rappresentate facilmente in logica classica, e sono state studiate altre logiche come, ad esempio la "logica epistemica" (che permette di rappresentare conoscenze espresse in italiano da espressioni del tipo "l'agente sa che ...", "l'agente crede che ...") o le logiche "temporali", che consentono la rappresentazione di un mondo che cambia nel tempo mediante operatori che esprimono concetti quali "sarà sempre vero che ...", "in qualche momento futuro sarà vero che ...".

La logica classica è il formalismo fondamentale del ragionamento matematico: come le proposizioni della matematica, le espressioni della logica classica sono caratterizzate dal fatto di essere o vere o false (indipendentemente dal tempo, dalla nostra conoscenza di esse, ecc.).

Tra i campi di applicazione più importanti della logica rientrano diversi settori dell'informatica. Logica e informatica sono infatti discipline in stretta relazione, nonostante la diversità di "punti di vista": la logica si è principalmente interessata della verità delle proposizioni, cioè di "che cosa" si può dire, e questo viene stabilito mediante dimostrazioni; al contrario, l'informatica è interessata al "come" fare. La logica ha fornito e fornisce importanti contributi all'informatica, sotto diversi aspetti. Viceversa, non soltanto è campo d'applicazione per l'informatica, che fornisce ai logici utili strumenti di lavoro, ma l'informatica ha offerto allo sviluppo della logica nuovi problemi e punti di vista, quali, ad esempio, le problematiche relative allo studio di strutture finite o all'efficienza di procedure logiche.

Per quel che riguarda l'impatto della logica sull'informatica, osserviamo che essa è innanzitutto uno strumento di base per la meta-informatica: ha portato, ad esempio, a definire la nozione di calcolabilità, ai risultati di indecidibilità, alla classificazione dei problemi secondo la loro complessità. Questo è il ruolo esterno e teorico che ha la logica rispetto all'informatica, che riproduce il rapporto che ha e ha sempre avuto con la matematica, relativo alla questione dei loro fondamenti.

La logica ha fornito all'informatica molti concetti fondamentali, come quello di semantica formale, che è alla base della semantica denotazionale dei linguaggi di programmazione, e metodologie generali come il trattamento assiomatico di informazioni, che trova applicazione, ad esempio, nella dimostrazione di proprietà di programmi.

La logica svolge inoltre un ruolo interno e pratico rispetto all'informatica, dove trovano applicazione i suoi metodi e gli strumenti da essa forniti. La logica è infatti il formalismo fondamentale per la rappresentazione della conoscenza in intelligenza artificiale, dove trovano applicazione i metodi automatici per la dimostrazione di teoremi, studiati e realizzati prima ancora della loro applicazione nella costruzione di sistemi di informatica e intelligenza artificiale. Inoltre, i linguaggi di specifica formale e la specifica algebrica di tipi astratti di dati si fondano su metodologie logiche.

Uno dei terreni in cui la logica gioca un ruolo pratico essenziale è relativo ai paradigmi di calcolo. I due fondamentali sono quelli della risoluzione e della riduzione, associati rispettivamente alla *programmazione logica* e alla *programmazione funzionale*.

Il primo approccio è più direttamente basato sulla logica e consiste nel vedere il calcolo come la risoluzione di un problema. Esempio tipico è quello delle equazioni: la soluzione di un'equazione, ad esempio $x^2 - 9 = 0$, è data dai valori delle sue variabili che la rendono vera ($x = 3$ o $x = -3$). La programmazione logica si basa su importanti risultati teorici relativi alla possibilità di definire metodi di dimostrazione automatica e, restringendo opportunamente il linguaggio, metodi *costruttivi*. La costruttività di una dimostrazione consente di estrarre dalla dimostrazione stessa informazioni importanti, che si aggiungono al fatto di aver determinato che una certa formula è un teorema. Un po' più in dettaglio, un *programma logico* P consiste in un isieme di assiomi che descrivono in modo opportuno il dominio di interesse e viene eseguito a fronte di un *obiettivo* $A(x_1, ..., x_n)$; l'esecuzione consiste nella ricerca di una dimostrazione costruttiva di $\exists x_1 ... \exists x_n A(x_1, ..., x_n)$ a partire da P. Se tale dimostrazione esiste, è possibile da essa estrarre dei valori (termini) $t_1, ..., t_n$ tali che $A(t_1, ..., t_n)$ è dimostrabile da P. Tali termini costituiscono il risultato del calcolo.

Secondo il paradigma della *riduzione*, si calcola riducendo un'espressione a un'altra più semplice o più vicina a un valore, cioè a un'espressione non ulteriormente semplificabile. Ad esempio:

$$(6+3) \times (8-2) \Rightarrow 9 \times (8-2) \Rightarrow 9 \times 6 \Rightarrow 54$$

La *programmazione funzionale* si basa su questo: un programma funzionale è costituito dalla definizione di un insieme di funzioni, che possono richiamarsi l'una con l'altra. Eseguire un programma funzionale consiste nel calcolare il valore di una data espressione, utilizzando tali definizioni, semplificandola fin dove possibile.

Lo scheletro concettuale di molti linguaggi funzionali, come ML, Miranda, Hope e lo stesso LISP è costituito dal *lambda-calcolo* (λ-calcolo). È un linguaggio logico semplice, che consente, esplicitamente, soltanto la definizione di funzioni e la loro applicazione. Tuttavia, dietro questa semplicità sintattica, si cela un linguaggio di grande potenza espressiva: rispetto alla capacità di esprimere algoritmi, il lambda-calcolo è altrettanto potente quanto gli altri linguaggi di programmazione. D'altro canto, il λ-calcolo è una *teoria delle funzioni*. Esso è nato infatti prima dello sviluppo dei linguaggi di programmazione; è stato introdotto da Church negli anni 30, allo scopo di dare una definizione formale di ciò che è calcolabile e studiare i limiti della calcolabilità.

Il ruolo pratico della logica rispetto all'informatica, sopra brevemente illustrato, è di fatto la manifestazione dell'identità profonda esistente tra logica e programmazione. È dimostrato infatti che concepire un programma equivale a dimostrare una proposizione in maniera costruttiva, ed eseguire un programma funzionale equivale a trasformare una dimostrazione in una forma "normale".

Capitolo 2

La Logica Proposizionale

2.1 Il linguaggio della logica proposizionale

Consideriamo di nuovo i semplici ragionamenti del paragrafo precedente. Ciascuno di essi è caratterizzato da un insieme di *premesse* (le affermazioni precedenti la parola "quindi") e una *conclusione* (la proposizione che segue la parola "quindi"). La forma comune dei due ragionamenti può essere schematizzata come segue, separando le premesse dalla conclusione mediante una linea (la *linea di inferenza*), anziché la parola "quindi":

$$\frac{\text{Se } A\text{, allora } B \qquad A}{B}$$

Qui A e B sono simboli, che stanno per proposizioni qualsiasi. Invece "se" e "allora" sono "parole logiche" che connettono proposizioni. Se il ragionamento è corretto, le premesse – se sono vere – giustificano la conclusione. In altri termini, un ragionamento è corretto quando *ogni volta che sono vere* le premesse, è vera anche la conclusione (ovviamente, quando le premesse sono false, non si può dire niente sulla conclusione). È in base al significato delle parole logiche che possiamo dire se un ragionamento è corretto o no. Le parole logiche sono espresse da simboli particolari. Ad esempio, "se–allora" viene chiamato "implicazione" e rappresentato da una freccia. Utilizzando questo simbolismo, la forma di ragionamento sopra riportata si può scrivere così:

$$\frac{A \to B \qquad A}{B}$$

Ragionamenti di questa forma sono sempre corretti. Questo viene stabilito in base al significato di $\to$, che assicura, in particolare, che $A \to B$ è falso solo quando A è vero e B è falso (ciò si vedrà meglio in seguito). Quindi, se sia A che $A \to B$ sono veri, B deve essere vero.

La logica proposizionale studia il significato di parole logiche che, come l'implicazione, si possono usare per "connettere" proposizioni, e che sono chiamate appunto *connettivi proposizionali* (o *operatori booleani*). I mattoni di base di un linguaggio proposizionale sono costituiti dalle *proposizioni atomiche*, che rappresentano enunciati non ulteriormente analizzati. Un enunciato è un'affermazione di tipo dichiarativo per la quali ha senso chiedersi se sia vera o falsa. Le proposizioni atomiche sono rappresentate da simboli, detti *lettere proposizionali* o *variabili proposizionali.* Gli enunciati semplici possono essere composti per formare enunciati più complessi, mediante i connettivi proposizionali.

Un linguaggio proposizionale è dunque costituito da un insieme di simboli, alcuni dei quali (le lettere proposizionali) possono variare da un linguaggio all'altro, mentre altri (i connettivi) sono un patrimonio comune di tutti i linguaggi proposizionali. I principali connettivi proposizionali sono i seguenti:

$\neg$ è la negazione: si legge "non";

$\wedge$ è la congiunzione: si legge "e";

$\vee$ è la disgiunzione: si legge "oppure";

$\rightarrow$ è l'implicazione: si legge "implica" (o "se ... allora ...");

$\equiv$ è la doppia implicazione: si legge "se e solo se".

Oltre ai connettivi, i linguaggi che considereremo conterranno tutti due *costanti proposizionali*: $\top$, che sta per un enunciato sempre vero, e $\perp$, che sta per un enunciato sempre falso. Le espressioni, semplici o complesse, che si possono formare sulla base di un linguaggio proposizionale sono chiamate *formule proposizionali.*

Ad esempio, l'enunciato "Caino è fratello di Abele" è un atomo, che possiamo rappresentare mediante la lettera proposizionale p; "a ogni paziente piace qualche dottore" è un altro atomo, che rappresentiamo mediante un'altra variabile proposizionale, q. Le frasi a sinistra nella tabella che segue sono allora rappresentate dalle formule a destra:

Caino è fratello di Abele *oppure* a ogni paziente piace qualche dottore	$p \vee q$
Caino *non* è fratello di Abele	$\neg p$
Caino è fratello di Abele *e* a ogni paziente piace qualche dottore	$p \wedge q$
Se Caino è fratello di Abele, *allora* a ogni paziente piace qualche dottore	$p \rightarrow q$
Caino è fratello di Abele *se e solo se* a ogni paziente piace qualche dottore	$p \equiv q$
Se Caino *non* è fratello di Abele, *allora* Caino è fratello di Abele *oppure* a ogni paziente piace qualche dottore	$\neg p \rightarrow (p \vee q)$

La *sintassi* della logica proposizionale determina quali sono le espressioni corrette di tale logica, cioè definisce che cos'è un linguaggio proposizionale. Come abbiamo visto, un linguaggio proposizionale è costruito sulla base di un determinato insieme P di variabili proposizionali. Ad esempio potremmo considerare:

$$\begin{gathered} P = \{p, q, r\} \\ P' = \{p_0, p_1, p_2, ...\} \\ P'' = \{caino\_fratello\_di\_abele, pazienti\_dottori, ...\} \end{gathered}$$

Le espressioni del linguaggio della logica proposizionale sono dette **formule proposizionali** – a volte abbreviato in *fbf* per *"formule ben formate"*. Dato un insieme P di variabili proposizionali, l'insieme delle formule basate su P (il linguaggio proposizionale basato su P), $Prop[P]$, è definito induttivamente come segue.

Definizione 2.1.1 *Sia P un insieme, i cui elementi sono chiamati* **lettere (o variabili) proposizionali**. *L'insieme delle formule in $Prop[P]$ è definito induttivamente:*

1. *ogni variabile proposizionale in P è una formula;*

2. *$\top$ e $\perp$ sono formule;*

3. *se A è una formula, allora anche $\neg A$ è una formula;*

4. *se A e B sono formule, allora anche* $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \equiv B)$ *sono formule;*
5. *nient'altro è una formula.*

Una formula della forma $A \wedge B$ si dice congiunzione, e A e B sono i due congiunti. Una formula della forma $A \vee B$ si dice disgiunzione, e A e B sono i due disgiunti. Una formula della forma $A \rightarrow B$ si dice implicazione, A è l'antecedente e B il conseguente dell'implicazione. Una formula della forma $A \equiv B$ si dice doppia implicazione.

Consideriamo ad esempio l'insieme di lettere proposizionali $P = \{p, q, r, s\}$; le espressioni seguenti sono in $Prop[P]$:

$$\begin{array}{l}
p,\ q,\ r,\ s,\ \top,\ \bot, \\
\neg p,\ \neg q,\ \neg r,\ \neg s,\ \neg\top,\ \neg\bot, \\
(p \wedge q),\ (p \vee q),\ (q \rightarrow r),\ (r \equiv s),\ \ldots \\
\neg(p \wedge q),\ ((p \vee q) \equiv (q \rightarrow r)),\ ((q \rightarrow r) \vee \neg s),\ \ldots \\
(\neg(p \wedge q) \rightarrow ((p \vee q) \equiv (q \rightarrow r))),\ ((q \rightarrow r) \vee \neg s) \wedge \neg\top),\ \ldots
\end{array}$$

Convenzioni sull'uso delle parentesi: ometteremo le parentesi dalle formule quando non ci sia rischio di ambiguità, secondo le seguenti convenzioni:

- Si possono omettere le parentesi esterne. Ad esempio, anziché $(p \wedge q)$ si può scrivere $p \wedge q$.
- Associatività: i connettivi associano da sinistra a destra (se non indicato altrimenti dall'uso di parentesi). Ad esempio: $((p \wedge q) \wedge r)$ si può scrivere $p \wedge q \wedge r$, e $((p \rightarrow q) \rightarrow r)$ si può scrivere $p \rightarrow q \rightarrow r$. Ma $(p \rightarrow (q \rightarrow r))$ non si può scrivere $p \rightarrow q \rightarrow r$, che sta invece per $((p \rightarrow q) \rightarrow r)$.
- L'ordine di precedenza dei connettivi è:
  $$\neg,\ \wedge,\ \vee,\ \rightarrow,\ \equiv$$
  (se non indicato altrimenti dall'uso di parentesi). Ad esempio: $(p \rightarrow (q \wedge r))$ si può scrivere $p \rightarrow q \wedge r$, e $\neg(p \rightarrow (q \wedge r))$ si può scrivere $\neg(p \rightarrow q \wedge r)$. Ma $\neg(p \rightarrow (q \wedge r))$ non si può scrivere $\neg p \rightarrow q \wedge r$, che sta invece per $((\neg p) \rightarrow (q \wedge r))$; e $(p \rightarrow (q \equiv r))$ non si può scrivere $p \rightarrow q \equiv r$, che sta invece per $((p \rightarrow q) \equiv r)$.

Quindi, ad esempio:

$s \rightarrow p \wedge q \vee \neg r$ abbrevia $(s \rightarrow ((p \wedge q) \vee \neg r))$

$p \rightarrow (p \rightarrow r \wedge q)$ abbrevia $(p \rightarrow (p \rightarrow (r \wedge q)))$

$p \equiv \neg q \equiv r \vee s$ abbrevia $((p \equiv \neg q) \equiv (r \vee s))$

Se A è una formula, una sua sottoformula è una qualsiasi sottoespressione (propria o impropria) di A che sia essa stessa una formula, cioè ogni formula che compare nella costruzione di A secondo la definizione 2.1.1. Con maggior precisione, l'insieme delle sottoformule di una formula A, $subf(A)$, è definito induttivamente come segue:

1. Se A è atomica allora $subf(A) = \{A\}$.
2. $subf(\neg A) = subf(A) \cup \{\neg A\}$.
3. $subf(A \wedge B) = subf(A) \cup subf(B) \cup \{A \wedge B\}$.
4. $subf(A \vee B) = subf(A) \cup subf(B) \cup \{A \vee B\}$.
5. $subf(A \rightarrow B) = subf(A) \cup subf(B) \cup \{A \rightarrow B\}$.
6. $subf(A \equiv B) = subf(A) \cup subf(B) \cup \{A \equiv B\}$.

Il *connettivo principale* di una formula (non atomica) è l'ultimo connettivo utilizzato nella costruzione della formula:

1. il connettivo principale di $\neg A$ è $\neg$;
2. il connettivo principale di $A \wedge B$ è $\wedge$;
3. il connettivo principale di $A \vee B$ è $\vee$;
4. il connettivo principale di $A \rightarrow B$ è $\rightarrow$;
5. il connettivo principale di $A \equiv B$ è $\equiv$.

Poiché l'insieme delle formule è definito induttivamente, possiamo formulare un corrispondente principio di induzione, il *principio di induzione sulle formule*:

> Se P è una proprietà delle formule tale che:
>
> **(B)** P vale per ogni lettera proposizionale, per $\top$ e $\bot$, e
>
> **(PI)** per tutte le formule A e B:
>
> - se P vale per A allora P vale per $\neg A$;
> - se P vale per A e B, allora P vale per $(A \wedge B)$, per $(A \vee B)$, per $(A \rightarrow B)$, per $(A \equiv B)$;
>
> allora P vale per ogni formula.

Vedremo presto, nella dimostrazione del Teorema 2.1, un'applicazione di questo principio di induzione.

2.2 Semantica: interpretazioni e verità

La sintassi di un linguaggio determina la forma delle espressioni corrette nel linguaggio. La semantica assegna un significato alle espressioni sintatticamente corrette di tale linguaggio. Per stabilire la semantica di un linguaggio, occorre specificare innanzituto un **dominio di interpretazione**, cioè l'insieme degli oggetti che costituiscono possibili significati per le espressioni del linguaggio. La semantica di un linguaggio associa allora ad ogni espressione sintattica un oggetto del dominio di interpretazione.

Nel caso della logica proposizionale il dominio di interpretazione è costituito dall'insieme $Bool = \{T, F\}$ dei valori booleani (o valori di verità). Per studiare il significato dei connettivi proposizionali, infatti, si stabilisce che una proposizione atomica può essere vera o falsa. Il significato dei connettivi viene stabilito definendo il valore di verità di proposizioni complesse a partire da quello dei loro componenti immediati.

Il significato (il valore di verità) di una formula in $Prop[P]$ dipende quindi dal significato (il valore di verità) delle lettere proposizionali in P. Dunque, per stabilire il significato di una formula proposizionale occorre *assegnare* un valore di verità a ciascuna variabile proposizionale. Ad esempio, il significato della formula $p \wedge q$ si può stabilire soltanto conoscendo il significato di p e di q; se tali atomi sono entrambi veri, allora $p \wedge q$ è vero, altrimenti è falso.

Definizione 2.2.1 *Sia P un insieme di lettere proposizionali. Una* **interpretazione** *o* **assegnazione** $\mathcal{M}$ *per P è una funzione*

$$\mathcal{M} : P \rightarrow Bool$$

Ad esempio, se $P = \{p, q, r\}$, ci sono 8 possibili interpretazioni di P:

	p	q	r
$\mathcal{M}_1$	T	T	T
$\mathcal{M}_2$	T	T	F
$\mathcal{M}_3$	T	F	T
$\mathcal{M}_4$	T	F	F

	p	q	r
$\mathcal{M}_5$	F	T	T
$\mathcal{M}_6$	F	T	F
$\mathcal{M}_7$	F	F	T
$\mathcal{M}_8$	F	F	F

L'interpretazione di una formula dipende dall'interpretazione delle lettere proposizionali che occorrono in essa. Una volta fissata un'interpretazione $\mathcal{M}$ per P, risulta comunque stabilita l'interpretazione di qualsiasi formula in $Prop[P]$, in base al significato delle lettere proposizionali che occorrono in essa e al significato dei connettivi logici. Se A è una formula e $\mathcal{M}$ un'interpretazione di A, la notazione $\mathcal{M} \models A$ indica che A è vera in $\mathcal{M}$, cioè che il significato di A in $\mathcal{M}$ è T (e si dice allora che $\mathcal{M}$ è un *modello* di A). Se A è falsa in $\mathcal{M}$ scriviamo $\mathcal{M} \not\models A$ (e si dice allora che $\mathcal{M}$ è un *contromodello* di A). La verità o falsità di una formula in $\mathcal{M}$ costituisce la sua interpretazione secondo $\mathcal{M}$, ed è definita induttivamente come segue ("sse" sta per "se e solo se"):

Definizione 2.2.2 *La verità di una formula $A \in Prop[P]$ in un'interpretazione $\mathcal{M}$ per P è definita induttivamente come segue:*

1. $\mathcal{M} \models p$ *sse* $\mathcal{M}(p) = T$*, se* $p \in P$*;*
2. $\mathcal{M} \models \top$ *e* $\mathcal{M} \not\models \bot$*;*
3. $\mathcal{M} \models \neg A$ *sse* $\mathcal{M} \not\models A$
4. $\mathcal{M} \models A \wedge B$ *sse* $\mathcal{M} \models A$ *e* $\mathcal{M} \models B$
5. $\mathcal{M} \models A \vee B$ *sse* $\mathcal{M} \models A$ *oppure* $\mathcal{M} \models B$
6. $\mathcal{M} \models A \rightarrow B$ *sse* $\mathcal{M} \not\models A$ *oppure* $\mathcal{M} \models B$
7. $\mathcal{M} \models A \equiv B$ *sse* $\mathcal{M} \models A$ *e* $\mathcal{M} \models B$*, oppure* $\mathcal{M} \not\models A$ *e* $\mathcal{M} \not\models B$

Se $\mathcal{M} \models A$ diciamo che $\mathcal{M}$ è un modello *di A o che $\mathcal{M}$* soddisfa *A. Se $\mathcal{M} \not\models A$ allora $\mathcal{M}$ è un* contromodello *di A.*

Se S è un insieme di formule, allora $\mathcal{M}$ è un modello di S (e scriviamo $\mathcal{M} \models S$) sse $\mathcal{M} \models A$ per ogni formula $A \in S$.

Osserviamo che dalla definizione precedente segue la definizione di $\mathcal{M} \not\models A$:

1. $\mathcal{M} \not\models p$ sse $\mathcal{M}(p) = F$, se $p \in P$;
2. $\mathcal{M} \not\models \top$ è sempre falso e $\mathcal{M} \not\models \bot$ è sempre vero;
3. $\mathcal{M} \not\models \neg A$ sse $\mathcal{M} \models A$;
4. $\mathcal{M} \not\models A \wedge B$ sse $\mathcal{M} \not\models A$ oppure $\mathcal{M} \not\models B$;
5. $\mathcal{M} \not\models A \vee B$ sse $\mathcal{M} \not\models A$ e $\mathcal{M} \not\models B$;
6. $\mathcal{M} \not\models A \rightarrow B$ sse $\mathcal{M} \models A$ e $\mathcal{M} \not\models B$;
7. $\mathcal{M} \not\models A \equiv B$ sse $\mathcal{M} \models A$ e $\mathcal{M} \not\models B$, oppure $\mathcal{M} \not\models A$ e $\mathcal{M} \models B$;

La definizione 2.2.2, di fatto, stabilisce il significato dei connettivi logici. Essa si può riformulare infatti come segue. Se denotiamo con NOT, AND, OR, IMP e IFF le operazioni booleane su $\{T, F\}$, tali che:

	NOT
T	F
F	T

		AND	OR	IMP	IFF
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

abbiamo stabilito che il significato di:

$$\begin{array}{ccc@{\qquad}ccc} \neg & \text{è} & NOT & \wedge & \text{è} & AND \\ \vee & \text{è} & OR & \rightarrow & \text{è} & IMP \\ \equiv & \text{è} & IFF & & & \end{array}$$

Quindi questa definizione stabilisce il significato dei connettivi logici.

Se $A \in Prop[P]$, ad ogni assegnazione di valori di verità T o F alle lettere proposizionali che compaiono in A corrisponde un valore di verità della formula A stessa; questo si può ottenere utilizzando la definizione 2.2.2. Ad esempio, possiamo calcolare il valore di verità di $A = (\neg p \vee q) \rightarrow r$ nell'interpretazione $\mathcal{M}$ tale che $\mathcal{M}(p) = \mathcal{M}(q) = T$ e $\mathcal{M}(r) = F$ come segue:

$\mathcal{M} \models (\neg p \vee q) \rightarrow r$
sse $\mathcal{M} \not\models \neg p \vee q$ oppure $\mathcal{M} \models r$
sse $\mathcal{M} \not\models \neg p \vee q$ (perché $\mathcal{M}(r) = F$, quindi $\mathcal{M} \not\models r$)
sse $\mathcal{M} \not\models \neg p$ e $\mathcal{M} \not\models q$
sse $\mathcal{M} \models p$ e $\mathcal{M}(q) = F$
sse $\mathcal{M}(p) = T$ e $\mathcal{M}(q) = F$: FALSO.
Quindi $\mathcal{M} \not\models (\neg p \vee q) \rightarrow r$

2.3 Tavole di verità

Il significato dei connettivi proposizionali si può rappresentare sinteticamente mediante le *tavole di verità* dei connettivi logici, come illustrato nella tabella seguente. Nella prima e nella seconda colonna vengono mostrati i valori di verità di A e B e nelle colonne successive i valori di verità delle formule ottenute mediante applicazione dei connettivi.

A	B	$\neg A$	$A \vee B$	$A \wedge B$	$A \rightarrow B$	$A \equiv B$
T	T	F	T	T	T	T
T	F	F	T	F	F	F
F	T	T	T	F	T	F
F	F	T	F	F	T	T

Quando si vuole calcolare il valore di verità di una formula A in *tutte* le sue interpretazioni, un metodo compatto, alternativo all'applicazione ripetuta della definizione 2.2.2, consiste nella costruzione della *tavola di verità* per la formula A che utilizza le tavole di verità dei connettivi logici. Ad esempio, la tavola di verità della formula $A = (\neg p \vee q) \rightarrow r$ si costruisce come segue:

p	q	r	$\neg p$	$\neg p \vee q$	$(\neg p \vee q) \to r$
T	T	T	F	T	T
F	T	T	T	T	T
T	F	T	F	F	T
F	F	T	T	T	T
T	T	F	F	T	F
F	T	F	T	T	F
T	F	F	F	F	T
F	F	F	T	T	F

Ogni riga rappresenta un'assegnazione di valori di verità alle lettere p, q, r (cioè un'interpretazione per $\{p, q, r\}$) e il corrispondente valore di verità assunto dalle sottoformule di A.

Si noti che, se in una formula ci sono n lettere diverse, si hanno 2^n possibili assegnazioni di valori di verità alle lettere enunciative e quindi 2^n righe nella tavola di verità.

Una tavola di verità può essere abbreviata scrivendo solo la formula completa, mettendo i valori di verità delle lettere proposizionali sotto ciascuna di esse e scrivendo, passo per passo, il valore di verità di ogni sottoformula sotto il suo connettivo principale. Ad esempio, la tavola di verità abbreviata per la formula $(p \vee q) \to (p \wedge \neg q)$ è la seguente:

$(p$	$\vee$	$q)$	$\to$	$(p$	$\wedge$	$\neg$	$q)$
T	T	T	F	T	F	F	T
T	T	F	T	T	T	T	F
F	T	T	F	F	F	F	T
F	F	F	T	F	F	T	F

Si noti che, per determinare il valore di verità della formula completa, non sempre è necessario riempire tutti i dettagli di una data riga. Ad esempio, nell'ultima riga della tavola precedente, non appena è stato determinato che il valore di $(p \vee q)$ è F, si può concludere che il valore della formula intera è T, senza bisogno di calcolare il valore di $(p \wedge \neg q)$.

2.4 Connettivi logici e linguaggio naturale

La semantica dei connettivi logici non corrisponde sempre esattamente a quello delle parole corrispondenti utilizzate nel linguaggio naturale. Consideriamo quindi più in dettaglio il loro significato, in relazione al linguaggio naturale.

1. Una congiunzione ($\wedge$) è vera se e solo se entrambi gli argomenti sono veri. Sebbene la congiunzione "e" sia a volte utilizzata in italiano per dire "e poi", la congiunzione logica non ha alcun significato "temporale". Inoltre, $\wedge$ può corrispondere anche ad altre congiunzioni come "ma", "sebbene", ecc.

2. Una doppia implicazione ($\equiv$) è vera se e solo se i due argomenti hanno lo stesso valore di verità: IFF è l'uguaglianza sui booleani. In italiano si può dire "se e solo se" oppure "condizione necessaria e sufficiente". Ad esempio, "sarai promosso (p) *se e solo se* avrai scritto almeno 125 programmi corretti (c)" si può rappresentare con $p \equiv c$. "*Condizione necessaria e sufficiente* per passare l'esame (p) è studiare (s) e avere fortuna (f)" può essere rappresentata con $p \equiv (s \wedge f)$

3. La disgiunzione ($\vee$) è l'OR inclusivo (il *vel* latino): $A \vee B$ è vera anche quando sono veri tutti e due i disgiunti. L'OR esclusivo (XOR), la cui tavola di verità è la seguente (rappresentiamo XOR con il simbolo $\oplus$):

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

può essere rappresentato da altre formule. Ad esempio da $(A \vee B) \wedge \neg(A \wedge B)$ (è vera almeno una tra A e B, ma non entrambe), o anche, in modo più compatto, da $\neg(A \equiv B)$ (si faccia la tavola di verità di questa formula e si controlli che si ottiene effettivamente quel che si vuole).

4. L'implicazione ($\rightarrow$) è l'*implicazione materiale*:

$$A \rightarrow B \text{ è falsa se e solo se } A \text{ è vera e } B \text{ è falsa.}$$

In altri termini, $\rightarrow$ corrisponde alla relazione $\leq$ sui booleani (quando si conviene che $F < T$).

Anche se si legge come un "se ... allora", l'implicazione ha un significato che non corrisponde completamente al "se ... allora" del linguaggio naturale. Non c'è necessariamente un rapporto di causa-effetto tra l'antecedente A e il conseguente B (che non si può esprimere soltanto in termini di verità e falsità degli enuncitati).

Il significato logico dell'implicazione è stato oggetto di discussione tra i filosofi e i logici, perché non sempre la riduzione del "se ... allora" in termini di verità e falsità può sembrare adeguata. Essa in ogni caso lo è in logica matematica, dato che tra i fatti matematici non esistono nessi causali o temporali. Per quel che riguarda l'uso della logica per rappresentare conoscenze diverse, si deve aver chiaro che l'unico caso in cui la formula $A \rightarrow B$ è falsa è quello in cui A è vera e B è falsa; dunque, se la formula $A \rightarrow B$ è vera, allora dalla verità di A si può senz'altro derivare che anche B è vera.

Per convincerci comunque della ragionevolezza della definizione di $\rightarrow$ si consideri il caso seguente: Antonio dice "se piove, vengo a prenderti alla stazione". In quali di questi casi (interpretazioni) pensate che Antonio abbia detto una bugia (la sua affermazione è falsa)?

   (a) Piove, e Antonio va alla stazione
   (b) Piove, e Antonio non va alla stazione
   (c) Non piove, e Antonio va lo stesso
   (d) Non piove, e Antonio non va alla stazione

2.5 Soddisfacibilità, validità, equivalenza logica

La definizione che segue introduce alcune nozioni importanti.

Definizione 2.5.1 *Siano A e B formule nel linguaggio $Prop[P]$.*

1. *A è una* **tautologia** *o una formula* **logicamente valida** *sse per ogni interpretazione $\mathcal{M}$ di P, $\mathcal{M} \models A$ (A è vera in ogni interpretazione). Se A è una tautologia, si scrive $\models A$.*

2. *A è* **inconsistente** *(o è una* **contraddizione** *o è* insoddisfacibile*) sse per ogni interpretazione $\mathcal{M}$ di P, $\mathcal{M} \not\models A$ (A è falsa in ogni interpretazione).*

3. *A è* **soddisfacibile** *se esiste un'interpretazione $\mathcal{M}$ tale che $\mathcal{M} \models A$ (A è vera in almeno un'interpretazione).*

4. *Se $A \equiv B$ è una tautologia, allora A e B sono* **logicamente equivalenti**. *In tal caso si scrive $A \leftrightarrow B$.*

Si noti che una formula A è logicamente valida se e solo se $\neg A$ è una contraddizione, e A non è valida se e solo se $\neg A$ è soddisfacibile.

Consideriamo il linguaggio $P = \{p, q\}$ e la formula $A = (p \rightarrow q) \vee q$. A è soddisfacibile, perché esiste un'assegnazione che la soddisfa; ad esempio, $\mathcal{M}$ tale che $\mathcal{M}(p) = F$ e $\mathcal{M}(q) = F$ è un modello di A. Tuttavia A non è valida, infatti, per $\mathcal{M}'$ è tale che $\mathcal{M}'(p) = T$ e $\mathcal{M}'(q) = F$, si ha $\mathcal{M}' \not\models A$ ($\mathcal{M}'$ ̀dunque un contromodello di A).

Per dimostrare che una formula non è valida, è sufficiente trovare un suo contromodello. Ciò si può fare costruendo la tavola di verità della formula e controllando l'esistenza di almeno una riga in cui la formula è falsa, oppure mediante la ricerca diretta di un contromodello. Illustriamo questo secondo metodo mediante un esempio. Sia $A = (p \vee q) \rightarrow (p \rightarrow q)$ e cerchiamo di definire un'assegnazione $\mathcal{M} : \{p, q\} \rightarrow Bool$ tale che $\mathcal{M} \not\models A$. Ora, per avere $\mathcal{M} \not\models A$, si deve avere $\mathcal{M} \models p \vee q$ e $\mathcal{M} \not\models p \rightarrow q$; questo è infatti l'unico caso in cui un'implicazione è falsa. Per avere $\mathcal{M} \not\models p \rightarrow q$, di nuovo, deve essere $\mathcal{M} \models p$ e $\mathcal{M} \not\models q$, quindi $\mathcal{M}(p) = T$ e $\mathcal{M}(q) = F$. Verifichiamo allora che $\mathcal{M} \models p \vee q$; questo segue direttamente da $\mathcal{M} \models p$. Quindi $\mathcal{M}$ è un contromodello di A.

Per dimostrare invece che una formula è una tautologia, si possono controllare tutte le assegnazioni di valori di verità alle sue lettere proposizionali e verificare che la formula è vera in ogni interpretazione. Questo controllo può essere effettuato costruendo la tavola di verità della formula.

Un metodo alternativo per dimostrare la validità di una formula ricorre al ragionamento per assurdo: si dimostra l'impossibilità che la formula abbia un contromodello. Ad esempio, per dimostrare che la formula $\neg(p \vee \neg q) \rightarrow (\neg p \wedge q)$ è valida, assumiamo che esista un'interpretazione $\mathcal{M}$ in cui essa è falsa. Perché tale formula sia falsa, si deve avere $\mathcal{M} \models \neg(p \vee \neg q)$ e $\mathcal{M} \not\models \neg p \wedge q$. Da una parte, perché $\mathcal{M} \models \neg(p \vee \neg q)$, deve essere $\mathcal{M} \not\models p \vee \neg q$, dunque $\mathcal{M} \not\models p$, cioè (1) $\mathcal{M}(p) = F$, e $\mathcal{M} \not\models \neg q$, cioè (2) $\mathcal{M}(q) = T$. D'altro canto, se deve essere $\mathcal{M} \not\models \neg p \wedge q$, si deve avere $\mathcal{M} \not\models \neg p$, cioè $\mathcal{M}(p) = T$, oppure $\mathcal{M}(q) = F$. Ma il primo caso è impossibile per (1), il secondo per (2). Quindi non esiste alcuna assegnazione $\mathcal{M}$ tale che $\mathcal{M} \not\models \neg(p \vee \neg q) \rightarrow (\neg p \wedge q)$.

Lo stesso metodo si può adottare quando non si sa se una formula è valida o no: si cerca di costruirne un contromodello e, se il tentativo ha successo, allora la formula non è valida, altrimenti se si dimostra l'impossibilità di costruirne un contromodello, allora la formula è valida.

Ecco alcune tautologie importanti:

1. Identità: $A \rightarrow A$

2. Affermazione del conseguente: $A \rightarrow (B \rightarrow A)$

3. Negazione dell'antecedente: $\neg A \rightarrow (A \rightarrow B)$

4. *Ex falso quodlibet*: $\perp \rightarrow B$

5. Terzo escluso: $A \vee \neg A$

6. Non contraddizione: $\neg(A \wedge \neg A)$

7. Riduzione all'assurdo: $(A{\rightarrow}B){\rightarrow}((A{\rightarrow}\neg B){\rightarrow}\neg A)$
8. Legge di Pierce: $((A{\rightarrow}B){\rightarrow}A){\rightarrow}A$

Ed alcune equivalenze logiche importanti:

1. Leggi di De Morgan:
$$\begin{array}{lcl}\neg(A \vee B) & \leftrightarrow & \neg A \wedge \neg B \\ \neg(A \wedge B) & \leftrightarrow & \neg A \vee \neg B\end{array}$$
2. Leggi distributive:
$$\begin{array}{lcl}(A \vee (B \wedge C)) & \leftrightarrow & ((A \vee B) \wedge (A \vee C)) \\ (A \wedge (B \vee C)) & \leftrightarrow & ((A \wedge B) \vee (A \wedge C))\end{array}$$
3. Commutatività e associatività di $\wedge$ e $\vee$:
$$\begin{array}{rcll}A \wedge B & \leftrightarrow & B \wedge A & \text{commutatività di } \wedge \\ A \vee B & \leftrightarrow & B \vee A & \text{commutatività di } \vee \\ A \wedge (B \wedge C) & \leftrightarrow & (A \wedge B) \wedge C & \text{associatività di } \wedge \\ A \vee (B \vee C) & \leftrightarrow & (A \vee B) \vee C & \text{associatività di } \vee\end{array}$$
4. Semplificazioni:
$$\begin{array}{lcl}A \wedge \neg A & \leftrightarrow & \bot \\ A \vee \neg A & \leftrightarrow & \top \\ A \wedge \top & \leftrightarrow & A \\ A \vee \bot & \leftrightarrow & A \\ A \wedge \bot & \leftrightarrow & \bot \\ A \vee \top & \leftrightarrow & \top\end{array}$$
5. Doppia negazione: $A \leftrightarrow \neg\neg A$
6. Leggi di assorbimento:
$$\begin{array}{lcl}A \vee (A \wedge B) & \leftrightarrow & A \\ A \wedge (A \vee B) & \leftrightarrow & A\end{array}$$
7. Definibilità di $\equiv$: $A \equiv B \leftrightarrow (A{\rightarrow}B) \wedge (B{\rightarrow}A)$
8. Interdefinibilità dei connettivi logici $\rightarrow, \wedge, \vee$:
$$\begin{array}{ccccc}A{\rightarrow}B & \leftrightarrow & \neg A \vee B & \leftrightarrow & \neg(A \wedge \neg B) \\ \neg(A{\rightarrow}B) & \leftrightarrow & \neg(\neg A \vee B) & \leftrightarrow & A \wedge \neg B \\ A \wedge B & \leftrightarrow & \neg(\neg A \vee \neg B) & \leftrightarrow & \neg(A{\rightarrow}\neg B) \\ A \vee B & \leftrightarrow & \neg(\neg A \wedge \neg B) & \leftrightarrow & \neg A{\rightarrow}B\end{array}$$
9. Contrapposizione: $(A{\rightarrow}B) \leftrightarrow (\neg B{\rightarrow}\neg A)$

2.6 Conseguenza logica

La definizione che segue introduce l'importante nozione di *conseguenza logica*, che è quella che occorre per formalizzare il concetto di "ragionamento corretto".

Definizione 2.6.1 *Sia A una formula e S un insieme di formule nel linguaggio $Prop[P]$. Se A è vera in ogni interpretazione in cui sono vere tutte le formule di S, allora si dice che A è una* **conseguenza logica** *di S (o che S* **implica logicamente** *A), e si scrive $S \models A$. In altri termini, $S \models A$ sse per ogni interpretazione $\mathcal{M}$, se $\mathcal{M} \models S$, allora $\mathcal{M} \models A$.*

D'ora in avanti useremo la notazione $B_1, ..., B_n \models A$, anziché $\{B_1, ..., B_n\} \models A$. Se A non è una conseguenza logica di $\{B_1, ..., B_n\}$, scriviamo $B_1, ..., B_n \not\models A$.

La nozione di conseguenza logica, come abbiamo detto, formalizza quella di "ragionamento corretto". Consideriamo ad esempio il ragionamento già visto nel paragrafo 1:

a) Se nevica, la temperatura è di 0^oC.
   Nevica.
   Quindi la temperatura è di 0^oC.

La forma di tale ragionamento è:

$$\frac{A \rightarrow B \qquad A}{B}$$

Il ragionamento è corretto perché la conclusione B è una conseguenza logica di $\{A \rightarrow B, A\}$:

$$A \rightarrow B, A \models B$$

Per dimostrarlo, possiamo costruire la tavola di verità delle premesse e della conclusione e verificando che il valore della conclusione è T in ogni riga in cui sono vere tutte le premesse:

A	B	$A \rightarrow B$	A	B
T	T	T	T	T
T	F	F	T	F
F	T	T	F	T
F	F	T	F	F

L'unica riga in cui sono vere entrambe le premesse è la prima, e in tale riga è vera anche la conclusione. Quindi $A \rightarrow B, A \models B$.

Il metodo delle tavole di verità diventa però piuttosto complesso al crescere del numero di variabili proposizionali. In alternativa, possiamo ricorrere ad un ragionamento semantico, dimostrando direttamente che ogni modello delle premesse è necessariamente un modello della conclusione. Nel nostro caso, assumiamo che $\mathcal{M}$ sia una qualsiasi interpretazione tale che $\mathcal{M} \models A \rightarrow B$ e $\mathcal{M} \models A$. Per la definizione di $\models$ (Definizione 2.2.2), se $\mathcal{M} \models A \rightarrow B$, allora $\mathcal{M} \not\models A$ oppure $\mathcal{M} \models B$. Dato che, per ipotesi, $\mathcal{M} \models A$, si deve avere necessariamente che $\mathcal{M} \models B$. Quindi ogni modello di $\{A \rightarrow B, A\}$ è anche un modello di B.

In generale, dunque, per dimostrare che un ragionamento è corretto si determina innanzitutto un linguaggio in cui rappresentarlo e si "formalizza" il ragionamento, cioè se ne determina la forma. Infine, si dimostra che la conclusione del ragionamento è una conseguenza logica delle premesse.

Per dimostrare invece che un ragionamento non è corretto, dopo averlo formalizzato, si dimostra che la conclusione non è una conseguenza logica delle premesse, trovando un'interpretazione che è un modello delle premesse e un contromodello della conclusione.

Consideriamo ad esempio il ragionamento seguente:

Se piove, prendo l'ombrello. Prendo l'ombrello. Dunque piove

Se rappresentiamo piove con la lettera proposizionale A e prendo l'ombrello con B, la forma di tale ragionamento è:

$$\frac{A \rightarrow B \qquad B}{A}$$

Il ragionamento non è corretto: infatti se $\mathcal{M}(A) = F$ e $\mathcal{M}(B) = T$, $\mathcal{M} \models \{A \rightarrow B, B\}$ e $\mathcal{M} \not\models A$. Dunque $A \rightarrow B, B \not\models A$.

Consideriamo ancora un altro esempio, tratto da [10].

È stato compiuto un furto. Si sa che:

1. sono implicati tre uomini: Antonio, Biagio, Carlo;
2. i ladri sono fuggiti con un furgone;
3. Carlo non lavora mai senza la complicità di Antonio;
4. Biagio non sa guidare.

Il problema è: Antonio è colpevole?

Iniziamo con la formalizzazione delle informazioni in un linguaggio proposizionale. Utilizziamo le variabili proposizionali A, B, C, con il seguente significato:

A**:** Antonio è colpevole

B**:** Biagio è colpevole

C**:** Carlo è colpevole

L'informazione (1) si può riformulare con "Almeno uno tra Antonio, Biagio e Carlo è colpevole", quindi, nel nostro linguaggio:

F_1**:** $A \vee B \vee C$

L'informazione (3) è rappresentabile da:

F_2**:** $C \rightarrow A$

(se Carlo è colpevole, allora Antonio è suo complice). Infine, la (2) e la (4) assieme si possono riscrivere in "se Biagio è colpevole, allora anche qualcun altro è colpevole, dato che lui non poteva guidare il furgone"; quindi:

F_3**:** $B \rightarrow (A \vee C)$

Il problema è dunque quello di determinare se

$$F_1, F_2, F_3 \models A$$

Ovviamente, possiamo fare la tavola di verità per le formule F_1, F_2, F_3 e verificare se A è vera in ogni riga in cui esse sono tutte vere. Ma non è questo il modo naturale di ragionare.

Possiamo dedurre la colpevolezza di Antonio ragionando per assurdo: sia $\mathcal{M}$ una qualsiasi interpretazione in cui sono vere F_1, F_2 e F_3, cioè

$$(1)\ \mathcal{M} \models A \vee B \vee C, \quad (2)\ \mathcal{M} \models C \rightarrow A, \quad (3)\ \mathcal{M} \models B \rightarrow (A \vee C)$$

e tuttavia

$$(4)\ \mathcal{M} \not\models A$$

Allora, da (2) e (4) segue

$$(5)\ \mathcal{M} \not\models C$$

per la semantica di $\rightarrow$. Per il significato di $\vee$, da (4) e (5) segue

$$(6)\ \mathcal{M} \not\models A \vee C$$

e, per il significato di $\rightarrow$, da (3) e (6) segue

$$(7)\ \mathcal{M} \not\models B$$

Ma allora, da (4), (5) e (7) segue

$$(8)\ \mathcal{M} \not\models A \vee B \vee C$$

contraddicendo l'ipotesi (1). Poiché dunque assumere l'innocenza di Antonio porta a una contraddizione con le informazioni date, se ne deduce che Antonio è colpevole.

Lo stesso risultato si può seguire ragionando per casi dall'ipotesi che $A \vee B \vee C$ sia vera in $\mathcal{M}$. Secondo questa ipotesi, i casi sono tre: o è colpevole Antonio, oppure è colpevole Biagio oppure è colpevole Carlo. Dimostriamo allora che in tutti e tre i casi Antonio è colpevole. Ne potremo concludere che Antonio è comunque colpevole.

caso 1. Se è vero A, banalmente ne segue che è vero A.

caso 2. Se è vero B, allora, per l'ipotesi (3), è vero anche $A \vee C$; di nuovo, ragioniamo per casi:

sottocaso 2a. se è vero A, allora è vero A;

sottocaso 2b. se è vero C, poiché è vero anche $C \rightarrow A$ (2), allora è vero A.

Quindi possiamo concludere che, nel caso 2, A è comunque vero.

caso 3. Se è vero C, poiché è vero anche $C \rightarrow A$, allora è vero A.

In ogni possibile caso si ha dunque che A è vero. Se ne conclude allora che A è comunque vero: Antonio è colpevole.

2.7 L'isola dei cavalieri e furfanti

In questo paragrafo vediamo come utilizzare la logica dei predicati per risolvere alcuni dei quiz nel libro di R. Smullyan [10], ambientati nell'isola dei cavalieri e furfanti. Quest'isola è abitata da due tipi di persone: i cavalieri, che dicono sempre la verità, e i furfanti, che dicono sempre bugie. Noi siamo visitatori dell'isola e incontriamo due persone, che chiamiamo A e B. A dice: "Io sono un furfante oppure B è un cavaliere". Cosa sono A e B?

Vediamo come rappresentare adeguatamente questo problema in logica proposizionale. Determiniamo innanzitutto un linguaggio appropriato. Utilizziamo a tale scopo due lettere proposizionali: A, che rappresenta l'enunciato "A è un cavaliere", e B, che rappresenta l'enunciato "B è un cavaliere". Poiché nell'isola, eccetto noi, ci sono solo cavalieri e furfanti, l'enunciato "A è un furfante" sarà rappresentato da $\neg A$, e analogamente $\neg B$ rappresenta "B è un furfante".

L'affermazione di A è allora rappresentata dalla formula $\neg A \vee B$. Ma noi non sappiamo se questo è vero o falso, perché A potrebbe essere un furfante. Sappiamo solo che se A è un cavaliere allora $\neg A \vee B$ è vera, e se A è un furfante allora $\neg A \vee B$ è falsa. La nostra conoscenza è allora rappresentata dalle due formule:

1. $A \rightarrow (\neg A \vee B)$
2. $\neg A \rightarrow \neg(\neg A \vee B)$

Mediante le tavole di verità possiamo determinare quali sono le interpretazioni di A e B in cui le due formule 1 e 2 sono entrambe vere, e rispondere dunque alla domanda "cosa sono A e B?":

A	B	$A \rightarrow (\neg A \vee B)$	$\neg A \rightarrow \neg(\neg A \vee B)$
T	T	T	T
T	F	F	T
F	T	T	F
F	F	T	F

L'unica interpetazione in cui 1 e 2 sono entrambe vere è quella in cui A e B sono entrambe vere: A e B sono entrambi cavalieri.

In altri termini, A e B sono conseguenze logiche della nostra conoscenza:

$$A \to (\neg A \vee B), \neg A \to \neg(\neg A \vee B) \models A$$

$$A \to (\neg A \vee B), \neg A \to \neg(\neg A \vee B) \models B$$

Anziché utilizzare le tavole di verità per risolvere problemi di questo tipo, possiamo utilizzare un ragionamento semantico, come nell'esempio del paragrafo precedente. Consideriamo lo stesso problema visto sopra. Per determinare se A è una conseguenza logica di 1 e 2, dimostriamo che tutti i modelli di 1 e 2 sono modelli di A. Sia $\mathcal{M}$ una qualunque interpretazione che rende vere 1 e 2 e supponiamo (per assurdo) che $\mathcal{M} \not\models A$.

$$\begin{array}{llll}
\mathcal{M} \not\models A & \Rightarrow & \mathcal{M} \models \neg A & \text{def. di } \neg \\
& \Rightarrow & \mathcal{M} \models \neg(\neg A \vee B) & \text{2 e def. di } \to \\
& \Rightarrow & \mathcal{M} \not\models \neg A \vee B & \\
& \Rightarrow & \mathcal{M} \not\models \neg A & \text{def. di } \vee \\
& \Rightarrow & \mathcal{M} \models A : \text{contraddizione} &
\end{array}$$

Se fosse $\mathcal{M} \not\models A$, si avrebbe dunque contemporaneamente $\mathcal{M} \models A$ e $\mathcal{M} \not\models A$. Quindi $\mathcal{M} \not\models A$ è assurdo e deve essere $\mathcal{M} \models A$.

Con lo stesso metodo si può determinare che B è una conseguenza logica di 1 e 2.

2.8 Teorema di sostituzione

Utilizzeremo spesso in seguito, a volte in modo implicito, una proprietà importante della semantica della logica proposizionale, la sostituibilità di formule equivalenti all'interno di una formula. Tale proprietà, affermata dal teorema che segue, ci permette ad esempio di affermare che $B \to \neg\neg A$ è logicamente equivalente a $B \to A$, sulla base del fatto che $A \leftrightarrow \neg\neg A$.

Teorema 2.1 (Teorema di sostituzione per la logica proposizionale)
Se $B \leftrightarrow B'$ e $A[B']$ si ottiene da A sostituendo qualche occorrenza di B con B', allora $A \leftrightarrow A[B']$

Dimostrazione. Dimostriamo che, per ogni interpretazione $\mathcal{M}$, $\mathcal{M} \models A$ sse $\mathcal{M} \models A[B']$. Consideriamo innanzitutto un caso speciale, quello in cui $A = B$ e $A[B'] = B'$: la tesi segue allora immediatamente dall'ipotesi $B \leftrightarrow B'$.

Se non siamo nel caso speciale, la dimostrazione è per induzione su A:

(B) Se A è atomica e non siamo nel caso speciale, allora $A = A[B']$.

(PI) Consideriamo diversi casi, a seconda della forma di A. Trattiamo in dettaglio soltanto il caso della negazione e congiunzione. I casi degli altri connettivi binari sono simili a quello della congiunzione.

1. $A = \neg C$: se non siamo nel caso speciale, $A[B'] = \neg C[B']$. Per ipotesi induttiva $\mathcal{M} \models C$ sse $\mathcal{M} \models C[B']$, quindi $\mathcal{M} \models \neg C$ sse $\mathcal{M} \models \neg C[B']$.
2. Se $A = C_1 \wedge C_2$, allora $A[B'] = C_1[B'] \wedge C_2[B']$. Per ipotesi induttiva $\mathcal{M} \models C_1$ sse $\mathcal{M} \models C_1[B']$, e $\mathcal{M} \models C_2$ sse $\mathcal{M} \models C_2[B']$, quindi $\mathcal{M} \models C_1 \wedge C_2$ sse $\mathcal{M} \models C_1[B'] \wedge C_2[B']$.

2.9 Funzioni di verità e adeguatezza dei connettivi

Qualsiasi formula proposizionale contenente k lettere proposizionali genera una corrispondente **funzione di verità** a k argomenti: $Bool^k \longrightarrow Bool$. In questo paragrafo utilizziamo la notazione $\overline{\mathcal{M}}(A)$ per indicare il valore di verità di A secondo l'interpretazione $\mathcal{M}$:

$$\overline{\mathcal{M}}(A) = \begin{cases} T & \text{se } \mathcal{M} \models A \\ F & \text{altrimenti} \end{cases}$$

Per denotare la funzione corrispondente a una formula, useremo la notazione seguente. Se A è una formula e $p_1, \cdots, p_k$ sono variabili proposizionali distinte che includono tutte quelle che occorrono in A, denotiamo con

$$\lambda(p_1, \cdots, p_k).A$$

la funzione booleana $f:\ Bool^k \longrightarrow Bool$, tale che per ogni $x_1, \cdots, x_k \in Bool$:

$$f(x_1, \cdots, x_k) = \overline{\mathcal{M}}(A)$$

dove $\mathcal{M}(p_i) = x_i$, per $1 \leq i \leq k$. In altri termini, f è la funzione tale che per ogni interpretazione $\mathcal{M}$ di $\{p_1, ..., p_n\}$:

$$f(\mathcal{M}(p_1), ..., \mathcal{M}(p_n)) = \overline{\mathcal{M}}(A)$$

Ad esempio, se poniamo $f = \lambda(p, q).(p \vee \neg q)$, allora f è la funzione booleana tale che:

$$\begin{array}{ll} f(T,T) = T & f(T,F) = T \\ f(F,T) = F & f(F,F) = T \end{array}$$

Formule logicamente equivalenti generano la stessa funzione di verità: $\lambda(p_1, \cdots, p_k).A$ e $\lambda(p_1, \cdots, p_k).B$ sono funzioni (estensionalmente) uguali se e soltanto se $\models A \equiv B$.

Il teorema seguente assicura che, mediante i connettivi logici $\neg, \vee$ e $\wedge$ vengono generate tutte le funzioni di verità.

Teorema 2.9.1 (Adeguatezza dei connettivi) *Sia f una funzione booleana a n argomenti:*

$$f : Bool^n \longrightarrow Bool$$

e siano $p_1, \cdots, p_n$ variabili proposizionali distinte. Allora esiste una formula $A_f \in Prop[p_1, ..., p_n]$, nella quale occorrono soltanto i connettivi $\neg, \wedge$ e $\vee$ e tale che

$$f = \lambda(p_1, \cdots, p_n).A_f$$

Ciò significa che $\{\neg, \wedge, \vee\}$ è un insieme adeguato di connettivi.

Anziché dare una dimostrazione generale di questo teorema, mostriamo mediante un esempio come costruire, a partire da una funzione booleana, una formula che la rappresenta. Osserviamo innanzitutto che una funzione booleana a n argomenti si può descrivere mediante una tabella con 2^n righe, ciascuna corrispondente a una possibile n-pla di argomenti per f. Ad esempio, la tabella che segue descrive una funzione di verità a due argomenti:

x_1	x_2	$f(x_1, x_2)$
T	T	F
F	T	T
T	F	T
F	F	T

Ogni riga della tabella corrisponde ad una possibile assegnazione di valori di verità a una coppia di variabili proposizionali, diciamo p e q. La prima riga corrisponde ad una $\mathcal{M}_1$ tale che $f(\mathcal{M}_1(p), \mathcal{M}_1(q)) = F$. Le altre tre invece sono assegnazioni $\mathcal{M}_k$ $(2 \geq k \geq 4)$ tali che $f(\mathcal{M}_k(p), \mathcal{M}_k(q)) = T$. Consideriamo allora soltanto queste tre ultime e costruiamo le corrispondenti congiunzioni C_k:

$$\begin{array}{c} C_2 = p^{\mathcal{M}_2} \wedge q^{\mathcal{M}_2} = (\neg p \wedge q) \\ C_3 = p^{\mathcal{M}_3} \wedge q^{\mathcal{M}_3} = (p \wedge \neg q) \\ C_4 = p^{\mathcal{M}_4} \wedge q^{\mathcal{M}_4} = (\neg p \wedge \neg q) \end{array}$$

La formula che genera f è dunque: $(\neg p \wedge q) \vee (p \wedge \neg q) \vee (\neg p \wedge \neg q)$.

Il metodo di costruzione qui illustrato non produce necessariamente la formula più semplice che corrisponde alla funzione booleana considerata. Ad esempio, la formula che genera f può essere semplificata: utilizzando la legge distributiva

$$(\neg p \wedge (q \vee \neg q)) \;\leftrightarrow\; (\neg p \wedge q) \vee (\neg p \wedge \neg q)$$

e l'associatività e la commutatività della congiunzione, vediamo che essa è equivalente a:

$$(\neg p \wedge (q \vee \neg q)) \vee (p \wedge \neg q)$$

Quest'ultima è equivalente a:

$$\neg p \vee (p \wedge \neg q)$$

Applicando ancora una legge distributiva, otteniamo la formula equivalente:

$$(\neg p \vee p) \wedge (\neg p \vee \neg q)$$

che a sua volta equivale a:

$$\neg p \vee \neg q$$

2.10 Forme normali disgiuntive e congiuntive

Ogni formula è equivalente ad altre formule che hanno una forma particolare, o *normale.*

Definizione 2.10.1 *Un* **letterale** *è una lettera proposizionale o la negazione di una lettera proposizionale. Una formula è in* **forma normale disgiuntiva** *(DNF:* Disjunctive Normal Form*) se è una disgiunzione di congiunzioni di letterali. È in* **forma normale congiuntiva** *(CND:* Conjunctive Normal Form*) se è una congiunzione di disgiunzioni di letterali.*

Teorema 2.2 *Ogni formula è equivalente ad una formula in forma normale disgiuntiva ed è equivalente ad una formula in forma normale congiuntiva.*

La trasformazione di una formula in DNF o CNF può essere eseguita mediante trasformazioni successive, applicando le seguenti regole che trasformano una formula in una formula equivalente:

1. Eliminare le implicazioni e le doppie implicazioni mediante le seguenti trasformazioni:

$$\begin{array}{rcl} A \equiv B & \Longrightarrow & (A \wedge B) \vee (\neg A \wedge \neg B) \\ A \to B & \Longrightarrow & \neg A \vee B \\ \neg(A \to B) & \Longrightarrow & A \wedge \neg B \\ \neg(A \equiv B) & \Longrightarrow & (A \vee B) \wedge (\neg A \vee \neg B) \end{array}$$

2. Portare le negazioni sugli atomi, applicando le leggi di De Morgan e la legge di doppia negazione:

$$\begin{array}{rcl} \neg(A \vee B) & \Longrightarrow & (\neg A \wedge \neg B) \\ \neg(A \wedge B) & \Longrightarrow & (\neg A \vee \neg B) \\ \neg\neg A & \Longrightarrow & A \end{array}$$

   Si ottiene così una formula in *forma normale negativa*.

3. Applicare le leggi distributive:

$$\begin{array}{rcl} (A \vee (B \wedge C)) & \Longrightarrow & ((A \vee B) \wedge (A \vee C)) \\ (A \wedge (B \vee C)) & \Longrightarrow & ((A \wedge B) \vee (A \wedge C)) \end{array}$$

Ad esempio, possiamo trasformare in CNF la formula

$$\neg(\neg(p \vee \neg q) \to \neg(q \vee (r \wedge (s \to p))))$$

con i seguenti passaggi:

$$\begin{array}{rl} \multicolumn{2}{l}{\neg(\neg(p \vee \neg q) \to \neg(q \vee (r \wedge (s \to p))))} \\ \Rightarrow & \neg(p \vee \neg q) \wedge \neg\neg(q \vee (r \wedge (\neg s \vee p))) \\ \Rightarrow & \neg(p \vee \neg q) \wedge (q \vee (r \wedge (\neg s \vee p))) \\ \Rightarrow & (\neg p \wedge \neg\neg q) \wedge (q \vee (r \wedge (\neg s \vee p))) \\ \Rightarrow & (\neg p \wedge q) \wedge (q \vee (r \wedge (\neg s \vee p))) \\ \Rightarrow & (\neg p \wedge q) \wedge ((q \vee r) \wedge (q \vee (\neg s \vee p))) \\ = & \neg p \wedge q \wedge (q \vee r) \wedge (q \vee \neg s \vee p) \end{array}$$

Si noti che qualsiasi formula ha più di una forma normale congiuntiva o disgiuntiva, sintatticamente distinte anche se logicamente equivalenti.

2.11 Esercizi

1. Rappresentare le affermazioni seguenti mediante formule proposizionali:
   (a) Se l'umidità è elevata, pioverà questo pomeriggio o questa sera.
   (b) La mancia sarà pagata solo se il servizio è di qualità.
   (c) Il Cagliari vincerà lo scudetto, a meno che oggi non vinca l'Inter [5].
   (d) Condizione necessaria e sufficiente affinché uno sceicco sia felice è avere vino, donne e canti [5].

2. Dimostrare che sono tautologie le formule riportate a pagina 15.

3. Dimostrare le equivalenze logiche riportate a pagina 16.

4. Per ciascuna delle formule seguenti determinare se è una tautologia, se è soddisfacibile, se è inconsistente.
   (a) $(p \to q) \vee \neg p$
   (b) $(p \to (q \to r)) \to ((p \to q) \to (p \to r))$
   (c) $\neg\neg p \to p$
   (d) $p \vee (p \to q)$
   (e) $(p \to q) \equiv (\neg p \vee q)$
   (f) $\neg p \to p$

(g) $p \to \neg p$

(h) $\neg(p \vee q) \vee (\neg q)$

(i) $p \vee (q \to \neg p)$

(j) $p \to \neg(p \vee q)$

(k) $\neg p \wedge \neg(p \to q)$

(l) $((p \to q) \to q) \to q$

(m) $p \equiv (p \vee p)$

(n) $p \equiv (p \wedge p)$

(o) $\neg p \to (p \to q)$

(p) $(p \to q) \equiv ((\neg p) \vee q)$

(q) $(p \to q) \equiv \neg(p \wedge \neg q)$

(r) $\neg p \to (p \wedge q)$

(s) $p \wedge (p \vee q) \equiv p$

(t) $p \vee (p \wedge q) \equiv p$

5. Un uomo veniva processato per furto. Il pubblico ministero e l'avvocato difensore fecero le seguenti affermazioni.
   Pubblico Ministero: Se l'imputato è colpevole, allora ebbe un complice.
   Avvocato difensore: Non è vero!
   Perché questa fu la cosa peggiore che l'avvocato difensore potesse dire? ([10])

6. Determinare se i seguenti ragionamenti sono corretti.

   (a) Se i controllori di volo scioperano, allora prenderò il treno. O scioperano i controllori, oppure scioperano i piloti. Quindi prenderò il treno.

   (b) Se i controllori di volo scioperano, allora prenderò il treno. Se i piloti scioperano, allora prenderò il treno. O scioperano i controllori, oppure scioperano i piloti. Quindi prenderò il treno.

   (c) Se $\overset{\triangle}{ABC}$ è un triangolo equilatero, allora è un poligono regolare. Se $\overset{\triangle}{ABC}$ è un triangolo isoscele, allora è un poligono regolare. $\overset{\triangle}{ABC}$ è un triangolo equilatero oppure isoscele. Quindi $\overset{\triangle}{ABC}$ è un poligono regolare.

   (d) Se piove Antonio non va a scuola. Se Antonio non va a scuola, la mamma e la maestra si inquietano. Quindi, se piove la maestra si inquieta.

7. Nell'isola dei cavalieri e furfanti [10, 6]:

   (a) Antonio dice "io sono un furfante". Cos'è Antonio?

   (b) Incontriamo due abitanti, A e B. A dice: "Almeno uno di noi è un furfante". Cosa sono A e B?

   (c) Incontriamo due abitanti, A e B. A dice: "Io sono un furfante ma B non lo è". Cosa sono A e B?

   (d) A dice: "io e mia sorella siamo dello stesso tipo". Scoprite il tipo di almeno uno dei due.

   (e) A dice: "B e C sono entrambi cavalieri". Poi gli chiedete: "È vero che B è un cavaliere?" e A risponde: "No". Cos'è A e cosa sono B e C?

(f) Chiedete ad A: "B e C sono entrambi cavalieri?" Risponde di no. Poi gli chiedete: "B è un cavaliere?" Risponde: "Si". Cosa sono A, B e C?

(g) Ci sono tre abitanti, A, B e C. A dice: "Siamo tutti furfanti". B dice: "Esattamente uno di noi è un cavaliere". Cosa sono A, B e C?

(h) Ci sono tre persone: A, B e C. A dice: "B è un furfante". B dice: "A e C sono dello stesso tipo". Cos'è C?

(i) Ci sono tre persone: A, B e C. A dice: "B e C sono dello stesso tipo". Poi chiedete a C: "A e B sono dello stesso tipo?". Cosa risponde C?

(j) A dice: "Io sono un furfante oppure 2+2=5". Cosa si può concludere?

(k) D vi ruba la chiave della macchina e la mette in uno di 3 cassetti: A, B, C. Gli chiedete: "È vero che la chiave è in A o in B?" Risponde di no. Chiedete allora: "È nel cassetto A?" Risponde di sì. Dov'è la chiave?

(l) Tornate in albergo e non vi ricordate con precisione il numero della vostra stanza, che comunque deve essere il 33 o il 35. Chiedete al portiere: "È vero che il numero della mia stanza è il 33 o il 35?" Risponde di si. Poi gli chiedete: "è vero che è il 33?" e dice di no. Qual è il numero della vostra stanza?

8. Trovare forme normali disgiuntive e congiuntive equivalenti a:

   (a) $(\neg p \wedge q) \rightarrow r$

   (b) $p \rightarrow ((q \wedge r) \rightarrow s)$

   (c) $p \vee (\neg p \wedge q \wedge (r \vee q))$

   (d) $(p \rightarrow q) \rightarrow r$

9. È possibile che una formula sia contemporaneamente in CNF e DNF?

10. Sono corretti i seguenti ragionamenti? Per rispondere seguire la procedura seguente:

> Formalizzare innanzitutto il ragionamento. In seguito:
>
> - Per dimostrare che un ragionamento è corretto, si possono seguire due metodi:
>
>   (1) dare una dimostrazione semantica. La dimostrazione può essere diretta (supponiamo che esista un'interpretazione $\mathcal{M}$ in cui sono vere tutte le premesse, allora ... in $\mathcal{M}$ è vera anche la conclusione) oppure indiretta (supponiamo che esista un'interpretazione in cui sono vere tutte le premesse e falsa la conclusione, allora .... ASSURDO). Un terzo metodo semantico è quello che fa uso delle tavole di verità, ma cercate di evitarlo.
>
>   (2) dare una dimostrazione sintattica, utilizzando il metodo dei tableaux.
>
> - Per dimostrare che un ragionamento non è corretto può essere utilizzato un unico metodo: definire un'interpretazione in cui sono vere tutte le premesse e falsa la conclusione. Per far questo, è comunque possibile utilizzare il metodo dei tableaux.

(a) Se piove, o prendo l'ombrello o resto a casa. Se resto a casa non mi bagno. Quindi non mi bagno.

(b) Se piove, o prendo l'ombrello o resto a casa. Se resto a casa, sono al coperto. Se prendo l'ombrello, sono al coperto. Non mi bagno solo se sono al coperto. Quindi non mi bagno.

(c) Se piove, l'erba è bagnata. Non piove. Quindi l'erba è asciutta.

(d) Se piove, l'erba è bagnata. L'erba è bagnata. Quindi piove.

(e) Se Napoleone fosse tedesco, sarebbe asiatico. Napoleone non è asiatico. Quindi non è tedesco.

(f) Se l'investimento di capitali rimane costante, allora cresceranno le spese del governo oppure ci sarà disoccupazione. Se non aumenteranno le spese del governo, allora potranno essere ridotte le tasse. Se le tasse potranno essere ridotte e l'investimento di capitali rimane costante, allora non si avranno fenomeni di disoccupazione. Quindi aumenteranno le spese del governo [5].

(g) Se Bianchi non ha incontrato Rossi la notte scorsa, allora o Rossi era l'assassino, oppure Bianchi mente. Se Rossi non era l'assassino, allora Bianchi non ha incontrato Rossi la notte scorsa e il delitto è avvenuto dopo la mezzanotte. Se il delitto è avvenuto dopo la mezzanotte, allora o Rossi era l'assassino o Bianchi mente. Quindi Rossi era l'assassino [5].

(h) Aggiungere alle ipotesi di (10g): Se Bianchi mente, allora Bianchi ha incontrato Rossi la notte scorsa

(i) Aggiungere alle ipotesi di (10g): Bianchi non mente

2.12 Soluzione di alcuni esercizi

(1b) L'affermazione "la mancia sarà pagata solo se il servizio è di qualità" si può riformulare con "condizione necessaria perché la mancia venga pagata è che il servizio sia di qualità". In altri termini, se viene pagata la mancia, ciò significa che il servizio è stato di qualità (altrimenti niente mancia). Quindi, se rappresentiamo "la mancia sarà pagata" con l'atomo p e "il servizio è di qualità" con l'atomo q, la formula che rappresenta correttamente l'enunciato è: $p \rightarrow q$. La formula $q \rightarrow p$ rappresenta invece l'enunciato "se il servizio è di qualità, allora la mancia sarà pagata", o, in altri termini "condizione sufficiente perché la mancia venga pagata è che il servizio sia di qualità". Cioè l'implicazione è nel senso inverso rispetto a quello voluto: affermando che la mancia sarà pagata solo se il servizio è di qualità non si afferma che il servizio di qualità garantisce il pagamento della mancia.

(1c) Rappresentiamo "il Cagliari vincerà lo scudetto" con l'atomo p e "oggi vince l'Inter" con l'atomo q. L'enunciato "il Cagliari vincerà lo scudetto, a meno che oggi non vinca l'Inter" afferma che se oggi non vince l'Inter allora certamente il Cagliari vincerà lo scudetto. Dunque: $\neg q \rightarrow p$. Con questa interpretazione di "a meno che" non si esclude il caso in cui il Cagliari vinca comunque lo scudetto nonostante il fatto che oggi vinca l'Inter. Ma le frasi in linguaggio naturale sono spesso ambigue. Con l'enunciato considerato, potremmo intendere anche che la vittoria dell'Inter esclude la possibilità che il Cagliari vinca lo scudetto, allora possiamo riformulare l'enunciato in "il Cagliari vincerà lo scudetto se e solo se oggi non vince l'Inter", dunque la rappresentazione adeguata è in questo caso più forte: $p \equiv \neg q$.

(2) Negazione dell'antecedente (3): per dimostrare che $\models \neg A \rightarrow (A \rightarrow B)$ assumiamo, per assurdo, che esista un'interpretazione $\mathcal{M}$ tale che $\mathcal{M} \not\models \neg A \rightarrow (A \rightarrow B)$. L'unico caso in cui un'implicazione è falsa è quando è vero l'anteceente e falso il conseguente; quindi deve essere (1) $\mathcal{M} \models \neg A$ e (2) $\mathcal{M} \not\models A \rightarrow B$. Da (1) segue (3) $\mathcal{M} \not\models A$ e da (2) seguono

(4) $\mathcal{M} \models A$ e $\mathcal{M} \not\models B$. Ma (3) e (4) costituiscono una contraddizione, quindi non esiste alcuna interpretazione $\mathcal{M}$ tale che $\mathcal{M} \not\models \neg A \rightarrow (A \rightarrow B)$.

Legge di Pierce (8): per mostrare che $\models ((A \rightarrow B) \rightarrow A) \rightarrow A$, ragioniamo di nuovo per assurdo: assumiamo che, per qualche interpretazione $\mathcal{M}$, si abbia $\mathcal{M} \not\models ((A \rightarrow B) \rightarrow A) \rightarrow A$. Dunque (1) $\mathcal{M} \models (A \rightarrow B) \rightarrow A$ e (2) $\mathcal{M} \not\models A$. Ma se è falso il conseguente (2) di un'implicazione, deve essere falso anche l'antecedente, quindi (3) $\mathcal{M} \not\models A \rightarrow B$. Se un'implicazione è falsa, il suo antecedente è vero, quindi deve essere $\mathcal{M} \models A$, contraddicendo (2). Di conseguenza non esiste alcuna interpretazione $\mathcal{M}$ tale che $\mathcal{M} \not\models ((A \rightarrow B) \rightarrow A) \rightarrow A$.

(3) Dimostriamo la prima Legge di De Morgan: $\neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$. Sia $\mathcal{M}$ un'interpretazione qualsiasi. Per il significato della negazione, $\mathcal{M} \models \neg(A \vee B)$ se e solo se $\mathcal{M} \not\models A \vee B$. L'unico caso in cui una disgiunzione è falsa è quando sono falsi entrambi i disgiunti. Quindi $\mathcal{M} \not\models A \vee B$ se e solo se $\mathcal{M} \not\models A$ e $\mathcal{M} \not\models B$. Per la semantica della negazione, ciò vale se e solo se $\mathcal{M} \models \neg A$ e $\mathcal{M} \models \neg B$, che equivale, per il significato della congiunzione, a $\mathcal{M} \models \neg A \wedge \neg B$.

(4a) Cerchiamo di costruire un contromodello di $(p \rightarrow q) \vee \neg p$: se $\mathcal{M} \not\models (p \rightarrow q) \vee \neg p$, deve essere (1) $\mathcal{M} \not\models p \rightarrow q$ e (2) $\mathcal{M} \not\models \neg p$. (1) equivale a dire che $\mathcal{M} \models p$ e $\mathcal{M} \not\models q$ e (2) equivale a $\mathcal{M} \models p$. Dunque, se $\mathcal{M}(p) = T$ e $\mathcal{M}(q) = F$, allora $\mathcal{M} \not\models (p \rightarrow q) \vee \neg p$. Di conseguenza la formula non è valida.

Per verificare se è soddisfacibile occorre determinare un'interpretazione in cui è vera. Perché una disgiunzione sia vera è sufficiente che sia vero uno dei due disgiunti: ad esempio $\mathcal{M} \models \neg p$. Dunque la formula è soddisfatta in qualsiasi interpretazione $\mathcal{M}$ tale che $\mathcal{M}(p) = F$, ed è allora soddisfacibile. Dato che è soddisfacibile, non è una contraddizione.

(4i) Cerchiamo di costruire un'interpretazione $\mathcal{M}$ tale che $\mathcal{M} \not\models p \vee (q \rightarrow \neg p)$. $\mathcal{M}$ deve essere tale che (1) $\mathcal{M} \not\models p$ e (2) $\mathcal{M} \not\models q \rightarrow \neg p$. Da (2) segue che $\mathcal{M} \models q$ e $\mathcal{M} \not\models \neg p$, quindi $\mathcal{M} \models p$, contraddicendo (1). Di conseguenza la formula è valida.

Se è valida, è ovviamente soddisfacibile e non è inconsistente.

(4r) La formula non è certamente valida: se p è falso, $\neg p$ è vero e $p \wedge q$ è falso, quindi $\neg p \rightarrow (p \wedge q)$ è falso.

Verifichiamo se la formula è soddisfacibile. Per far ciò è sufficiente verificare se esiste un'interpretazione in cui l'antecedente è falso. Se $\mathcal{M}(p) = T$, allora $\mathcal{M} \not\models \neg p$, quindi $\mathcal{M} \models \neg p \rightarrow (p \wedge q)$.

(5) Un'implicazione è falsa (affermazione dell'avvocato difensore) solo se l'antecedente è vero (e il conseguente falso): dunque l'imputato è colpevole.

(7) Nella soluzione di questi esercizi rappresentiamo con A, B e C gli enunciati "A è un cavaliere", "B è un cavaliere" e "C è un cavaliere", rispettivamente. Per ciascun esercizio risolto, indichiamo le formule che rappresentano la nostra conoscenza e le rispettive tavole di verità, oppure un ragionamento semantico che consente di risolvere il problema.

(7b) Le formule $A \rightarrow \neg A \vee \neg B$ e $\neg A \rightarrow \neg(\neg A \vee \neg B)$ implicano logicamente A e $\neg B$:

A	B	$A \rightarrow \neg A \vee \neg B$	$\neg A \rightarrow \neg(\neg A \vee \neg B)$
T	T	F	T
T	F	T	T
F	T	T	F
F	F	T	F

(7c) Le formule $A \rightarrow \neg A \wedge B$ e $\neg A \rightarrow \neg(\neg A \wedge B)$ implicano logicamente $\neg A$ e $\neg B$:

A	B	$A \rightarrow \neg A \wedge B$	$\neg A \rightarrow \neg(\neg A \wedge B)$
T	T	F	T
T	F	F	T
F	T	T	F
F	F	T	T

(7g) Le formule che rappresentano la nostra conoscenza sono:

F_1. $A \rightarrow \neg A \wedge \neg B \wedge \neg C$
F_2. $\neg A \rightarrow A \vee B \vee C$
F_3. $B \rightarrow (A \wedge \neg B \wedge \neg C) \vee (\neg A \wedge B \wedge \neg C) \vee (\neg A \wedge \neg B \wedge C)$
F_4. $\neg B \rightarrow (\neg A \wedge \neg B \wedge \neg C) \vee (A \wedge B) \vee (A \wedge C) \vee (B \wedge C)$

Esse implicano logicamente $\neg A$, B e $\neg C$:

A	B	C	F_1	F_2	F_3	F_4
T	T	T	F			
T	T	F	F			
T	F	T	F			
T	F	F	F			
F	T	T	T	T	F	
F	T	F	T	T	T	T
F	F	T	T	T	T	F
F	F	F	T	F		

(7h) La conoscenza può essere rappresentata dalle quattro formule seguenti:

F_1. $A \rightarrow \neg B$
F_2. $\neg A \rightarrow B$
F_3. $B \rightarrow (A \wedge C) \vee (\neg A \wedge \neg C)$
F_4. $\neg B \rightarrow (A \wedge \neg C) \vee (\neg A \wedge C)$

Questa volta risolviamo il problema mediante un ragionamento semantico. Supponiamo che $\mathcal{M}$ sia un'interpretazione in cui sono vere le quattro formule date sopra. Ora, i casi sono due: $\mathcal{M} \models A$ oppure $\mathcal{M} \models \neg A$. Mostriamo che in entrambi i casi $\mathcal{M} \models \neg C$.

* Caso 1: $\mathcal{M} \models A$. Allora, dato che anche $\mathcal{M} \models A \rightarrow \neg B$ (F_1), $\mathcal{M} \models \neg B$, e poichè $\mathcal{M} \models F_4$, $\mathcal{M} \models (A \wedge \neg C) \vee (\neg A \wedge C)$. Quindi i casi sono due: (a) $\mathcal{M} \models A \wedge \neg C$, oppure (b) $\mathcal{M} \models \neg A \wedge C$. Mostriamo che il secondo caso è assurdo: se fosse $\mathcal{M} \models \neg A \wedge C$, allora, per la semantica della congiunzione, $\mathcal{M} \models \neg A$, contraddicendo l'ipotesi del caso 1. Quindi deve essere vero (a) e, sempre per la semantica della congiunzione, $\mathcal{M} \models \neg C$.
* Caso 2: $\mathcal{M} \models \neg A$. Poiché anche $\mathcal{M} \models F_2$, $\mathcal{M} \models B$ e, poiché $\mathcal{M} \models F_3$, si ha $\mathcal{M} \models (A \wedge C) \vee (\neg A \wedge \neg C)$. Quindi abbiamo due casi: (c) $\mathcal{M} \models A \wedge C$, oppure (d) $\mathcal{M} \models \neg A \wedge \neg C$. Mostriamo che il caso (c) è assurdo: in tal caso infatti si avrebbe $\mathcal{M} \models A$, contraddicendo l'ipotesi del caso 2. Quindi deve essere vero (d), dunque $\mathcal{M} \models \neg C$ per il significato della congiunzione.

Poiché in entrambi i casi possibili si ha $\mathcal{M} \models \neg C$, ne segue che $F_1, F_2, F_3, F_4 \models \neg C$.

(7i) Indichiamo in questo caso soltanto come impostare il problema. Le formule che rappresentano la nostra conoscenza sono:

$$\begin{array}{ll} F_1. & A \to (B \wedge C) \vee (\neg B \wedge \neg C) \\ F_2. & \neg A \to (B \wedge \neg C) \vee (\neg B \wedge C) \end{array}$$

Il problema può essere risolto determinando se una delle due formule seguenti è una conseguenza logica di $\{F_1, F_2\}$:

$$\begin{array}{ll} (SI) & (C \to (A \wedge B) \vee (\neg A \wedge \neg B)) \wedge (\neg C \to (A \wedge \neg B) \vee (\neg A \wedge B)) \\ (NO) & (C \to (A \wedge \neg B) \vee (\neg A \wedge B)) \wedge (\neg C \to (A \wedge B) \vee (\neg A \wedge \neg B)) \end{array}$$

(9) Un letterale, una congiunzione di letterali, una disgiunzione di letterali sono tutte formule in CNF e anche in DFN.

(10b) Utilizziamo il seguente linguaggio: P (piove), Q (prendo l'ombrello), C (resto a casa), R (sono al coperto), B (mi bagno). Il ragionamento è corretto se e solo se:

$$P \to Q \vee C,\ C \to R,\ Q \to R,\ \neg B \to R\ \models \neg B$$

Cerchiamo di costruire un'interpretazione $\mathcal{M}$ che renda vere tutte le premesse e falso $\neg B$. Si deve quindi avere $\mathcal{M}(B) = T$. Perché sia vera $P \to Q \vee C$, è sufficiente avere $\mathcal{M} \not\models P$, quindi $\mathcal{M}(P) = F$. Le altre tre premesse sono vere se $\mathcal{M}(R) = T$. Quindi qualsiasi interpretazione in cui sono veri B e R e falso P è un modello delle premesse e un contromodello della conclusione: il ragionamento non è corretto.

(10f) Utilizziamo il linguaggio seguente: C (l'investimento di capitali rimane costante), S (cresceranno le spese del governo), D (ci sarà disoccupazione), R (le tasse potranno essere ridotte). Il ragionamento è corretto se e solo se:

$$C \to S \vee D,\ \neg S \to R,\ R \wedge C \to \neg D\ \models S$$

Cerchiamo di costruire un'interpretazione $\mathcal{M}$ che renda vere le premesse e falso S. Allora deve essere $\mathcal{M} \not\models S$. In tal caso, poiché $\mathcal{M} \models \neg S \to R$, deve essere anche $\mathcal{M} \models R$. Se inoltre $\mathcal{M} \not\models C$, si ha che $\mathcal{M} \models C \to S \vee D$ e $\mathcal{M} \models R \wedge C \to \neg D$, poiché gli antecedenti di entrambe le implicazioni sono falsi. Quindi, se $\mathcal{M}(R) = T$ e $\mathcal{M}(C) = \mathcal{M}(S) = F$, allora $\mathcal{M}$ è un modello di tutte le premesse e un contromodello della conclusione: il ragionamento non è corretto.

Capitolo 3

La Logica dei Predicati

3.1 Introduzione

Si considerino i due ragionamenti seguenti:

$$\frac{\text{Tutti i triangoli sono poligoni;} \quad \overset{\triangle}{ABC} \text{ è un triangolo}}{\overset{\triangle}{ABC} \text{ è un poligono}}$$

$$\frac{\text{Tutti i pinguini sono mammiferi;} \quad \text{Alberto è un pinguino}}{\text{Alberto è un mammifero}}$$

Essi hanno la stessa forma:

$$\frac{\text{Tutti gli } A \text{ sono } B; \quad \text{l'oggetto } c \text{ è un } A}{\text{L'oggetto } c \text{ è un } B}$$

Qui A e B non denotano proposizioni ma “proprietà” o “predicati”. La correttezza di questa forma di ragionamento non può essere riconosciuta rappresentandola in logica proposizionale, dove i tre enunciati “tutti gli A sono B”, “l'oggetto c è un A” e “l'oggetto c è un B” non possono essere rappresentati che da tre lettere proposizionali distinte.

Per rappresentare adeguatamente la forma del ragionamento, occorre un'analisi più fine degli enunciati, ricorrendo alla *logica dei predicati del primo ordine* (o semplicemente *logica dei predicati* o *logica del primo ordine*), dove le proposizioni non sono “atomi”, ma sono ulteriormente analizzate in termini di oggetti, proprietà e relazioni.

Ad esempio, l'enunciato “uno più due è uguale a tre”, che in logica proposizionale sarebbe rappresentabile soltanto con una lettera proposizionale, viene analizzato in termini degli oggetti cui si riferisce (uno, due, tre), una funzione (più) e una relazione (essere uguale a). Per rappresentare gli oggetti si utilizzano *costanti*, per esempio 1, 2 e 3. La funzione è rappresentata da un *simbolo funzionale* (+) e la relazione mediante un *simbolo di predicato* (=). Se utilizziamo la notazione ordinaria infissa per + e =, l'enunciato è rappresentato da $1 + 2 = 3$. In generale, però in logica dei predicati si utilizza la notazione prefissa, per cui l'enunciato sarebbe rappresentato da $= (+(1, 2), 3)$.

Per rappresentare i ragionamenti visti sopra, possiamo utilizzare due simboli di predicato a un posto (o *unari*[1]), A e B, e una costante c. Avremo inoltre bisogno di alcuni *simboli logici*: una *variabile* x, il *quantificatore universale* $\forall$ e l'implicazione. Il ragionamento può essere rappresentato in simboli così:

[1]Unario = a un posto; binario = a due posti; ternario = a tre posti,...

$$\frac{\forall x(A(x) \rightarrow B(x)) \qquad A(c)}{B(c)}$$

$A(x)$, $B(x)$, $A(c)$ e $B(c)$ sono *atomi*, il cui significato è, rispettivamente, "la proprietà A vale per x", "la proprietà B vale per x", "la proprietà A vale per c", "la proprietà B vale per c". Un atomo della logica dei predicati è dunque un enunciato che asserisce l'esistenza di una proprietà o relazione tra oggetti. La formula $\forall x(A(x) \rightarrow B(x))$ ha un significato *universale*: per ogni oggetto x, se x ha la proprietà A, allora x ha la proprietà B. Qui, la variabile x è utilizzata come segnaposto per gli oggetti su cui "quantifica" il simbolo $\forall$.

3.2 Sintassi

3.2.1 Termini e formule

Un linguaggio del primo ordine è costruito sulla base di un alfabeto costituito da

1. *Simboli logici*:
   (a) un insieme infinito di variabili individuali: $x, y, z, x_1, x_2, ...$;
   (b) i connettivi proposizionali e le costanti $\top, \bot$;
   (c) il quantificatore universale $\forall$ e il quantificatore esistenziale $\exists$;
   (d) simboli separatori (parentesi e virgola).
2. *Simboli non logici*:
   (a) un insieme non vuoto di simboli di predicato, con associata "arità" (cioè il numero di argomenti a cui si applicano): $\{p^n, q^m, r^k, ...\}$;
   (b) un insieme (eventualmente vuoto) di costanti individuali: $a, b, c, a_1, a_2, ...$;
   (c) un insieme (eventualmente vuoto) di simboli funzionali, con associata "arità": $f_1^{n_1}, f_2^{n_2}, ...$

Se $=$ è tra i simboli di predicato di un linguaggio $\mathcal{L}$, allora $\mathcal{L}$ si dice un *linguaggio con uguaglianza*. L'uguaglianza è considerata un simbolo logico, in quanto il suo significato è sempre lo stesso, come quello dei connettivi o dei quantificatori.

Le espressioni ben formate del linguaggio sono di due tipi: i termini, che denotano oggetti, e le formule, che denotano enunciati.

Definizione 3.2.1 *L'insieme dei termini è definito induttivamente come segue:*

1. *Ogni variabile è un termine.*
2. *Ogni costante è un termine.*
3. *Se $t_1, .., t_n$ sono termini e f^n è un simbolo funzionale, allora $f(t_1, ..., t_n)$ è un termine.*
4. *Nient'altro è un termine.*

Un termine si dice chiuso *(o* ground*) se non contiene variabili.*

Definizione 3.2.2 *L'insieme delle formule è definito induttivamente come segue:*

1. *Se p è un simbolo di predicato a n argomenti, e $t_1, ..., t_n$ sono termini, allora $p(t_1, ..., t_n)$ è una formula (formula atomica).*

2. *$\top$ e $\bot$ sono formule (atomiche).*

3. *Se A è una formula, allora anche $\neg A$ è una formula.*

4. *Se A e B sono formule, allora anche $(A \wedge B)$, $(A \vee B)$, $(A \rightarrow B)$, $(A \equiv B)$ sono formule.*

5. *Se A è una formula e x una variabile, allora $\forall x A$ e $\exists x A$ sono formule.*

6. *Nient'altro è una formula.*

Consideriamo ad esempio un linguaggio del primo ordine contenente le costanti individuali *caino*, *abele*, *giovanni*, *robin_hood* e i simboli di predicato $fratello^2$, $assassino^1$, $padre^2$. Le espressioni seguenti sono formule di tale linguaggio:

$$fratello(caino, abele)$$
$$\neg fratello(giovanni, robin\_hood)$$
$$fratello(x, abele) \rightarrow assassino(x)$$

$$\forall x(fratello(x, abele) \rightarrow assassino(x))$$
$$\exists x(fratello(x, abele) \wedge assassino(x))$$

$$\forall x(fratello(x, y) \rightarrow fratello(y, x))$$
$$\exists x(padre(x, abele) \wedge padre(x, caino))$$
$$\forall x \forall y(fratello(x, y) \equiv \exists z(padre(z, x) \wedge padre(z, y)))$$

Ciascuna di queste formule può essere costruita passo passo applicando la definizione di termine e formula. Ad esempio, si può mostrare che l'espressione $fratello(x, abele) \rightarrow assassino(x)$ è una formula (del linguaggio dato) come segue:

1.	Poiché x è una variabile, x è un termine	(definizione di termine, clausola 1)
2.	Poiché *abele* è una costante, *abele* è un termine	(definizione di termine, clausola 2)
3.	Poiché *fratello* è un simbolo di predicato binario, per 1 e 2, $fratello(x, abele)$ è una formula	(definizione di formula, clausola 1)
4.	Poiché *assassino* è un simbolo di predicato unario, per 1, $assassino(x)$ è una formula	(definizione di formula, clausola 1)
5.	Per 3 e 4, $fratello(x, abele) \rightarrow assassino(x)$ è una formula	(definizione di formula, clausola 4)

Essendo le nozioni di termine e formula definite induttivamente, è giustificato l'uso di corrispondenti principi di induzione:

Principio di induzione sui termini

Se P è una proprietà dei termini di un linguaggio tale che:

(B) P vale per ogni variabile e per ogni costante, e

(PI) per ogni simbolo funzionale f^n e per tutti i termini $t_1, ..., t_n$: se P vale per t_1, e ... , e P vale per t_n, allora P vale anche per $f(t_1, ..., t_n)$;

allora P vale per ogni termine.

Principio di induzione sulle formule

Se P è una proprietà delle formule tale che:

(B) P vale per ogni formula atomica, e

(PI) per tutte le formule A e B:

- se P vale per A allora P vale per $\neg A$;
- se P vale per A e B, allora P vale per $(A \wedge B)$, per $(A \vee B)$, per $(A \rightarrow B)$, per $(A \equiv B)$;
- per ogni variabile x, se P vale per A allora P vale per $\forall x A$ e $\exists x A$;

allora P vale per ogni formula.

La nozione di sottoformula si estende al caso della logica dei predicati come segue:

1. Se A è atomica allora $subf(A) = \{A\}$.
2. $subf(\neg A) = subf(A) \cup \{\neg A\}$.
3. $subf(A \wedge B) = subf(A) \cup subf(B) \cup \{A \wedge B\}$.
4. $subf(A \vee B) = subf(A) \cup subf(B) \cup \{A \vee B\}$.
5. $subf(A \rightarrow B) = subf(A) \cup subf(B) \cup \{A \rightarrow B\}$.
6. $subf(A \equiv B) = subf(A) \cup subf(B) \cup \{A \equiv B\}$.
7. $subf(\forall x A) = subf(A) \cup \{\forall x A\}$.
8. $subf(\exists x A) = subf(A) \cup \{\exists x A\}$.

3.2.2 Variabili libere e vincolate

Lo *scopo* o *campo d'azione* di un quantificatore in una formula è definito come segue: il campo d'azione del quantificatore universale $\forall x$ nella formula $\forall x A$ è A. Analogamente, il campo d'azione del quantificatore $\exists x$ nella formula $\exists x A$ è A.

Per esempio:

$$p(c) \wedge \forall x \overbrace{(q(x,c) \rightarrow \exists y \underbrace{(p(y) \vee r(x,y,z))}_{scopo\ di\ \exists y})}^{scopo\ di\ \forall x} \equiv \exists x \overbrace{\forall y \underbrace{q(x,y)}_{scopo\ di\ \forall y}}^{scopo\ di\ \exists x}$$

Una occorrenza di una variabile x in una formula si dice *vincolata* se tale occorrenza occorre nello scopo di un quantificatore $\forall x$ o $\exists x$ (che quantifica su x), oppure è la variabile stessa di un quantificatore (cioè è la x di un $\forall x$ o $\exists x$).

Una occorrenza *libera* di x è un'occorrenza non vincolata di x.

Ad esempio, nella formula seguente sono sottolineate (tutte e solo) le occorrenze libere di x e y:

$$p(\underline{x}) \wedge \forall x(q(x,\underline{y}) \vee \neg q(c,x))$$

Una variabile x si dice libera in una formula A se x ha almeno un'occorrenza libera in A. La variabile x si dice vincolata in A se x ha almeno un'occorrenza vincolata in A. Si noti dunque che una variabile può essere sia libera che vincolata in una formula A. Ne è un esempio la variabile x nella formula scritta sopra.

Una formula si dice *chiusa* se non contiene variabili libere.

3.3 Semantica

Come già accennato, la semantica (il significato) di un termine è un oggetto, e, come nella logica proposizionale, la semantica di una formula chiusa è un valore booleano. Come nella logica proposizionale, inoltre, per stabilire la semantica di un termine o di una formula, occorre definire una interpretazione del linguaggio.

Consideriamo inizialmente un semplice esempio. Sia A la formula chiusa

$$\exists x\, p(x) \vee \exists y\, q(y) {\rightarrow} \exists z\, (p(z) \wedge q(z))$$

Per dare un'interpretazione di A occorre innanzitutto stabilire un *dominio* dell'interpretazione, cioè l'insieme degli oggetti a cui si fa riferimento. Stabiliamo ad esempio che il dominio è costituito dall'insieme $D = \{1, 2\}$. In secondo luogo occorre stabilire una interpretazione dei simboli di predicato p e q. Stabiliamo ad esempio che $p(x)$ significa "x è pari" e $q(x)$ significa "x è maggiore o uguale a 1". A volte però non è facile caratterizzare *intensionalmente*[2] una proprietà o relazione, perciò in generale, l'interpretazione di un simbolo di predicato unario è definita *estensionalmente*, come un insieme (un sottoinsieme del dominio): l'insieme di tutti gli oggetti del dominio che godono della proprietà denotata da p (cioè l'estensione della proprietà). Nel nostro caso, l'interpretazione di p è l'insieme $\{2\}$ e quella di q è $\{1, 2\}$. Ora, intuitivamente, possiamo dire che:

1. $\exists x\, p(x)$ è vero se esiste $d \in D = \{1, 2\}$ tale che d appartiene all'interpretazione di p, cioè $d \in \{2\}$; quindi $\exists x\, p(x)$ è vero nell'interpretazione data.
2. $\exists x\, q(x)$ è vero se esiste $d \in \{1, 2\}$ tale che d appartiene all'interpretazione di q, cioè $d \in \{1, 2\}$; quindi $\exists x\, q(x)$ è vero.
3. $\exists z\, (p(z) \wedge q(z))$ è vero se esiste $d \in \{1, 2\}$ tale che d appartiene sia all'interpretazione di p che a quella di q, cioè $d \in \{2\}$ e $d \in \{1, 2\}$. Quindi anche $\exists z\, (p(z) \wedge q(z))$ è vero.
4. Di conseguenza, per la semantica di $\vee$ e $\rightarrow$, A è vera nell'interpretazione data.

Consideriamo ora una diversa interpretazione di A, con lo stesso dominio $D = \{1, 2\}$, ma dove $p(x)$ significa "x è pari" e $q(x)$ significa "x è dispari". Quindi l'interpretazione di p è $\{2\}$ e quella di q è $\{1\}$. In questa interpretazione:

1. $\exists x\, p(x)$ è vero se esiste $d \in \{1, 2\}$ tale che $d \in \{2\}$; quindi è vero.
2. $\exists x\, q(x)$ è vero se esiste $d \in \{1, 2\}$ tale che $d \in \{1\}$; quindi è vero.
3. $\exists z\, (p(z) \wedge q(z))$ è vero se esiste $d \in \{1, 2\}$ tale che $d \in \{2\}$ e $d \in \{1\}$; questo è falso, quindi $\exists z\, (p(z) \wedge q(z))$ è falsa.
4. Di conseguenza, per il significato di $\vee$ e $\rightarrow$, la formula A è falsa in questa interpretazione.

Se il linguaggio contiene anche simboli funzionali, occorrerà associare una funzione sul dominio (con la stessa "arità" del simbolo) a ciascuno di essi. L'interpretazione di un simbolo di predicato n-ario è una relazione a n argomenti sul dominio, quindi un insieme di n-uple di elementi del dominio.

La definizione formale di interpretazione di un linguaggio del primo ordine è la seguente:

[2] In termini semplici, l'*intensione* di un predicato è il modo in cui viene descritto, la sua *estensione* è ciò che denota, cioè l'insieme degli oggetti per cui il predicato è vero. Due predicati possono avere diverse intensioni ma la stessa estensione. Ad esempio il predicato "x è un numero primo appartenente all'insieme $\{2, 9, 15\}$" e il predicato "x è un numero pari appartenente all'insieme $\{2, 9, 15\}$" hanno diverse intensioni (sono caratterizzati in modo diverso, da proprietà diverse), ma la stessa estensione: l'insieme $\{2\}$.

Definizione 3.3.1 *Sia $\mathcal{L}$ un linguaggio, con costanti $\mathcal{C}$, simboli funzionali $\mathcal{F}$ e simboli di predicato $\mathcal{P}$. Una interpretazione $\mathcal{M}$ di $\mathcal{L}$ è costituita da:*

1. *Un insieme non vuoto D, chiamato dominio o universo dell'interpretazione.*

2. *Una funzione di interpretazione che associa:*

   (a) *a ogni $c \in \mathcal{C}$ un elemento $\mathcal{M}(c) \in D$;*

   (b) *a ogni $f^n \in \mathcal{F}$ una funzione $\mathcal{M}(f) : D^n \longrightarrow D$;*

   (c) *a ogni $p^n \in \mathcal{P}$ una relazione n-aria su D: $\mathcal{M}(p) \subseteq D^n$.*

Se $\mathcal{L}$ è un linguaggio con eguaglianza, il simbolo di predicato $=$ è trattato come simbolo logico e l'interpretazione di $=$ è sempre l'identità:

$$\mathcal{M}(=) \ = \ \{< d, d > \mid d \in D\}$$

Una interpretazione di una formula A è una interpretazione di qualsiasi linguaggio che contenga tutti i simboli non logici di A.

Ad esempio, sono interpretazioni di $\forall x\, p(a, x)$, e del linguaggio con $\mathcal{C} = \{a\}$, $\mathcal{F} = \emptyset$, $\mathcal{P} = \{p\}$:

1. $\mathcal{M}_1$ con dominio $\mathbb{N}$, $\mathcal{M}_1(a) = 0$, e $\mathcal{M}_1(p) = \{\langle n, m\rangle \in \mathbb{N}^2 \mid n \leq m\}$.

2. $\mathcal{M}_2$ con dominio $\mathbb{N}$, $\mathcal{M}_2(a) = 1$, e $\mathcal{M}_2(p) = \{\langle n, m\rangle \in \mathbb{N}^2 \mid n \leq m\}$.

3. $\mathcal{M}_3$ con dominio $\mathbb{Z}$, $\mathcal{M}_3(a) = 0$, e $\mathcal{M}_3(p) = \{\langle x, y\rangle \in \mathbb{Z}^2 \mid x \leq y\}$.

4. $\mathcal{M}_4$ con dominio S uguale all'insieme di tutte le stringhe di caratteri alfanumerici, $\mathcal{M}_4(a) =$ ‘ ‘ (la stringa vuota), e $\mathcal{M}_4(p) = \{\langle s_1, s_2\rangle \in S^2 \mid s_1$ è una sottostringa di $s_2\}$.

5. $\mathcal{M}_5$ con dominio $D = \{0, 1, 2\}$, $\mathcal{M}_5(a) = 1$ e $\mathcal{M}_5(p) = \{\langle 0, 1\rangle, \langle 1, 0\rangle, \langle 1, 2\rangle\}$.

3.3.1 Interpretazione dei termini

Sia $\mathcal{L}$ un linguaggio del primo ordine e $\mathcal{M}$ una sua interpretazione. $\mathcal{M}$ definisce già l'interpretazione di tutte le costanti c del linguaggio, $\mathcal{M}(c)$. È semplice estendere tale interpretazione a tutti i termini *chiusi* t, indicandola sempre con $\mathcal{M}(t)$, ricorsivamente:

1. se c è una costante in $\mathcal{L}$, l'interpretazione di c è $\mathcal{M}(c)$;

2. se $f(t_1, ..., t_n)$ è un termine chiuso, la sua interpretazione è

$$\mathcal{M}(f(t_1, ..., t_n)) = \mathcal{M}(f)(\mathcal{M}(t_1), ..., \mathcal{M}(t_n))$$

Consideriamo ad esempio il linguaggio con costanti $\mathcal{C} = \{zero\}$ e simboli funzionali $\mathcal{F} = \{succ, sum, times\}$. Sia $\mathcal{M}$ è un'interpretazione con dominio $\mathbb{N}$, tale che $\mathcal{M}(zero) = 0$, $\mathcal{M}(succ)$ è l'operazione successore $\lambda x.x{+}1$ (che associa a ogni numero n il suo successore $n+1$), $\mathcal{M}(sum)$ è la somma e $\mathcal{M}(times)$ è il prodotto sui naturali. Possiamo allora determinare l'interpretazione del termine chiuso $times(succ(succ(zero)), sum(succ(zero), zero))$ in $\mathcal{M}$ come segue:

$$
\begin{aligned}
&\mathcal{M}(times(succ(succ(zero)), sum(succ(zero), zero))) \\
&\quad = \mathcal{M}(times)(\mathcal{M}(succ(succ(zero))), \mathcal{M}(sum(succ(zero), zero))) \\
&\quad = \mathcal{M}(succ(succ(zero)) \times \mathcal{M}(sum(succ(zero), zero)) \\
&\quad = \mathcal{M}(succ)(\mathcal{M}(succ(zero))) \times \mathcal{M}(sum)(\mathcal{M}(succ(zero)), \mathcal{M}(zero)) \\
&\quad = (\mathcal{M}(succ(zero)) + 1) \times (\mathcal{M}(succ(zero)) + \mathcal{M}(zero)) \\
&\quad = (\mathcal{M}(succ)(\mathcal{M}(zero)) + 1) \times (\mathcal{M}(succ)(\mathcal{M}(zero)) + 0) \\
&\quad = ((\mathcal{M}(zero) + 1) + 1) \times ((\mathcal{M}(zero) + 1) + 0) \\
&\quad = ((0 + 1) + 1) \times ((0 + 1) + 0) \\
&\quad = 2 \times 1 = 2
\end{aligned}
$$

Per definire l'interpretazione di termini che contengono variabili, è necessario determinare come vengono interpretate tali variabili. In generale, una variabile individuale denoterà un oggetto del dominio, ma quale oggetto sia non è definito dall'interpretazione $\mathcal{M}$ del linguaggio (che non contiene infatti un'interpretazione delle variabili). Accanto a $\mathcal{M}$, allora, dobbiamo avere una specifica interpretazione delle variabili, per assegnare un significato ai termini che contengono variabili.

Definizione 3.3.2 *Se $\mathcal{M}$ è un'interpretazione con dominio D, una interpretazione delle variabili su $\mathcal{M}$ (o* assegnazione*) è una funzione s che associa un oggetto di D a ogni variabile x. Cioè, se X è l'insieme delle variabili del linguaggio:*

$$s : X \to D$$

Una volta fissata un'interpretazione $\mathcal{M}$ e un'assegnazione s, è possibile determinare il significato di un termine qualsiasi (in $\mathcal{M}$, secondo s). Ad esempio, sia f un simbolo funzionale unario e sia $\mathcal{M}$ l'interpretazione il cui dominio è $\mathbb{N}$ e $\mathcal{M}(f)$ è la funzione "doppio" $\lambda n.2 \times n$ (che associa ad ogni n il valore $2 \times n$). Se s è una qualsiasi interpretazione delle variabili tale che $s(x) = 3$, allora l'interpretazione del termine $f(x)$ è 2×3, cioè 6.

L'estensione di un'assegnazione all'insieme di tutti i termini del linguaggio è definita come segue.

Definizione 3.3.3 *Sia $\mathcal{M}$ un'interpretazione e $s : X \to D$ un'assegnazione su $\mathcal{M}$. Allora $\overline{s}$ è l'estensione di s all'insieme di tutti i termini del linguaggio, definita induttivamente come segue:*

1. *se $x \in X$ è una variabile, allora $\overline{s}(x) = s(x)$;*
2. *se c è una costante, allora $\overline{s}(c) = \mathcal{M}(c)$;*
3. *se $f(t_1, ..., t_n)$ è un termine, allora $\overline{s}(t) = \mathcal{M}(f)(\overline{s}(t_1), ..., \overline{s}(t_n))$.*

In seguito, dato che non c'è rischio di confusione, scriveremo spesso $s(t)$ al posto di $\overline{s}(t)$.

3.3.2 Interpretazione delle formule

Data una interpretazione $\mathcal{M}$, una formula che contiene n variabili libere può essere soddisfatta da (vera per) alcuni valori del dominio, falsa per altri. Essa identifica dunque una relazione n-aria sul dominio di $\mathcal{M}$: quella costituita dalle tuple di elementi che la soddisfano. Ad esempio, sia f un simbolo funzionale unario e p un simbolo di predicato binario e sia $\mathcal{M}$ l'interpretazione il cui dominio è $\mathbb{N}$, $\mathcal{M}(f)$ è la funzione "doppio" $\lambda n.2 \times n$, e $\mathcal{M}(p) = \{\langle n, m\rangle \mid n^2 \leq m\}$. La formula $p(x, f(y))$ identifica una relazione binaria su $\mathbb{N}$:

$$\{\langle m, n\rangle \mid n, m \in \mathbb{N} \text{ e } n^2 \leq 2 \times m\}$$

L'interpretazione di una formula contenente variabili libere (o "aperta") in una interpretazione $\mathcal{M}$ è dunque una relazione e non un valore di verità.

Se però, accanto all'interpretazione delle costanti, dei simboli di funzione e di predicato, fissiamo anche una interpretazione delle variabili, allora possiamo associare a qualsiasi formula un valore di verità. Consideriamo ad esempio l'interpretazione $\mathcal{M}$ sopra definita e la formula $p(x, f(y))$. Sia s una qualsiasi interpretazione delle variabili tale che $s(x) = 3$ e $s(y) = 5$. Intuitivamente, possiamo dire che la formula $p(x, f(y))$ è vera in $\mathcal{M}$ secondo l'assegnazione s: infatti la coppia costituita dall'interpretazione di x secondo s e l'interpretazione di $f(y)$ secondo s, $\langle s(x), s(f(y))\rangle = \langle 3, 2 \times 5\rangle$ appartiene all'interpretazione di p, perché $3^2 \leq 10$. Se invece s' è una assegnazione con $s'(x) = 4$ e $s'(y) = 5$, $p(x, f(y))$ è falsa in $\mathcal{M}$ secondo l'assegnazione s', perché $4^2 \leq 2 \times 5$ è falso.

Quella che occorre definire è allora una relazione tra interpretazioni, assegnazioni e formule: "l'assegnazione s soddisfa la formula A nell'interpretazione $\mathcal{M}$". Tale relazione viene scritta $(\mathcal{M}, s) \models A$.

Nel definire tale relazione occorre utilizzare la nozione di x-variante di un'assegnazione s, cioè di un'assegnazione che è in tutto uguale a s, tranne che, eventualmente, per il valore che essa assegna a x:

Definizione 3.3.4 *Se s e s' sono assegnazioni su $\mathcal{M}$ e x è una variabile, s' si dice una x-variante di s sse per ogni variabile y diversa da x: $s'(y) = s(y)$.*

Si noti che, in particolare, ogni assegnazione è una x-variante di se stessa. La x-variante di s e che assegna l'oggetto d alla variabile x sarà denotata da $s[d/x]$.

Possiamo ora definire la relazione di "soddisfacimento" $(\mathcal{M}, s) \models A$.

Definizione 3.3.5 *Sia A una formula, $\mathcal{M}$ un'interpretazione di A e $s : X \to D$ un'assegnazione su $\mathcal{M}$. Allora la relazione $(\mathcal{M}, s) \models A$ (s soddisfa A in $\mathcal{M}$) è definita induttivamente come segue:*

1. $(\mathcal{M}, s) \models p(t_1, ..., t_n)$ *sse* $\langle \overline{s}(t_1), ..., \overline{s}(t_n)\rangle \in \mathcal{M}(p)$
2. $(\mathcal{M}, s) \models \top$ *e* $(\mathcal{M}, s) \not\models \bot$*;*
3. $(\mathcal{M}, s) \models \neg A$ *sse* $(\mathcal{M}, s) \not\models A$*;*
4. $(\mathcal{M}, s) \models A \wedge B$ *sse* $(\mathcal{M}, s) \models A$ *e* $(\mathcal{M}, s) \models B$*;*
5. $(\mathcal{M}, s) \models A \vee B$ *sse* $(\mathcal{M}, s) \models A$ *oppure* $(\mathcal{M}, s) \models B$*;*
6. $(\mathcal{M}, s) \models A \to B$ *sse* $(\mathcal{M}, s) \not\models A$ *oppure* $(\mathcal{M}, s) \models B$*;*
7. $(\mathcal{M}, s) \models A \equiv B$ *sse:*
   $(\mathcal{M}, s) \models A$ *e* $(\mathcal{M}, s) \models B$, *oppure* $(\mathcal{M}, s) \not\models A$ *e* $(\mathcal{M}, s) \not\models B$
8. $(\mathcal{M}, s) \models \forall x A$ *sse per ogni* $d \in D$: $(\mathcal{M}, s[d/x]) \models A$
9. $(\mathcal{M}, s) \models \exists x A$ *sse esite* $d \in D$, *tale che:* $(\mathcal{M}, s[d/x]) \models A$

Si noti che il simbolo di uguaglianza è sempre interpretato con la relazione di identità, quindi, se t e t' sono termini,

$$(\mathcal{M}, s) \models t = t' \text{ sse } \overline{s}(t) = \overline{s}(t')$$

Può essere utile avere sotto mano anche la caratterizzazione induttiva di "non soddisfacimento" o di "contromodello" (diciamo che la coppia $(\mathcal{M}, s)$ è un *contromodello* di A se $(\mathcal{M}, s) \not\models A$), che ovviamente deriva dalla definizione 3.3.5:

1. $(\mathcal{M}, s) \not\models p(t_1, ..., t_n)$ sse $\langle \overline{s}(t_1), ..., \overline{s}(t_n)\rangle \notin \mathcal{M}(p)$

2. $(\mathcal{M}, s) \not\models \top$ è sempre falso e $(\mathcal{M}, s) \not\models \bot$ è sempre vero;
3. $(\mathcal{M}, s) \not\models \neg A$ sse $(\mathcal{M}, s) \models A$;
4. $(\mathcal{M}, s) \not\models A \wedge B$ sse $(\mathcal{M}, s) \not\models A$ oppure $(\mathcal{M}, s) \not\models B$;
5. $(\mathcal{M}, s) \not\models A \vee B$ sse $(\mathcal{M}, s) \not\models A$ e $(\mathcal{M}, s) \not\models B$;
6. $(\mathcal{M}, s) \not\models A \rightarrow B$ sse $(\mathcal{M}, s) \models A$ e $(\mathcal{M}, s) \not\models B$;
7. $(\mathcal{M}, s) \not\models A \equiv B$:
   sse $(\mathcal{M}, s) \not\models A$ e $(\mathcal{M}, s) \models B$, oppure $(\mathcal{M}, s) \models A$ e $(\mathcal{M}, s) \not\models B$
8. $(\mathcal{M}, s) \not\models \forall x A$ sse esiste $d \in D$, tale che: $(\mathcal{M}, s[d/x]) \not\models A$
9. $(\mathcal{M}, s) \not\models \exists x A$ sse per ogni $d \in D$: $(\mathcal{M}, s[d/x]) \not\models A$

La nozione di *verità* di una formula in una interpretazione stabilisce che il significato delle variabili libere è "universale": una formula è vera in $\mathcal{M}$ se qualsiasi assegnazione soddisfa A in $\mathcal{M}$.

Definizione 3.3.6 *Se A è una formula e $\mathcal{M}$ un'interpretazione di A, allora A è vera in $\mathcal{M}$ e $\mathcal{M}$ è un modello di A ($\mathcal{M} \models A$) sse per ogni assegnazione s su $\mathcal{M}$:* $(\mathcal{M}, s) \models A$.

La falsità di una formula in una interpretazione non è la stessa cosa della "non verità":

Definizione 3.3.7 *Una formula A è falsa in una interpretazione $\mathcal{M}$ sse nessuna assegnazione la soddisfa, cioè se per ogni assegnazione s su $\mathcal{M}$:* $(\mathcal{M}, s) \not\models A$.

Dunque non è sufficiente che $\mathcal{M} \not\models A$ (cioè che esista qualche assegnazione che non soddisfa A) per stabilire che A è falsa in $\mathcal{M}$. Infatti, può accadere che una formula non sia né vera né falsa in una interpretazione: ciò avviene quando esiste almeno un'assegnazione che la soddisfa e una che non la soddisfa.

Le relazioni di soddisfacimento, verità e falsità hanno le seguenti proprietà:

1. A è falsa in una interpretazione $\mathcal{M}$ se e solo se $\neg A$ è vera in $\mathcal{M}$; e A è vera in $\mathcal{M}$ se e solo se $\neg A$ è falsa in $\mathcal{M}$.
2. Nessuna formula può essere contemporaneamente vera e falsa in una interpretazione.
3. Se A e $A \rightarrow B$ sono vere in $\mathcal{M}$, allora B è vera in $\mathcal{M}$.
4. $A \rightarrow B$ è falsa in $\mathcal{M}$ se e solo se A è vera e B è falsa.
5. A è vera in $\mathcal{M}$ sse $\forall x A$ è vera in $\mathcal{M}$. Quindi A è vera in $\mathcal{M}$ sse la sua *chiusura universale* è vera in $\mathcal{M}$:

   $$\mathcal{M} \models A \text{ sse } \mathcal{M} \models \forall x_1 ... \forall x_n A$$

   dove $x_1, ..., x_n$ sono tutte le variabili libere in A. Nel seguito indicheremo con $\forall A$ la chiusura universale della formula A.
6. Se s e s' sono assegnazioni su $\mathcal{M}$ tali tali che $s(x) = s'(x)$ per ogni variabile x che occorre libera in A (cioè s e s' coincidono su tutte le variabili che occorrono libere in A), allora $(\mathcal{M}, s) \models A$ sse $(\mathcal{M}, s') \models A$ (vedi Lemma 3.3.9). In altri termini, per determinare se $(\mathcal{M}, s) \models A$ è sufficiente conoscere il valore assegnato da s alle variabili che occorrono libere in A.

7. Se A è chiusa, allora per ogni data interpretazione $\mathcal{M}$ o A è vera oppure $\neg A$ è vera (cioè A è falsa). Infatti, se A non contiene variabili libere, come conseguenza di 6, si ha che se s e s' sono assegnazioni qualsiasi su $\mathcal{M}$, allora $(\mathcal{M}, s) \models A$ sse $(\mathcal{M}, s') \models A$. Cioè o tutte le assegnazioni soddisfano A, oppure nessuna la soddisfa (la soddisfacibilità di A è indipendente dall'assegnazione).

   Come conseguenza di questo fatto, si ha che, se A è chiusa, $\mathcal{M} \models A$ sse esiste un'assegnazione s tale che $(\mathcal{M}, s) \models A$.

La definizione 3.3.5 (e la definizione equivalente di $\not\models$) può essere usata per dimostrare che una formula è vera (o falsa) in una interpretazione. Consideriamo come esempio la formula $A = \forall x\, p(a, x)$ e le interpretazioni di pagina 35.

1. $\mathcal{M}_1 \models A$ sse per qualsiasi s:
   $(\mathcal{M}_1, s) \models A$
   sse per ogni $n \in \mathbb{N}$, $(\mathcal{M}_1, s[n/x]) \models p(a, x)$
   sse per ogni $n \in \mathbb{N}$, $\langle s[n/x](a), s[n/x](x)\rangle \in \mathcal{M}_1(p)$
   sse per ogni $n \in \mathbb{N}$, $\langle 0, n\rangle \in \mathcal{M}_1(p)$
   sse per ogni $n \in \mathbb{N}$, $0 \leq n$: vero

2. $\mathcal{M}_2 \models A$ sse per qualsiasi s:
   $(\mathcal{M}_2, s) \models A$
   sse per ogni $n \in \mathbb{N}$, $(\mathcal{M}_2, s[n/x]) \models p(a, x)$
   sse per ogni $n \in \mathbb{N}$, $\langle s[n/x](a), s[n/x](x)\rangle \in \mathcal{M}_2(p)$
   sse per ogni $n \in \mathbb{N}$, $\langle 1, n\rangle \in \mathcal{M}_2(p)$
   sse per ogni $n \in \mathbb{N}$, $1 \leq n$: falso

3. $\mathcal{M}_3 \models A$ sse per qualsiasi s:
   $(\mathcal{M}_3, s) \models A$
   sse per ogni $n \in \mathbb{Z}$, $(\mathcal{M}_3, s[n/x]) \models p(a, x)$
   sse per ogni $n \in \mathbb{N}$, $\langle s[n/x](a), s[n/x](x)\rangle \in \mathcal{M}_3(p)$
   sse per ogni $n \in \mathbb{Z}$, $\langle 0, n\rangle \in \mathcal{M}_3(p)$
   sse per ogni $n \in \mathbb{Z}$, $0 \leq n$: falso

4. $\mathcal{M}_4 \models A$ sse per qualsiasi s:
   $(\mathcal{M}_4, s) \models A$
   sse per ogni $d \in \mathcal{S}$, $(\mathcal{M}_4, s[d/x]) \models p(a, x)$
   sse per ogni $d \in \mathcal{S}$, $\langle s[d/x](a), s[d/x](x)\rangle \in \mathcal{M}_4(p)$
   sse per ogni $d \in \mathcal{S}$, $\langle \text{‘ ‘}, d\rangle \in \mathcal{M}_4(p)$
   sse per ogni $d \in \mathcal{S}$, ‘ ‘ è una sottostringa di d: vero

5. $\mathcal{M}_5 \models A$ sse per qualsiasi s:
   $(\mathcal{M}_5, s) \models A$
   sse per ogni $d \in D$, $(\mathcal{M}_5, s[d/x]) \models p(a, x)$
   sse per ogni $d \in D$, $\langle s[d/x](a), s[d/x](x)\rangle \in \mathcal{M}_5(p)$
   sse per ogni $d \in D$, $\langle 1, d\rangle \in \mathcal{M}_5(p)$
   falso perché $\langle 1, 1\rangle \notin \mathcal{M}_5(p)$.

Consideriamo ora il linguaggio con:

$$\mathcal{C} = \{felix,\ silvestro,\ giovanni,\ riccardo\}$$
$$\mathcal{F} = \{padre^1\}$$
$$\mathcal{P} = \{gatto^1,\ fratello^2, =\}$$

Sia $\mathcal{M}$ l'interpretazione così definita:

- $D = \{0, 1, 2, 3\}$

- $\mathcal{M}(felix) = 0$, $\mathcal{M}(silvestro) = 1$, $\mathcal{M}(giovanni) = 2$, $\mathcal{M}(riccardo) = 2$
- $\mathcal{M}(padre) = \{< 0, 0 >, < 1, 0 >, < 2, 3 >, < 3, 2 >\}$.
  In altri termini:

  $$\mathcal{M}(padre) = F \text{ tale che } \begin{array}{l} F(0) = 0, \\ F(1) = 0, \\ F(2) = 3, \\ F(3) = 2 \end{array}$$

- $\mathcal{M}(gatto) = \{0, 1\}$, $\mathcal{M}(fratello) = \{< 0, 1 >, < 1, 0 >\}$

Consideriamo la formula $A = gatto(padre(padre(silvestro)))$ e determiniamo la sua interpretazione in $\mathcal{M}$:

$$\begin{array}{ll} \multicolumn{2}{l}{\mathcal{M} \models A \text{ sse per qualsiasi assegnazione } s\text{:}} \\ & (\mathcal{M}, s) \models A \\ \text{sse} & s(padre(padre(silvestro))) \in \mathcal{M}(gatto) \\ \text{sse} & F(s(padre(silvestro))) \in \{0, 1\} \\ \text{sse} & F(F(s(silvestro))) \in \{0, 1\} \\ \text{sse} & F(F(1)) \in \{0, 1\} \\ \text{sse} & F(0) \in \{0, 1\} \\ \text{sse} & 0 \in \{0, 1\}\text{: vero} \end{array}$$

Determiniamo ora l'interpretazione in $\mathcal{M}$ della formula:

$$A = \exists x(padre(silvestro) = x \wedge \exists y(padre(x) = y \wedge gatto(y)))$$

$$\begin{array}{ll} \multicolumn{2}{l}{\mathcal{M} \models \exists x(padre(silvestro) = x \wedge \exists y(padre(x) = y \wedge gatto(y))) \text{ sse per ogni } s\text{:}} \\ & (\mathcal{M}, s) \models A \\ \text{sse} & \text{esiste } d \in D \text{ tale che} \\ & (\mathcal{M}, s[d/x]) \models padre(silvestro) = x \wedge \exists y(padre(x) = y \wedge gatto(y)) \\ \text{sse} & \text{esiste } d \in D \text{ tale che } (\mathcal{M}, s[d/x]) \models padre(silvestro) = x \text{ e} \\ & (\mathcal{M}, s[d/x]) \models \exists y(padre(x) = y \wedge gatto(y)) \\ \text{sse} & \text{esiste } d \in D \text{ tale che } < s[d/x](padre(silvestro)), s[d/x](x) > \in \mathcal{M}(=) \text{ e} \\ & \text{esiste } d' \in D \text{ tale che } (\mathcal{M}, s[d/x][d'/y]) \models padre(x) = y \wedge gatto(y) \\ \text{sse} & \text{esiste } d \in D \text{ tale che } F(1) = s[d/x](x) \text{ e} \\ & \text{esiste } d' \in D \text{ tale che } (\mathcal{M}, s[d/x][d'/y]) \models padre(x) = y \\ & \qquad \text{e } (\mathcal{M}, s[d/x][d'/y]) \models gatto(y) \\ \text{sse} & \text{esiste } d \in D \text{ tale che } 0 = d \text{ e} \\ & \text{esiste } d' \in D \text{ tale che } s[d/x][d'/y](padre(x)) = s[d/x][d'/y](y) \\ & \qquad \text{e } s[d/x][d'/y](y) \in \mathcal{M}(gatto) \\ \text{sse} & \text{VERO (esiste } d \in D \text{ tale che } 0 = d) \text{ e} \\ & \text{esiste } d' \in D \text{ tale che } F(0) = d' \text{ e } d' \in \mathcal{M}(gatto) \\ \text{sse} & \text{esiste } d' \in D \text{ tale che } 0 = d' \text{ e } d' \in \mathcal{M}(gatto) \\ \text{sse} & \text{VERO perché } 0 \in \{0, 1\}. \end{array}$$

Se si vogliono verificare i passaggi costruiti a mano per verificare la verità di una formula chiusa in un'interpretazione con dominio finito, si può utilizzare il programma OCaml `models`, che si può scaricare da [7].

3.3.3 Sostituzione di variabili con termini

Se A è una formula, x una variabile e t un termine, allora

$$A[t/x]$$

denota la formula che si ottiene da A sostituendo ogni occorrenza *libera* della variabile x con il termine t.

Una notazione simile si usa per la sostituzione *simultanea* di più variabili:

$$A[t_1/x_1, ..., t_n/x_n]$$

Per comprendere la differenza tra sostituzione simultanea e non, si consideri ad esempio che:

$$p(x, y)[f(y)/x, f(x)/y] = p(f(y), f(x))$$

mentre se si sostituisce prima x e poi y, o viceversa, si ottengono risultati diversi:

$$p(x, y)[f(y)/x][f(x)/y] = p(f(y), y)[f(x)/y] = p(f(f(x)), f(x))$$

e:

$$p(x, y)[f(x)/y][f(y)/x] = p(x, f(x))[f(y)/x] = p(f(y), f(f(y)))$$

3.3.4 Soddisfacibilità, validità, conseguenza logica

Sia A una formula. Le seguenti nozioni sono l'estensione alla logica dei predicati delle corrispondenti nozioni per la logica proposizionale:

1. A è *soddisfacibile* se esiste un'interpretazione $\mathcal{M}$ di A e un'assegnazione s su $\mathcal{M}$ tale che $(\mathcal{M}, s) \models A$.

2. A è *valida* ($\models A$) sse è vera in ogni sua interpretazione: cioè se per ogni interpretazione $\mathcal{M}$ di A e per ogni assegnazione s su $\mathcal{M}$: $(\mathcal{M}, s) \models A$.

   Quindi A è valida sse la sua chiusura universale è valida. E A è valida sse $\neg A$ non è soddisfacibile.

3. A è *contraddittoria* o è una *contraddizione* sse essa è falsa in ogni interpretazione: non esiste nessuna interpretazione $\mathcal{M}$ di A e nessuna assegnazione s su $\mathcal{M}$ tale che $(\mathcal{M}, s) \models A$.

4. A e B sono logicamente equivalenti ($A \leftrightarrow B$) sse per ogni interpretazione $\mathcal{M}$ e assegnazione s su $\mathcal{M}$: $(\mathcal{M}, s) \models A$ sse $(\mathcal{M}, s) \models B$.

5. Un'assegnazione s soddisfa un insieme S di formule nell'interpretazione $\mathcal{M}$ ($(\mathcal{M}, s) \models S$) se s soddisfa ogni formula di S in $\mathcal{M}$.
   $\mathcal{M}$ è un modello di un insieme di formule S ($\mathcal{M} \models S$) sse è un modello di ciascuna formula in S.

6. A è una *conseguenza logica* di S ($S \models A$) sse per ogni interpretazione $\mathcal{M}$ e assegnazione s su $\mathcal{M}$, se $(\mathcal{M}, s) \models C$ per ogni formula $C \in S$, allora anche $(\mathcal{M}, s) \models A$. Se le formule in $S \cup \{A\}$ sono tutte chiuse, allora ciò equivale a dire che ogni modello di S è un modello di A.

7. A implica logicamente B sse per ogni interpretazione $\mathcal{M}$ e assegnazione s su $\mathcal{M}$, se $(M, s) \models A$, allora $(M, s) \models B$.

Si noti, in particolare, che $S \models A$ è più forte della relazione "A è vera in tutte le interpretazioni in cui sono vere tutte le formule in S". Ad esempio, $\forall x\, p(x)$ è vera in tutte le interpretazioni in cui è vera $p(x)$, in quanto $p(x)$ è vera in $\mathcal{M}$ sse è vera la sua chiusura universale. Tuttavia $p(x) \not\models \forall x\, p(x)$: sia $\mathcal{M}$ l'interpretazione con dominio $D = \{0, 1\}$ e $\mathcal{M}(p) = \{0\}$, e sia s un'assegnazione tale che $s(x) = 0$. Allora $(\mathcal{M}, s) \models p(x)$, ma $(\mathcal{M}, s) \not\models \forall x\, p(x)$.

Anche nel caso della logica dei predicati, per dimostrare semanticamente che una formula A è valida, si può ragionare per assurdo: si assume che esistano una interpretazione $\mathcal{M}$ e un'assegnazione s su $\mathcal{M}$ tali che $(\mathcal{M}, s) \not\models A$, e si deriva una contraddizione. Per dimostrare invece che la formula A non è valida, se ne deve costruire un contromodello (cioè un modello $\mathcal{M}$ e un'assegnazione s tali che $(\mathcal{M}, s) \not\models A$).

Una relazione importante che vale tra la nozione di conseguenza logica e la soddisfacibilità di insiemi è enunciata nel seguente teorema.

Teorema 3.1 *Sia S un insieme di formule e A una formula. $S \models A$ sse $S \cup \{\neg A\}$ è insoddisfacibile.*

Dimostrazione. Dimostriamo separatamente la parte "se" e "solo se".

- Se $S \models A$ allora $S \cup \{\neg A\}$ è insoddisfacibile. La dimostrazione è per contrapposizione: dimostriamo che se $S \cup \{\neg A\}$ è soddisfacibile allora $S \not\models A$. Se $S \cup \{\neg A\}$ è soddisfacibile, allora esiste un'interpretazione $\mathcal{M}$ di $S \cup \{\neg A\}$ e un'assegnazione s su $\mathcal{M}$ tali che $(\mathcal{M}, s) \models C$ per ogni $C \in S$ e $(\mathcal{M}, s) \not\models A$. Quindi $S \not\models A$.
- Se $S \cup \{\neg A\}$ è insoddisfacibile, allora $S \models A$. Anche in questo caso la dimostrazione è per contrapposizione: dimostriamo che se $S \not\models A$, allora $S \cup \{\neg A\}$ è soddisfacibile. Assumiamo che $S \not\models A$: quindi esiste una interpretazione $\mathcal{M}$ e un'assegnazione s su $\mathcal{M}$ tali che $(\mathcal{M}, s) \models C$ per ogni $C \in S$ e $(\mathcal{M}, s) \not\models A$. Di conseguenza, $(\mathcal{M}, s) \models \neg A$. Dunque $S \cup \{\neg A\}$ è soddisfacibile.

Riportiamo qui di seguito alcune formule valide. Per la prima di esse, forniamo la dimostrazione di validità, come esempio. Le altre dimostrazioni sono lasciate per esercizio.

1. Istanziazione: $\forall x\, p(x) \rightarrow p(t)$.
   Per dimostrare la validità di questa formula, ragioniamo per assurdo. Assumiamo che la formula non sia valida. Dunque esiste una sua interpretazione $\mathcal{M}$ e un'assegnazione s su $\mathcal{M}$ tali che $(\mathcal{M}, s) \not\models \forall x\, p(x) \rightarrow p(t)$; cioè valgono (a) e (b):

   (a) $(\mathcal{M}, s) \models \forall x\, p(x)$
   Quindi per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \models p(x)$, cioè per ogni $d \in D$: $s[d/x] \in \mathcal{M}(p)$, dunque per ogni $d \in D$: $d \in \mathcal{M}(p)$.

   (b) $(\mathcal{M}, s) \not\models p(t)$. Dunque, se $s(t) = d^*$, $d^* \notin \mathcal{M}(p)$.

   Da (a) e (b) si deriva una contraddizione. Quindi non esiste alcuna interpretazione in cui $\forall x\, p(x) \rightarrow p(t)$ è falsa.

2. Interdefinibilità dei quantificatori:

$$\forall x A \equiv \neg \exists x \neg A$$
$$\exists x A \equiv \neg \forall x \neg A$$

   Di conseguenza sono valide anche le formule $\neg \forall x A \equiv \exists x \neg A$ e $\neg \exists x A \equiv \forall x \neg A$.

3. Regole di distribuzione:

$$\forall x (A \rightarrow B) \rightarrow (\forall x A \rightarrow \forall x B)$$
$$\forall x (A \rightarrow B) \rightarrow (\exists x A \rightarrow \exists x B)$$
$$\forall x (A \wedge B) \equiv (\forall x A \wedge \forall x B)$$
$$\exists x (A \vee B) \equiv (\exists x A \vee \exists x B)$$

4. $(\forall xA \vee \forall xB) \to \forall x(A \vee B)$

5. $\exists x(A \wedge B) \to \exists xA \wedge \exists xB$

6. $\exists y \forall xA \to \forall x \exists yA$

7. se A non contiene x libera, allora $\models \forall x(A \to B) \to (A \to \forall xB)$

8. se A non contiene x libera, allora $\models \forall x(B \to A) \to (\exists xB \to A)$

Le formule che seguono non sono invece valide. Per la prima di esse lo dimostriamo, fornendone un controesempio, cioè un'istanza della formula con un appropriato contromodello. Per le altre forniamo alcuni suggerimenti, lasciando per esercizio la costruzione di un contromodello.

1. $(\forall xA \to \forall xB) \to \forall x(A \to B)$

   Qui A e B sono "metaformule", quindi l'espressione sopra riportata è uno "schema" di formula. Consideriamo la seguente istanza dello schema:

   $$(\forall xp(x) \to \forall xq(x)) \to \forall x(p(x) \to q(x))$$

   dove p e q sono simboli di predicato. Un caso particolare in cui $\forall xp(x) \to \forall xq(x)$ è vera è quando $\forall xp(x)$ è falsa (non tutti gli oggetti del dominio sono in $\mathcal{M}(p)$). Ma non è detto che, in questo caso, ogni oggetto in $\mathcal{M}(p)$ sia anche in $\mathcal{M}(q)$.

   Un contromodello $\mathcal{M}$ della formula è dunque l'interpretazione con dominio $D = \{0, 1\}$, $\mathcal{M}(p) = \{0\}$, $\mathcal{M}(q) = \emptyset$. Dimostrare per esercizio che questo costituisce effettivamente un contromodello (per qualsiasi assegnazione s).

2. $\forall x(A \vee B) \to \forall xA) \vee \forall xB$

   Si consideri $A = p(x)$ e $B = q(x)$ e pensare alle interpretazioni di p e q: "essere pari" e "essere dispari".

3. $\exists xA \wedge \exists xB \to \exists x(A \wedge B)$

   Come al punto precedente

4. $\forall x \exists yA \to \exists y \forall xA$

   Pensare alla differenza tra continuità (per ogni $x_0 \in I$ esiste $\delta > 0$ ... : δ dipende da x_0) e continuità uniforme (esiste $\delta > 0$ tale che per ogni $x_0 \in I$ ... : δ è lo stesso per ogni punto).

3.3.5 Forme normali prenesse

Una formula si dice in *forma normale prenessa* se essa ha la forma:

$$Q_1 x_1 Q_2 x_2 ... Q_n x_n \, A$$

dove $Q_1 ... Q_n$ sono quantificatori e A non contiene quantificatori. La sequenza di quantificatori $Q_1 x_1 Q_2 x_2 ... Q_n x_n$ si chiama *prefisso* della formula e A si chiama la *matrice* della formula.

Ogni formula è logicamente equivalente a una formula in forma normale prenessa. Per trasformare una formula in forma prenessa si utilizzano le seguenti equivalenze logiche:

- Ridenominazione di variabili vincolate: se y è una variabile nuova (che non occorre in $\forall x A$), allora

  $$\begin{array}{rcl} \forall x A & \leftrightarrow & \forall y A[y/x] \\ \exists x A & \leftrightarrow & \exists y A[y/x] \end{array}$$

  Il primo passo nella trasformazione di una formula in forma normale prenessa consiste nel ridenominare tutte le variabili vincolate in modo che ciascun quantificatore agisca su una variabile distinta. A conclusione delle ridenominazioni quindi, ciascuna variabile vincolata occorre soltanto nello scopo del proprio quantificatore.

- Se x non occorre in B, allora:

  $$\begin{array}{rcl} \forall x A \wedge B & \leftrightarrow & \forall x(A \wedge B) \\ \exists x A \wedge B & \leftrightarrow & \exists x(A \wedge B) \\ \forall x A \vee B & \leftrightarrow & \forall x(A \vee B) \\ \exists x A \vee B & \leftrightarrow & \exists x(A \vee B) \\ \forall x A \rightarrow B & \leftrightarrow & \exists x(A \rightarrow B) \\ \exists x A \rightarrow B & \leftrightarrow & \forall x(A \rightarrow B) \\ B \rightarrow \forall x A & \leftrightarrow & \forall x(B \rightarrow A) \\ B \rightarrow \exists x A & \leftrightarrow & \exists x(B \rightarrow A) \\ \neg \forall x A & \leftrightarrow & \exists x \neg A \\ \neg \exists x A & \leftrightarrow & \forall x \neg A \end{array}$$

  Applicando questa seconda serie di trasformazioni (da sinistra verso destra), è possibile "portar fuori" tutti i quantificatori, dato che le variabili sono state ridenominate. Si noti che la "natura" di un quantificatore che "esce" dall'antecedente di un'implicazione si inverte: un universale sull'antecedente diventa un esistenziale esterno, e viceversa. Per ricordarlo, si pensi ad una trasformazione per stadi:

  $$\forall x A \rightarrow B \Longrightarrow \neg \forall x A \vee B \Longrightarrow \exists x \neg A \vee B \Longrightarrow \exists x(\neg A \vee B) \Longrightarrow \exists x(A \rightarrow B)$$

Ad esempio, per trasformare la formula $\neg(\forall x \exists y\, p(x,y) \vee (\forall x\, p(x,c) \rightarrow \exists y\, q(y)))$ in forma normale prenessa, possiamo procedere in questo modo:

$$\begin{array}{ll} \multicolumn{2}{l}{\neg(\forall x \exists y\, p(x,y) \vee (\forall x\, p(x,c) \rightarrow \exists y\, q(y)))} \\ \Longrightarrow & \neg(\forall x \exists y\, p(x,y) \vee (\forall x_1\, p(x_1,c) \rightarrow \exists y_1\, q(y_1))) \\ \Longrightarrow & \neg \forall x(\exists y\, p(x,y) \vee \exists x_1\, (p(x_1,c) \rightarrow \exists y_1\, q(y_1))) \\ \Longrightarrow & \neg \forall x \exists y\, (p(x,y) \vee \exists x_1 \exists y_1\, (p(x_1,c) \rightarrow q(y_1))) \\ \Longrightarrow & \neg \forall x \exists y \exists x_1 \exists y_1\, (p(x,y) \vee (p(x_1,c) \rightarrow q(y_1))) \\ \Longrightarrow & \exists x \forall y \forall x_1 \forall y_1\, \neg(p(x,y) \vee (p(x_1,c) \rightarrow q(y_1))) \end{array}$$

Su [7] si può trovare un programma OCaml (`fol.ml`) nel quale sono definite diverse operazioni sulle formule, tra cui la trasformazione di formule in forma normale prenessa, basato sul metodo sopra descritto.

3.3.6 Logica dei predicati e linguaggio naturale

Il compito di rappresentare in logica dei predicati enunciati del linguaggio naturale non è sempre facile. Consideriamo innanzitutto esempi dei più semplici enunciati che coinvolgono i quantificatori.

1. "Esiste un corvo nero". Se il dominio dell'interpretazione è costituito soltanto da corvi, possiamo rappresentare questo enunciato in un linguaggio che contiene soltanto il simbolo di predicato *nero*, a un posto, mediante la formula $\exists x\ nero(x)$. Ma se

l'universo del discorso (il dominio) include anche oggetti che non sono corvi si deve includere nel linguaggio anche un predicato che rappresenti la proprietà di essere un corvo: *corvo*, a un posto. L'enunciato significa allora che esiste un oggetto del dominio che è un corvo ed è nero: $\exists x(corvo(x) \wedge nero(x))$.

Dal punto di vista insiemistico, l'affermazione "esiste un corvo nero" significa che l'intersezione dell'insieme dei corvi e dell'insieme degli oggetti neri non è vuota: esiste un oggetto appartenente a tale intersezione (la congiunzione corrisponde infatti all'intersezione insiemistica, così come la disgiunzione corrisponde all'unione).

È un errore invece rappresentare la frase "esiste un corvo nero" mediante la formula: $A = \exists x(corvo(x) \rightarrow nero(x))$. Questa formula significa che esiste un oggetto tale che, se tale oggetto è un corvo, allora esso è nero. A causa del significato dell'implicazione, che è vera ogni volta che l'antecedente è falso, la formula A è vera in ogni interpretazione in cui esista un oggetto d^* che non è un corvo (oppure esiste almeno un oggetto nero d^*). Infatti:

Se	$d^* \notin \mathcal{M}(corvo)$ (d^* non è un corvo)
oppure se	$d^* \in \mathcal{M}(nero)$ (d^* è nero)
allora per ogni s:	$(\mathcal{M}, s[d^*/x]) \not\models corvo(x)$ oppure $(\mathcal{M}, s[d^*/x]) \models nero(x)$.
Questo vale sse	$(\mathcal{M}, s[d^*/x]) \models corvo(x) \rightarrow nero(x)$,
che implica	esiste $d \in D$ tale che $(\mathcal{M}, s[d/x]) \models corvo(x) \rightarrow nero(x)$,
cioè	$(\mathcal{M}, s) \models \exists x(corvo(x) \rightarrow nero(x))$

Si ricordi sempre che $\exists x(corvo(x) \rightarrow nero(x))$ è logicamente equivalente a $\exists x(\neg corvo(x) \vee nero(x))$.

2. "Tutti i corvi sono neri". Anche in questo caso, se il dominio dell'interpretazione è costituito soltanto da corvi, possiamo rappresentare questo enunciato semplicemente mediante la formula $\forall x\, nero(x)$. Ma se il dominio include anche oggetti che non sono corvi, l'enunciato significa allora che per ogni oggetto x del dominio, se x è un corvo allora x è nero (sugli oggetti che non sono corvi non si dice niente): $\forall x(corvo(x) \rightarrow nero(x))$.

   Dal punto di vista insiemistico, l'affermazione "tutti i corvi sono neri" significa che l'insieme dei corvi è incluso nell'insieme degli oggetti neri.

   Se nella formula data sopra sostituiamo l'implicazione con la congiunzione: $\forall x(corvo(x) \wedge nero(x))$, si ottiene un diverso significato: tutti gli oggetti del dominio sono corvi e sono neri. La formula è infatti logicamente equivalente a $\forall x\, corvo(x) \wedge \forall x\, nero(x)$.

   Si noti che la formula $\forall x(corvo(x) \rightarrow nero(x))$ è vera in ogni interpretazione in cui non esistano corvi. Infatti:

   Per ogni $d \in D$, $d \notin \mathcal{M}(corvo)$ (non ci sono corvi)

vale sse:	per ogni s e per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \not\models corvo(x)$.
Ciò implica che:	per ogni s e per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \not\models corvo(x)$ oppure $(\mathcal{M}, s[d/x]) \models nero(x)$,
che equivale a:	per ogni s e per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \models corvo(x) \rightarrow nero(x)$,
cioè:	per ogni s, $(\mathcal{M}, s) \models \forall x(corvo(x) \rightarrow nero(x))$.

   La stessa conclusione si può ottenere con un altro ragionamento. La formula $\forall x(corvo(x) \rightarrow nero(x))$ è falsa sse esiste un oggetto d tale che $(\mathcal{M}, s[d/x]) \not\models corvo(x) \rightarrow nero(x)$, cioè tale che: $(\mathcal{M}, s[d/x]) \models corvo(x)$ e $(\mathcal{M}, s[d/x]) \not\models nero(x)$, cioè se ne esiste un controesempio: un corvo che non è nero. In particolare, deve esistere un corvo. Se non esistono corvi, non possiamo trovare controesempi e $\forall x(corvo(x) \rightarrow nero(x))$ non può essere falsa, quindi è vera. D'altronde $\forall x(corvo(x) \rightarrow nero(x))$ è logicamente equivalente a $\forall x(\neg corvo(x) \vee nero(x))$, che è vera nel caso particolare in cui sia vera $\forall x \neg corvo(x)$.

Se non esistono corvi, siamo in un caso in cui l'enunciato "tutti i corvi sono neri" è *vero a vuoto*: è vero semplicemente perché afferma che l'insieme vuoto (quello dei corvi) è incluso in un altro insieme (quello degli oggetti neri).

Consideriamo ora esempi più complessi, individuando una metodologia per trovarne una rappresentazione corretta.

1. "Tutte le scimmie sono fuggite su un albero". In questo caso non possiamo assumere ipotesi esemplificative, quale ad esempio quella che il dominio sia costituito soltanto da scimmie (esistono anche alberi). Per rappresentare l'enunciato occorrono dunque predicati a un posto per rappresentare l'insieme delle scimmie e degli alberi (*scimmia* e *albero*) e un simbolo di predicato binario $fugge$ che rappresenta la relazione "essere fuggito su".

   Analizziamo la frase con metodologia *top-down*: si tratta di un'affermazione universale sulle scimmie, rappresentata quindi correttamente da una formula della forma:

   $$\forall x\,(scimmia(x) \rightarrow A(x)) \tag{3.1}$$

   Qui $A(x)$ deve rappresentare l'enunciato "x è fuggito su un albero".

   Questo secondo enunciato è un'affermazione esistenziale: esiste un albero su cui x è fuggito. Possiamo quindi rappresentarlo con il quantificatore esistenziale e la congiunzione:

   $$\exists y\,(albero(y) \wedge fugge(x,y))$$

   Sostituiamo quindi questa formula a $A(x)$ in 3.1 ottenendo:

   $$\forall x\,(scimmia(x) \rightarrow \exists y\,(albero(y) \wedge fugge(x,y))) \tag{3.2}$$

   Questa formula è logicamente equivalente a:

   $$\forall x \exists y\,(scimmia(x) \rightarrow albero(y) \wedge fugge(x,y))$$

   che costituisce dunque una rappresentazione di "tutte le scimmie sono fuggite su un albero"– sintatticamente diversa ma semanticamente equivalente alla 3.2.

   Si noti che l'enunciato "tutte le scimmie sono fuggite su un albero" è in realtà ambiguo: non è chiaro se le scimmie sono fuggite tutte sullo stesso albero oppure su alberi diversi. La formula 3.2 rappresenta questa seconda interpretazione, più "liberale": l'albero y su cui fugge x può dipendere da x.

   Se l'albero è lo stesso per tutte le scimmie, dobbiamo riformulare l'enunciato in "esiste un albero su cui sono fuggite tutte le scimmie". In questo caso si tratta di un'affermazione esistenziale, quindi rappresentabile da una formula della forma:

   $$\exists y\,(albero(y) \wedge B(y))$$

   Qui $B(y)$ rappresenta la proprietà (di y): tutte le scimmie sono fuggite su y. $B(y)$ può quindi essere espansa in:

   $$\forall x\,(scimmia(x) \rightarrow fugge(x,y))$$

   La formula che rappresenta dunque correttamente "tutte le scimmie sono fuggite su uno stesso albero" è:

   $$\exists y\,(albero(y) \wedge \forall x\,(scimmia(x) \rightarrow fugge(x,y))) \tag{3.3}$$

Si noti che la formula 3.2 potrebbe essere vera in un mondo in cui non esistono alberi: infatti, essa è vera comunque quando non esistono scimmie. Al contrario, anche nel caso in cui non esistano scimmie, la 3.3 richiede l'esistenza di un albero.

Osserviamo infine la formula seguente, che potrebbe a prima vista sembrare una rappresentazione corretta dell'enunciato che stiamo considerando:

$$\forall x \exists y\, (scimmia(x) \wedge albero(y) \wedge fugge(x, y))$$

Poiché y non occorre in $scimmia(x)$, questa formula è equivalente a

$$\forall x\, (scimmia(x) \wedge \exists y\, (albero(y) \wedge fugge(x, y)))$$

e a:

$$\forall x\ scimmia(x) \wedge \forall x \exists y\, (albero(y) \wedge fugge(x, y))$$

Quindi afferma che tutti gli oggetti del dominio sono scimmie e che tutti gli oggetti del dominio sono fuggiti su un albero.

2. "Esiste una tartaruga che è più vecchia di qualunque essere umano". Questa è un'affermazione esistenziale: esiste un oggetto x che è una tartaruga e ha la proprietà $C(x)$ di essere più vecchia di qualunque essere umano:

$$\exists x\, (tartaruga(x) \wedge C(x))$$

Come possiamo espandere $C(x)$? Questa è un'affermazione di tipo universale: per ogni y, se y è un essere umano, allora x è più vecchia di y:

$$\forall y\, (umano(y) \rightarrow piu\_vecchio(x, y))$$

Sostituendo, otteniamo:

$$\exists x\, (tartaruga(x) \wedge \forall y\, (umano(y) \rightarrow piu\_vecchio(x, y))) \tag{3.4}$$

Questa formula è logicamente equivalente a:

$$\exists x \forall y\, (tartaruga(x) \wedge (umano(y) \rightarrow piu\_vecchio(x, y))) \tag{3.5}$$

Perché invece la formula

$$\exists x\, (tartaruga(x) \rightarrow \forall y\, (umano(y) \rightarrow piu\_vecchio(x, y)))$$

non rappresenta correttamente l'enunciato? Basta notare che essa è vera in un mondo in cui non esistano tartarughe, mentre l'enunciato che stiamo considerando afferma l'esistenza di almeno una tartaruga.

La formula

$$\exists x\, (tartaruga(x) \wedge \forall y\, (umano(y) \wedge piu\_vecchio(x, y)))$$

analogamente, non è una buona rappresentazione: perché sia vera, occorre, in particolare, che tutti gli oggetti del dominio siano umani e che x sia più vecchia di tutti gli oggetti del dominio (inclusa la tartaruga stessa).

Consideriamo infine:

$$\exists x \forall y\, (tartaruga(x) \wedge umano(y) \rightarrow piu\_vecchio(x, y))$$

(attenzione: poiché $\wedge$ ha un ordine di precedenza maggiore di $\rightarrow$, questa formula non è equivalente alla 3.5, ma è la stessa cosa di:

$$\exists x \forall y\,((tartaruga(x) \wedge umano(y)) \rightarrow piu\_vecchio(x, y)) \tag{3.6}$$

Anche questa formula è vera quando non esistono tartarughe. Possiamo dimostrarlo semanticamente, oppure trasformando la "matrice" della formula in forma normale disgiuntiva. Se eliminiamo l'implicazione e applichiamo la legge di De Morgan che consente di portare la negazione all'interno di una congiunzione, otteniamo la formula (equivalente alla 3.6)

$$\exists x \forall y\,(\neg tartaruga(x) \vee \neg umano(y) \vee piu\_vecchio(x, y))$$

Perché questa formula sia vera, è sufficiente che esista un oggetto che non è una tartaruga, oppure che nessun oggetto sia un essere umano, oppure che esista un oggetto più vecchio di tutti gli altri.

3. "Il papà di Gennaro batte a scopone i papà di tutti i bambini del quartiere di Santa Lucia". Iniziamo a determinare il linguaggio in cui esprimere questo enunciato. Gennaro sarà denotato da una costante, *gennaro*. Per indicare il padre di qualcuno, possiamo usare un simbolo funzionale a un posto (in questo caso si assume che tutti gli oggetti del dominio abbiano un padre), oppure un simbolo di predicato a due posti, che denota la relazione "essere padre di". Adottiamo, in prima istanza, la prima soluzione, in quanto conduce a una rappresentazione più semplice. Se il simbolo funzionale utilizzato è *padre_di*, assumiamo dunque che la funzione denotata da *padre_di* sia definita per tutti gli argomenti del dominio. Utilizziamo poi il predicato *vince*, a due posti (per la relazione "battere a scopone"), il predicato *bambino*, a un posto ("essere un bambino") e il predicato *santa_lucia*, a un posto, per denotare la proprietà di abitare nel quartiere di Santa Lucia. Quest'ultima potrebbe essere analizzata in termini di una relazione a due posti, "abitare", il cui secondo argomento è il posto dove si abita. Preferiamo la prima soluzione per due motivi: innanzitutto essa è sufficiente a rappresentare adeguatamente la proposizione data, in secondo luogo ci consente di assumere che il dominio dell'interpretazione "intesa" sia costituito da esseri umani, rendendo ragionevole l'ipotesi che la funzione denotata da *padre_di* sia definita per tutti gli elementi del dominio (altrimenti quale sarebbe il padre del quartiere di Santa Lucia?). In realtà sarebbe addirittura sufficiente utilizzare, al posto di *bambino* e *santa_lucia*, un unico predicato a un posto che denoti la proprietà "essere un bambino del quartiere di Santa Lucia".

   La frase considerata si può analizzare in termini di un'affermazione universale, relativa a tutti i papà dei bambini del quartiere di Santa Lucia. Se $A(x)$ denota la proprietà "essere un papà di un bambino del quartiere di Santa Lucia" e $B(x)$ denota la proprietà di essere battuto a scopone dal papà di Gennaro, la formula che vogliamo costruire avrà la forma:

$$\forall x\,(A(x) \rightarrow B(x)) \tag{3.7}$$

   Vediamo ora quale formula del linguaggio scelto possiamo sostituire al posto di $A(x)$. Possiamo analizzare "x è il papà di un bambino del quartiere di Santa Lucia" riconoscendo la forma di un'affermazione esistenziale: "esiste un bambino y che abita nel quartiere di Santa Lucia e di cui y è il padre":

$$\exists y\,(bambino(y) \wedge santa\_lucia(y) \wedge P(x, y))$$

   dove $P(x, y)$ deve denotare "x è il padre di y". Per "espandere" $P(x, y)$ utilizziamo il simbolo di uguaglianza: $x = padre\_di(y)$. La formula $A(x)$ è dunque:

$$\exists y\,(bambino(y) \wedge santa\_lucia(y) \wedge x = padre\_di(y))$$

Consideriamo ora la formula $B(x)$ in 3.7. Il papà di Gennaro è denotato dal termine $padre\_di(gennaro)$. La formula $B(x)$ può essere facilmente sostituita da $vince(padre\_di(gennaro), x)$.

Otteniamo quindi:

$$\forall x\,(\exists y\,(bambino(y) \land santa\_lucia(y) \land x = padre\_di(y)) \\ \rightarrow vince(padre\_di(gennaro), x))$$

Osserviamo che, poiché y non occorre in $vince(padre\_di(gennaro), x)$, questa formula è equivalente a:

$$\forall x\,\forall y\,(bambino(y) \land santa\_lucia(y) \land x = padre\_di(y) \\ \rightarrow vince(padre\_di(gennaro), x))$$

che possiamo leggere "per ogni x e per ogni y, se y è un bambino che abita nel quartiere di Santa Lucia, e x è il padre di y, allora il papà di Gennaro vince x a scopone".

Se anziché il simbolo funzionale $padre\_di$ utilizziamo un simbolo di predicato a due posti, $padre$, la frase va considerata come un'affermazione esistenziale: esiste z che è il padre di Gennaro e che vince a scopone ...". La rappresentazione è dunque:

$$\exists z\,(padre(z, gennaro) \land \\ \forall x\,\forall y\,(bambino(y) \land santa\_lucia(y) \land padre(x, y) \\ \rightarrow vince(z, x))$$

3.3.7 Il significato dei quantificatori

Per il significato del quantificatore universale, in generale, vorremmo che se $\forall xA$ è vero in un'interpretazione, allora anche $A[t/x]$ sia vero, per ogni termine t. Dunque ci chiediamo: se t è un termine, è sempre vero che $\models \forall xA \rightarrow A[t/x]$?

Ad esempio, consideriamo un linguaggio per l'aritmetica, con un numero infinito di costanti $0, 1, 2, ...$, con i simboli funzionali $+, -, \times$ ed i simboli di predicato $=$ e $<$, che scriveremo infissi, nel modo abituale. Nell'interpretazione "standard" di questo linguaggio è vero che $\forall x \exists y\,(x < y)$. Quindi è vero anche $\exists y\,(0 < y)$, $\exists y\,(1 < y)$, ..., $\exists y\,(2+1 < y)$, ... Ma consideriamo ora il termine $y + 1$: è vero che $\exists y\,(y + 1 < y)$? Chiaramente no. Quindi non è corretto dire che *per ogni* termine t, $\forall x \exists y\,(x < y) \rightarrow \exists y\,(t < y)$ è vero nell'interpretazione standard dell'aritmetica (quindi, a maggior ragione, la formula non è valida).

Questo è dovuto al fatto che la formula $A[t/x]$ non "dice" sempre di t quello che A dice di x. Nell'esempio sopra considerato, $\exists y\,(x < y)$ "dice" che esiste un numero maggiore di x, ma $\exists y\,(y+1 < y)$ non dice che esiste un numero maggiore di $y+1$. Come ulteriore esempio, consideriamo la formula $A = \exists y(x = 2 \times y)$ e il termine $t = y + 1$; la formula A "dice" che x è pari; ma $A[y + 1/x] = \exists y(y + 1 = 2 \times y)$ non dice che $y + 1$ è pari, ma che esiste una soluzione dell'equazione $y + 1 = 2 \times y$.

Per garantire la validità di $\forall xA \rightarrow A[t/x]$, ci si deve assicurare che nessuna variabile in t venga "catturata" da un quantificatore in A. A questo scopo serve la definizione che segue:

Definizione 3.3.8 *Un termine t è* sostituibile per x in A *sse nessuna occorrenza libera di x in A si trova nel campo d'azione di un quantificatore $\forall y$ o $\exists y$, dove y occorre in t.*

Ad esempio: $f(y)$ è sostituibile per x in $\forall z\,p(z, x)$, e in $\forall x\,p(z, x)$. Ma non è sostituibile per x in $\forall y\,p(y, x)$, né in $\exists y\,p(y, x)$.

Come caso particolare, se t è un termine chiuso, t è sempre sostituibile per qualsiasi variabile in una formula. Inoltre x è sempre sostituibile per x in qualsiasi formula.

Sotto la condizione che t sia libero per x in A si può garantire che $\forall xA \rightarrow A[t/x]$ è valida, ed anche che $A[t/x] \rightarrow \exists xA$ è valida (vedi Teorema 3.2 nel paragrafo seguente). In particolare dunque, se t è un termine chiuso è sempre vero che $\models \forall xA \rightarrow A[t/x]$.

3.3.8 Il teorema di sostituzione ed altri teoremi

In questo paragrafo vengono enunciati e dimostrati alcuni teoremi relativi alla sostituzione. Iniziamo con alcuni risultati relativi alla sostituzione di variabili.

Il risultato che segue stabilisce che l'interpretazione $s(t)$ di un termine t dipende soltanto dal valore assegnato da s alle variabili che occorrono effettivamente in t.

Lemma 3.3.9 (Lemma di coincidenza) *Se t è un termine e s e s' sono assegnazioni su $\mathcal{M}$ tali che $s(x) = s'(x)$ per ogni variabile x che occorre in t (cioè s e s' coincidono su tutte le variabili che occorrono in t), allora $s(t) = s'(t)$.*

Ioltre, se A è una formula e se s e s' sono assegnazioni su $\mathcal{M}$ tali tali che $s(x) = s'(x)$ per ogni variabile x che occorre libera in A (cioè s e s' coincidono su tutte le variabili che occorrono libere in A), allora $(\mathcal{M}, s) \models A$ sse $(\mathcal{M}, s') \models A$.

La dimostrazione, per induzione sui termini e sulle formule, è lasciata per esercizio.

Il risultato che segue stabilisce, in termini semplici, che sostituire in un termine (in una formula) oppure in un'assegnazione produce lo stesso risultato. Il risultato infatti si applica anche quando $s' = s[s(t)/x]$, stabilendo dunque che $s[s(t)/x](u) = s(u[t/x])$ e che $(\mathcal{M}, s[s(t)/x]) \models A$ sse $(\mathcal{M}, s) \models A[t/x]$.

Lemma 3.3.10 (Lemma di sostituzione) *Siano t e u termini, $\mathcal{M}$ un'interpretazione e s, s' assegnazioni su $\mathcal{M}$ tali che $s'(x) = s(t)$ e per ogni variabile y diversa da x che occorre in u, $s(y) = s'(y)$. Allora $s'(u) = s(u[t/x])$.*

Inoltre, se A è una formula, e se t è libero per x in A, allora $(\mathcal{M}, s') \models A$ sse $(\mathcal{M}, s) \models A[t/x]$.

Dimostrazione. La prima parte del lemma di sostituzione si dimostra per induzione sui termini. Se u coincide con la variabile x allora si ha:

$$s'(u) = s'(x) = s(t) = s(u[t/x])$$

Se invece u è una variabile y distinta da x allora risulta:

$$s'(u) = s'(y) = s(y) = s(y[t/x]) = s(u[t/x])$$

Se $u = f(u_1, ..., u_n)$, si assuma, per ipotesi induttiva che, per ogni $i = 1, ..., n$, $s'(u_i) = s(u_i[t/x])$. Allora:

$$s'(u) = s'(f(u_1, ..., u_n)) = \mathcal{M}(f)(s'(u_1), ..., s'(u_n)) =$$

$$\mathcal{M}(f)(s(u_1[t/x]), ..., s(u_n[t/x])) = s(f(u_1[t/x], ..., u_n[t/x])) = s(u[t/x])$$

Per dimostrare la seconda parte, assumiamo che t sia libero per x in A. La dimostrazione è per induzione su A.

1. Se $A = p(u_1, ..., u_n)$ allora $(\mathcal{M}, s') \models A$ sse $(\mathcal{M}, s') \models p(u_1, ..., u_n)$ sse $\langle s'(u_1), ..., s'(u_n)\rangle \in \mathcal{M}(p)$. Per la prima parte del Lemma, $s'(u_i) = s(u_i[t/x])$, quindi $(\mathcal{M}, s') \models A$ sse $\langle s(u_1[t/x]), ..., s(u_n[t/x])\rangle \in \mathcal{M}(p)$ sse $(\mathcal{M}, s) \models A[t/x]$.

2. Se A ha la forma $\neg B$ oppure $B \bullet C$ dove $\bullet$ è un connettivo binario, la tesi segue immediatamente dall'ipotesi induttiva.

3. Sia $A = \forall y B$. Se $y = x$ allora $A[t/x] = A$, quindi $(\mathcal{M}, s') \models A$ sse $(\mathcal{M}, s') \models A[t/x]$. D'altronde, poiché s e s' coincidono su tutte le variabili che occorrono libere in A (differiscono solo su x che non occorre libera in A), per il Lemma 3.3.9, $(\mathcal{M}, s') \models A[t/x]$ sse $(\mathcal{M}, s) \models A[t/x]$. Assumiamo dunque che $y \neq x$. Allora $(\mathcal{M}, s') \models A$ sse per ogni $d \in D$: $(\mathcal{M}, s'[d/y]) \models B$. Distinguiamo due casi:

(a) Se x non occorre in B, allora $B = B[t/x]$ e poiché s e s' differiscono soltanto per l'interpretazione di x, che non è tra le variabili che occorrono in B, per il Lemma 3.3.9 $(\mathcal{M}, s'[d/y]) \models B$ sse $(\mathcal{M}, s[d/y]) \models B$ sse $(\mathcal{M}, s[d/y]) \models B[t/x]$.

(b) Se x occorre in B allora, poiché t è sostituibile per x in A, y non occorre in t, quindi per il Lemma 3.3.9 $s[d/y](t) = s(t)$. D'altronde, sempre per il Lemma 3.3.9, poiché assumiamo che $x \neq y$, $s'[d/y](x) = s'(x)$. Per ipotesi $s'(x) = s(t)$, quindi anche $s'[d/y](x) = s[d/y](t)$. Possiamo allora applicare l'ipotesi induttiva a B, ottenendo che $(\mathcal{M}, s'[d/y]) \models B$ sse $(\mathcal{M}, s[d/y]) \models B[t/x]$.

In entrambi i casi abbiamo dunque che per ogni $d \in D$: $(\mathcal{M}, s'[d/y]) \models B$ sse $(\mathcal{M}, s[d/y]) \models B[t/x]$, quindi $(\mathcal{M}, s') \models \forall x B$ sse $(\mathcal{M}, s[d/y]) \models \forall x B[t/x]$.

4. Il caso del quantificatore esistenziale è simile.

Dal Lemma 3.3.10 segue il teorema 3.2, che stabilisce che il significato dei quantificatori è effettivamente quello voluto.

Teorema 3.2 *Se t è sostituibile per x in A, allora*

$$\models \forall x A \to A[t/x] \qquad e \qquad \models A[t/x] \to \exists x A$$

Dimostrazione. Dimostriamo la prima affermazione per contrapposizione: assumiamo che esista un'interpretazione $\mathcal{M}$ e un'assegnazione s tali che $(\mathcal{M}, s) \not\models A[t/x]$. Sia $d = s(t)$. Poiché $s(t) = s[d/x](x)$, per il Lemma 3.3.10, $(\mathcal{M}, s[d/x]) \not\models A$. Quindi $(\mathcal{M}, s) \not\models \forall x A$.

La dimostrazione della seconda affermazione è lasciata per esercizio.

Il lemma seguente stabilisce che sostituire un sottotermine di un termine t con uno "equivalente" (con la stessa interpretazione) non cambia il significato di t. La semantica dei termini è dunque composizionale e rispetta il principio della sostituibilità degli equivalenti.

Lemma 3.3.11 (Sostituibilità di termini nei termini) *Siano t, t' e u termini e x una variabile. Se $\mathcal{M}$ è un'interpretazione e s un'assegnazione su $\mathcal{M}$, tale che $s(t) = s(t')$, allora*

$$s(u[t/x]) = s(u[t'/x])$$

Dimostrazione. La dimostrazione è per induzione sul termine u

1. Se u è una costante, allora $u[t/x] = u = u[t'/x]$
2. Se u è una variabile diversa da x, allora $u[t/x] = u = u[t'/x]$
3. Se $u = x$, allora $u[t/x] = t$ e $u[t'/x] = t'$. Per ipotesi $s(t) = s(t')$.
4. Sia $u = f(t_1, ..., t_n)$. Per ipotesi induttiva, per ogni t_i, $s(t_i[t/x]) = s(t_i[t'/x])$. Allora:

$$\begin{array}{ll} s(u[t/x]) & \\ = & s(f(t_1[t/x], ..., t_n[t/x])) \\ = & \mathcal{M}(f)(s(t_1[t/x]), ..., s(t_n[t/x])) \\ = & \mathcal{M}(f)(s(t_1[t'/x]), ..., s(t_n[t'/x])) \\ = & s(f(t_1[t'/x], ..., t_n[t'/x])) \\ = & s(u[t'/x]) \end{array}$$

Il lemma che segue stabilisce che sostituire un termine t con uno "equivalente" in una formula A non cambia il significato di A.

Lemma 3.3.12 (Sostituibilità di termini nelle formule) *Sia $\mathcal{M}$ un'interpretazione e s un'assegnazione su $\mathcal{M}$. Siano inoltre t e t' termini con $s(t) = s(t')$, e A una formula. Allora*

$$(\mathcal{M}, s) \models A[t/x] \text{ sse } (\mathcal{M}, s) \models A[t'/x]$$

Dimostrazione. La dimostrazione è per induzione sulla formula A.

1. A è atomica: $A = p(t_1, ..., t_n)$. Allora:
   $(\mathcal{M}, s) \models A[t/x]$
   sse $(\mathcal{M}, s) \models p(t_1[t/x], ..., t_n[t/x])$
   sse $< s(t_1[t/x]), ..., s(t_n[t/x]) > \in \mathcal{M}(p)$
   sse $< s(t_1[t'/x]), ..., s(t_n[t'/x]) > \in \mathcal{M}(p)$
   (per il Lemma 3.3.11)
   sse $(\mathcal{M}, s) \models p(t_1[t'/x], ..., t_n[t'/x])$
   sse $(\mathcal{M}, s) \models A[t'/x]$

2. $A = \neg B$. Per ipotesi induttiva:
   $(\mathcal{M}, s) \models B[t/x]$ sse $(\mathcal{M}, s) \models B[t'/x]$
   Quindi:
   $(\mathcal{M}, s) \models \neg B[t/x]$
   sse $(\mathcal{M}, s) \not\models B[t/x]$ sse $(\mathcal{M}, s) \not\models B[t'/x]$
   sse $(\mathcal{M}, s) \models \neg B[t'/x]$.

3. I casi in cui $A = B \wedge C$, $A = B \vee C$, $A = B \rightarrow C$ sono analoghi al precedente.

4. $A = \forall y B$. Per ipotesi induttiva, per ogni assegnazione s:
   $(\mathcal{M}, s) \models B[t/x]$ sse $(\mathcal{M}, s) \models B[t'/x]$.
   Se $y = x$, allora $\forall y B[t/x] = \forall y B = \forall y B[t'/x]$. Altrimenti:
   $(\mathcal{M}, s) \models \forall y B[t/x]$
   sse per ogni $d \in D$, $(\mathcal{M}, s[d/y]) \models B[t/x]$
   sse per ogni $d \in D$, $(\mathcal{M}, s[d/y]) \models B[t'/x]$ (per ipotesi induttiva)
   sse $(\mathcal{M}, s) \models \forall y B[t'/x]$

5. Il caso di $A = \exists x B$ è simile al precedente.

Il teorema di sostituzione che segue stabilisce che sostituire una sottoformula B con una logicamente equivalente in una formula A non cambia il significato di A.

Teorema 3.3 (Teorema di sostituzione) *Se $B \leftrightarrow B'$ e $A[B']$ si ottiene da A sostituendo qualche occorrenza di B con B', allora $A \leftrightarrow A[B']$*

Dimostrazione. Dimostriamo che per ogni interpretazione $\mathcal{M}$ e assegnazione s: $(\mathcal{M}, s) \models A$ sse $(\mathcal{M}, s) \models A[B']$.

La dimostrazione estende quella del teorema di sostituzione per la logica proposizionale (Teorema 2.1), aggiungendo i casi dei quantificatori all'induzione sulle formule che tratta il caso in cui $A[B'] \neq B'$ (il "caso speciale" trattato a parte è quello in cui invece $A[B'] = B'$).

1. Se $A = \exists x C$ e non siamo nel caso speciale: $A[B'] = \exists x C[B']$. Per ipotesi induttiva, $C \leftrightarrow C[B']$. Dunque:

   $$(\mathcal{M}, s) \models \forall (C \rightarrow C[B']) \quad \text{e} \quad (\mathcal{M}, s) \models \forall (C[B'] \rightarrow C)$$

   Per la seconda regola di distribuzione a pagina 42:

   $$(\mathcal{M}, s) \models \forall (\exists x C \rightarrow \exists x C[B']) \quad \text{e} \quad (\mathcal{M}, s) \models \forall (\exists x C[B'] \rightarrow \exists x C)$$

Cioè
$$(\mathcal{M}, s) \models \exists x C \rightarrow \exists x C[B'] \qquad \text{e} \qquad (\mathcal{M}, s) \models \exists x C[B'] \rightarrow \exists x C$$
Qindi $(\mathcal{M}, s) \models \exists x C$ sse $(\mathcal{M}, s) \models \exists x C[B']$.

2. Il caso in cui $A = \forall x C$ è lasciato per esercizio.

3.4 Sistemi di inferenza

Un sistema di inferenza è dato da un insieme di regole che gestiscono asserzioni che si intende provare. Ad esempio, le asserzioni possono essere del tipo “A è una formula, “La proposizione A è vera, “Il valore dell'espressione aritmetica E è v, ecc.

Una *regola di inferenza* è costituita da un numero finito (eventualmente nullo) di *premesse* e una *conclusione*. Una regola con premesse $P_1, ..., P_k$ e conclusione C viene abitualmente scritta
$$\frac{P_1, ..., P_k}{C}$$
Intuitivamente, tale regola permette di inferire (derivare, dedurre) C da $P_1, ..., P_k$; essa stabilisce cioè che, ogniqualvolta si abbiano ragioni sufficienti per asserire $P_1, ..., P_k$, si può sulla stessa base asserire C. Ad esempio, una regola della forma:
$$\frac{A \text{ è vera} \qquad B \text{ è vera}}{(A \text{ e } B) \text{ è vera}}$$
stabilisce che, per qualsiasi proposizione A e B, se si è stabilito che A è vera e B è vera, allora si può concludere che anche la congiunzione delle due proposizioni, $(A \text{ e } B)$, è vera.

Spesso alle regole è associato un nome, che consente di riferirsi ad esse. Il nome della regola viene abitualmente scritto accanto alla *linea di inferenza*, che separa le premesse dalla conclusione (alla sua destra o alla sua sinistra).

Quando il numero di premesse di una regola è nullo, essa viene detta *assioma*. Gli assiomi di un sistema di inferenza costituiscono cioè asserzioni che non hanno bisogno di ulteriore giustificazione. Normalmente negli assiomi la linea di inferenza viene omessa.

Gli esempi più semplici di sistemi di inferenza sono quelli che rispecchiano definizioni induttive, come quelli seguenti.

Esempio 3.4.1

1. *L'insieme delle espressioni aritmetiche costruite sull'alfabeto* $\{+, \times, 0, 1, 2, ...\}$ *è definito dal sistema di inferenza che gestisce espressioni della forma* $N : expr$ *(*N *è un'espressione) e consiste dei seguenti assiomi e regole:*
   - *Per ogni numero naturale* n, $n : expr$ *è un assioma.*
   - *Le regole di inferenza sono:*
   $$\frac{N : expr \quad M : expr}{(N + M) : expr}\ (+) \qquad\qquad \frac{N : expr \quad M : expr}{(N \times M) : expr}\ (\times)$$
2. *L'insieme delle formule proposizionali in* $Prop[P]$ *è definito mediante il sistema di inferenza che gestisce asserzioni della forma* $A : fbf$ *(*A *è una formula) e consiste dei seguenti assiomi e regole:*
   - *Per ogni* $p \in P$, $p : fbf$ *è un assioma.*

- *$\top : fbf$ e $\bot : fbf$ sono assiomi.*
- *Le regole di inferenza sono:*

$$\frac{A : fbf}{\neg A : fbf}\,(\neg) \qquad \frac{A : fbf \quad B : fbf}{(A \wedge B) : fbf}\,(\wedge)$$

$$\frac{A : fbf \quad B : fbf}{(A \vee B) : fbf}\,(\vee) \qquad \frac{A : fbf \quad B : fbf}{(A {\rightarrow} B) : fbf}\,(\rightarrow)$$

$$\frac{A : fbf \quad B : fbf}{(A \equiv B) : fbf}\,(\equiv)$$

Utilizzando le regole di un sistema si costruiscono nuovi oggetti sintattici: le *dimostrazioni.* Essenzialmente una dimostrazione è un albero, normalmente rappresentato con la radice in basso, le cui foglie sono etichettate da assiomi del sistema di inferenza e l'etichetta di ciascun nodo intermedio N è la conclusione di una regola di inferenza le cui premesse sono le etichette dei nodi immediatamente superiori a N. La nozione di dimostrazione può essere definita induttivamente.

Definizione 3.4.2 *Sia $\mathcal{S}$ un sistema di inferenza. L'insieme delle* dimostrazioni *in $\mathcal{S}$ è definito induttivamente come segue:*

1. *se E è un assioma di $\mathcal{S}$, allora esso stesso è una dimostrazione in $\mathcal{S}$ con conclusione E;*
2. *se $\dfrac{E_1 \quad \cdots \quad E_n}{E}$ è una regola di inferenza in $\mathcal{S}$ e $\Pi_1, \ \cdots \ \Pi_n$ sono dimostrazioni in $\mathcal{S}$ con conclusioni, rispettivamente $E_1 \ \cdots \ E_n$, allora*

$$\frac{\begin{matrix}(\Pi_1) \\ E_1\end{matrix} \quad \cdots \quad \begin{matrix}(\Pi_n) \\ E_n\end{matrix}}{E}$$

è una dimostrazione in $\mathcal{S}$ con conclusione E.

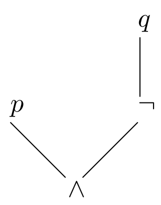
Esempio 3.4.3 *Quella seguente è una dimostrazione di $(p \wedge \neg q) : fbf$ nel sistema dell'esempio 3.4.1.3, quando P contiene i simboli p e q:*

$$(\wedge)\frac{(p)\dfrac{}{p : fbf} \qquad (\neg)\dfrac{(q)\dfrac{}{q : fbf}}{\neg q : fbf}}{(p \wedge \neg q) : fbf}$$

La radice dell'albero (in basso) è etichettata dalla conclusione *della dimostrazione.*

Se esiste una dimostrazione di un'asserzione A nel sistema di inferenza $\mathcal{S}$, si dice che A è un *teorema* di $\mathcal{S}$, o che A è derivabile in $\mathcal{S}$, e si scrive anche $\vdash_{\mathcal{S}} A$. Quando non vi sia possibilità di confusione, possiamo scrivere semplicemente $\vdash A$, omettendo l'indice $\mathcal{S}$. Ad esempio, la dimostrazione dell'esempio 3.4.3 stabilisce che $\vdash (p \wedge \neg q) : fbf$. Un sistema di inferenza caratterizza allora l'insieme delle espressioni che in esso sono derivabili.

Un'interessante caratteristica dei sistemi di inferenza considerati nell'esempio 3.4.1 è che le regole sono tutte *deterministiche*: le premesse sono determinate univocamente dalla conclusione. Quindi, se $\vdash A : fbf$, allora $\vdash A : fbf$ ha un'unica dimostrazione. Questa dimostrazione rispecchia la struttura profonda dell'espressione A. La struttura della dimostrazione è quella che viene considerata la *sintassi astratta* della formula. Infatti, quando si guarda la dimostrazione di $A : fbf$ osservando solo la struttura dell'albero e il nome delle regole, si ottiene l'abituale rappresentazione ad albero (rovesciata) di A. Si consideri ancora, ad esempio, la dimostrazione dell'esempio 3.4.3 e la si riscriva omettendo asserzioni e linee di inferenza; si ottiene l'albero:



La sintassi astratta è indipendente dalle diverse sintassi superficiali con le quali si può rappresentare una stessa espressione; ad esempio, un termine A può essere rappresentato in notazione prefissa o infissa, ma la dimostrazione di $A : fbf$ identifica una stessa sintassi astratta.

3.4.1 Linguaggio e metalinguaggio

...

Achille: Desidero poter esprimere CENTO desideri invece di tre!

...

Genio: Mi dispiace, Achille, ma non posso esaudire meta-desideri.

...

Achille: Dimmi, Genio, che cos'è un meta-desiderio?

Genio: È semplicemente un desiderio rigurardante altri desideri. E io non sono autorizzato ad esaudire meta-desideri. La mia competenza è limitata soltanto a desideri comuni, come avere dieci bottiglie di birra, incontrare Miss Universo a quattr'occhi o vincere un viaggio per due a Copacabana. [...] Vediamo se posso fare qualcosa. Mi ci vorrà un secondo ...

Achille: Che cos'è quella?

Genio: Questa è la mia Meta-Lampada...

(strofina la Meta-Lampada ...)

Meta-Genio: Io sono il Meta-Genio. Mi hai chiamato, o Genio?

(D. R. Hofstadter, *Gödel, Escher, Bach: un'Eterna Ghirlanda Brillante*, Adelphi Edizioni 1984)

Quando si parla di un linguaggio, si deve distinguere tra il linguaggio *di cui* si parla ed il linguaggio *in cui* si parla. Il primo è chiamato *linguaggio oggetto*, il secondo *metalinguaggio*. In generale, "Meta-X" significa "X sopra X"; così un metalinguaggio è un linguaggio su (per parlare di) un altro linguaggio, un metatermine è un termine (del metalinguaggio) che rappresenta un termine del linguaggio, una metadomanda è una domanda sulla domanda ("posso fare una domanda?"), ecc.

Quando diciamo, ad esempio, "sia A una formula, il simbolo A non è esso stesso una formula: normalmente non appartiene nemmeno all'alfabeto. A è in realtà un simbolo del metalinguaggio che sta ad indicare una formula qualsiasi del linguaggio oggetto. Diciamo allora che A è una *metaformula*.

La stessa scrittura delle regole di inferenza per la definizione delle espressioni ben formate di un linguaggio (vedi esempio 3.4.1) utilizza metaespressioni o metaformule. In una dimostrazione vera e propria, le metaespressioni devono essere sostituite da espressioni del linguaggio.

Quando studiamo un sistema di inferenza e dimostriamo proprietà che valgono per le sue dimostrazioni, facciamo una *metadimostrazione*. Ad esempio, si consideri la seguente affermazione: dato un sistema di inferenza $\mathcal{S}$ che definisce le espressioni ben formate di un linguaggio e un'asserzione E di tale sistema, allora esiste al massimo una dimostrazione di E (cioè, se E è dimostrabile, esiste un'unica dimostrazione di E). Questa è un'affermazione che parla di dimostrazioni. Il ragionamento che mostra la verità di tale affermazione è una dimostrazione che parla di dimostrazioni, dunque una metadimostrazione.

3.4.2 Ipotesi e derivazioni

Abbiamo osservato che la formulazione delle regole di un sistema di inferenza che definisce le espressioni ben formate di un linguaggio fa uso di metatermini. Si consideri il sistema di inferenza dell'esempio 3.4.1.2 e la seguente struttura sintattica:

$$\dfrac{A : fbf \qquad \dfrac{B : fbf}{\neg B : fbf}\,(\neg)}{(A \wedge \neg B) : fbf}\,(\wedge)$$

Essa appartiene al livello del metalinguaggio usato per parlare del sistema di inferenza dell'esempio 3.4.1.2 e si può considerare una prova del fatto che per ogni espressione A e B, se è dimostrabile che A è una formula ed è dimostrabile che B è una formula, allora è dimostrabile anche che $(A \wedge \neg B)$ è una formula. È evidente, infatti, che, qualora si disponga, per particolari espressioni A e B, di dimostrazioni di $A : fbf$ e $B : fbf$, esse possono essere utilizzate per costruire una dimostrazione di $(A \wedge \neg B) : fbf$.

La struttura sopra considerata differisce da una dimostrazione anche per un altro motivo: essa non è una dimostrazione in quanto le foglie dell'albero non sono assiomi del sistema di inferenza. Le foglie sono *ipotesi* o *assunzioni* da cui dipende la conclusione.

Chiamiamo *derivazioni* strutture sintattiche come questa: dato un sistema di inferenza $\mathcal{S}$, una derivazione in $\mathcal{S}$ è un albero in cui ogni nodo E che non sia una foglia è la conclusione di una regola di inferenza di $\mathcal{S}$. Diremo che la radice dell'albero è la conclusione della derivazione e che essa *dipende* dall'insieme costituito dalle foglie che non sono assiomi (le ipotesi della derivazione). Una dimostrazione è allora una derivazione particolare, la cui conclusione non dipende da alcuna ipotesi.

Con più precisione, definiamo il concetto di derivazione come segue.

Definizione 3.4.4 *Sia $\mathcal{S}$ un sistema di inferenza. Le* derivazioni *in $\mathcal{S}$ sono definite induttivamente come segue:*

1. *Ogni asserzione E è una derivazione di E che dipende dall'insieme di ipotesi $\{E\}$.*

2. *Se E è un assioma di $\mathcal{S}$, allora esso stesso è una derivazione di E che dipende dall'insieme vuoto di ipotesi.*

3. *Se per $i = 1, ..., n$, Π_i è una derivazione di E_i che dipende dall'insieme di ipotesi Γ_i e se $\dfrac{E_1 \quad \cdots \quad E_n}{E}$ è una regola di $\mathcal{S}$, allora*

$$\dfrac{\begin{matrix}(\Pi_1) & & (\Pi_n)\\ E_1 & \cdots & E_n\end{matrix}}{E}$$

è una derivazione di E che dipende da $\Gamma_1 \cup \ldots \cup \Gamma_n$.

Una dimostrazione *di E in $\mathcal{S}$ è una derivazione di E che dipende dall'insieme vuoto di ipotesi.*

Vi sono anche sistemi di inferenza senza nessun assioma, in cui dunque l'unico modo di iniziare una dimostrazione è mediante l'assunzione di ipotesi. Normalmente, tali sistemi dispongono anche di regole che consentono di "scaricare ipotesi": la conclusione di tali regole può non dipendere più da qualche ipotesi da cui invece dipendono le premesse.

Un esempio di regola che consente di scaricare ipotesi è la regola che rappresenta il *ragionamento per assurdo*. Un ragionamento per assurdo ha la forma seguente:

> Assumiamo che A sia falso. Quindi ... e ne segue un assurdo. Di conseguenza A è vero.

Il ragionamento inizia con l'assunzione dell'ipotesi $\neg A$. Ma la conclusione non dipende più da tale ipotesi: abbiamo dimostrato che A è vero, non che "se A è falso allora A è vero".

Le derivazioni sono anch'esse dati di un tipo induttivo. Abbiamo dunque per esse un corrispondente principio di induzione che consente di dimostrare proprietà delle derivazioni:

> Se P è una proprietà delle derivazioni di un sistema di inferenza tale che:
>
> **(B)** P vale per ogni derivazione costituita da un solo nodo, cioè un assioma o un'ipotesi, e
>
> **(PI)** per ogni derivazione Π della forma
>
> $$(\mathcal{J})\ \frac{\Pi_1 \quad \cdots \quad \Pi_n}{E}$$
>
> dove $\mathcal{J}$ è una regola d'inferenza del sistema, se P vale per Π_1 e ... P vale per Π_n, allora P vale anche per Π,
>
> allora P vale per ogni derivazione del sistema di inferenza.

3.4.3 Il sistema di inferenza Hilbertiano

Una logica è caratterizzata da tre componenti:

1. una sintassi, che definisce il linguaggio;
2. una semantica, che definisce il significato delle espressioni del linguaggio;
3. un sistema di inferenza.

Per la logica dei predicati abbiamo già considerato sintassi e semantica nei paragrafi precedenti. Un sistema di inferenza adeguato per una logica deve essere tale che, comunque si scelga un insieme di formule S e una formula A, A è derivabile da S se e soltanto se A è una conseguenza logica di S. In altri termini, il sistema di inferenza deve servire a caratterizzare la nozione semantica di conseguenza logica mediante la nozione di derivabilità. La derivabilità di una formula da un insieme di altre formule viene stabilita mediante manipolazioni sintattiche, sotto la guida delle regole di inferenza del sistema. Essa dunque non richiede alcuna comprensione del significato delle formule.

Esistono diversi calcoli che caratterizzano la nozione di conseguenza logica. Un esempio (semplice da raccontare, ma difficile da usare) è il sistema di inferenza di Hilbert. Questo è caratterizzato da un insieme di assiomi e da un insieme di regole di inferenza. Gli assiomi sono formule valide che possono essere assunte in ogni punto di una derivazione, senza che la conclusione dipenda da essi. Il linguaggio del sistema Hilbertiano contiene soltanto i connettivi $\rightarrow$ e $\neg$ e il quantificatore universale. Gli altri connettivi e $\exists$ sono simboli definiti. Il sistema Hilbertiano, che chiamiamo H, è caratterizzato da:

- Un linguaggio del primo ordine con $\rightarrow$, $\neg$ e $\forall$
- I seguenti *assiomi logici*:[3]
  1. $A \rightarrow (B \rightarrow A)$
  2. $(A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C))$
  3. $(\neg B \rightarrow \neg A) \rightarrow ((\neg B \rightarrow A) \rightarrow B)$
  4. $\forall x A \rightarrow A[t/x]$ se t è sostituibile per x in A
  5. $\forall x(A \rightarrow B) \rightarrow (A \rightarrow \forall x B)$ se A non contiene x libera
- Le seguenti regole di inferenza:

$$\frac{A \quad A \rightarrow B}{B}(\text{MPP}) \qquad\qquad \frac{A}{\forall x A}(\text{Generalizzazione})$$

Il sistema H è chiamato il *Calcolo dei Predicati del Primo Ordine*.

Nel caso del sistema Hilbertiano, in cui nessuna regola consente di scaricare ipotesi, possiamo definire il concetto di derivazione di una formula A da un insieme di formule S come una sequenza $A_1, ..., A_n$ di formule in cui $A_n = A$ e per ogni i, $0 \leq i \leq n$:

- A_i è un assioma, oppure
- $A_i \in S$, oppure
- A_i deriva da formule precedenti per mezzo di una regola di inferenza

Indichiamo con $S \vdash_H A$ la nozione di derivabilità di A da S nel sistema Hilbertiano.

Il calcolo Hilbertiano è corretto e completo:

Correttezza: se $S \vdash_H A$ e se $\mathcal{M} \models S$, allora $\mathcal{M} \models A$. In particolare, se S è un insieme di formule chiuse, e se $S \vdash_H A$, allora $S \models A$. Infatti:

- tutti gli assiomi logici sono validi;
- le regole di inferenza sono corrette, nel senso che conservano la verità:
  - se $\mathcal{M} \models A$ e $\mathcal{M} \models A \rightarrow B$, allora $\mathcal{M} \models B$;
  - se $\mathcal{M} \models A$ allora $\mathcal{M} \models \forall x A$.

Attenzione: la regola di generalizzazione conserva la verità, ma non conserva la relazione di soddisfacimento: non è sempre vero che se $(\mathcal{M}, s) \models A$, allora $(\mathcal{M}, s) \models \forall x A$.

Completezza: se $S \models A$ allora $S \vdash_H A$

[3] In realtà si tratta di *schemi d'assioma*: A, B e C sono *metaformule*, che possono essere sostituite da formule qualsiasi per ottenere assiomi. Ad esempio, se p e q sono atomi del linguaggio, allora la formula $(p \vee \neg q) \rightarrow (\neg p \rightarrow (p \vee \neg q))$ è un esempio del primo schema d'assioma, ed è dunque un assioma del sistema Hilbertiano per il linguaggio considerato.

3.4.4 Teorie del primo ordine

Una teoria del primo ordine è costituita da:

1. un linguaggio del primo ordine;
2. un sistema di inferenza, costituito da:
   (a) un insieme di assiomi logici;
   (b) un insieme di regole di inferenza;
   (c) un insieme di *assiomi propri* (o non logici), che descrivono il dominio di interesse.

Una teoria è dunque caratterizzata essenzialmente dall'insieme dei suoi assiomi propri, in quanto gli assiomi logici e le regole di inferenza possono essere equivalentemente scelti in base a un qualsiasi calcolo corretto e completo per la logica dei predicati. Ecco alcuni esempi di teorie, in cui indichiamo con $\mathcal{P}$ l'insieme dei simboli di predicato del linguaggio, con $\mathcal{C}$ l'insieme delle costanti e con $\mathcal{F}$ l'insieme dei simboli funzionali:

Teoria dei grafi non orientati. $\mathcal{P} = \{R^2\}, \quad \mathcal{F} = \mathcal{C} = \emptyset.$
Assioma proprio: $\forall x \forall y (R(x,y) \rightarrow R(y,x))$

Teoria degli ordini parziali. $\mathcal{P} = \{<\}, \quad \mathcal{F} = \mathcal{C} = \emptyset.$

Assiomi propri: $\forall x \neg(x < x)$
$\forall x \forall y \forall z (x < y \wedge y < z \rightarrow x < z)$

Teorie con uguaglianza. Si tratta di teorie i cui simboli di predicato includono l'uguaglianza (=) e gli assiomi propri includono quelli specifici per essa:

$\forall x (x = x)$ (riflessivita')
$\forall x \forall y (x = y \rightarrow (A \rightarrow A'))$
dove A' si ottiene da A sostituendo alcune (non necessariamente tutte) occorrenze libere di x con y.

Da tali assiomi sono derivabili:
$\forall x \forall y (x = y \rightarrow y = x)$
$\forall x \forall y \forall z (x = y \wedge y = z \rightarrow x = z)$

Teoria dei numeri. $\mathcal{P} = \{=\}, \quad \mathcal{F} = \{succ, +, \times\}, \quad \mathcal{C} = \{0\}$
Gli assiomi propri includono gli assiomi per l'uguaglianza e i seguenti *assiomi di Peano*:[4]
$\forall x \neg(0 = succ(x))$
$\forall x \forall y (succ(x) = succ(y) \rightarrow x = y)$
$\forall x (x + 0 = x)$
$\forall x \forall y (x + succ(y) = succ(x + y))$
$\forall x (x \times 0 = 0)$
$\forall x \forall y (x \times succ(y) = x + (x \times y))$
$A[0/x] \wedge \forall x (A \rightarrow A[succ(x)/x]) \rightarrow \forall x A$
L'ultima espressione è uno schema d'assioma, che formalizza il principio di induzione matematica.

[4] La teoria dei numeri qui presentata corrisponde alla una sistemazione assiomatica dell'aritmetica dei numeri naturali proposta per la prima volta, nel 1889, dal matematico italiano Giuseppe Peano.

3.5 Proprietà della logica dei predicati

3.5.1 Problemi decidibili, indecidibili, semidecidibili

Abbiamo visto nel Capitolo 2 che il problema di determinare se una formula A della logica proposizionale sia valida oppure no può essere risolto mediante un procedimento meccanico: infatti, il numero di interpretazioni di una data formula è finito, e si può dunque controllare la validità di una formula esaminando ad una ad una tutte le sue interpretazioni (questo è, sostanzialmente, il metodo delle tavole di verità). Ci chiediamo ora se lo stesso problema può ancora essere risolto in modo automatico in logica dei predicati.

Utilizzeremo qui la nozione di "procedimento automatico" affidandoci alla comprensione intuitiva che di essa possiamo avere. In realtà tale nozione richiederebbe uno studio molto più approfondito (si vedano ad esempio [1, 4, 8], o qualsiasi testo classico di logica come [5, 9]). Negli anni 30 i logici hanno infatti proposto diverse caratterizzazioni formali della "calcolabilità", o della nozione di *funzione calcolabile* (mediante un procedimento automatico). I sistemi introdotti per definire in modo rigoroso l'insieme delle funzioni (numeriche) calcolabili in modo meccanico si sono rivelati tutti equivalenti, nel senso che caratterizzano lo stesso insieme di funzioni. Ciò ha portato a ipotizzare che le funzioni calcolabili siano proprio quelle catturate da tali sistemi formali. Questa ipotesi è nota come *tesi di Church* e, fino ad oggi, non ha trovato smentita.

Accenniamo qui soltanto brevemente alla caratterizzazione delle funzioni calcolabili dovuta a Gödel, quella che ricorre alle *funzioni ricorsive.* L'insieme delle funzioni ricorsive è il sottoinsieme delle funzioni sui naturali definito induttivamente come segue:

1. Le seguenti *funzioni iniziali* sono ricorsive:
   (a) $zero$, la funzione costante 0, tale che per ogni $n \in \mathbb{N}$: $zero(n) = 0$.
   (b) la funzione successore $succ$, tale che per ogni $n \in \mathbb{N}$: $succ(n) = n + 1$.
   (c) per ogni $k > 0$ e $i \in \mathbb{N}$, $1 \leq i \leq k$, le funzioni di proiezione π^i_k, tali che per ogni $n_1, ..., n_k \in \mathbb{N}$: $\pi^i_k(n_1, ..., n_k) = n_i$.
2. Se le funzioni $h, g_1, ..., g_p$ sono ricorsive, allora la funzione f tale che:
   $$f(n_1, ..., n_k) = h(g_1(n_1, ..., n_k), ..., g_p(n_1, ..., n_k))$$
   è ricorsiva (*Regola di composizione*).
3. Se h e g sono ricorsive e f è tale che:
   $$\begin{array}{rcl} f(0, n_1, ..., n_k) & = & g(n_1, ..., n_k) \\ f(m + 1, n_1, ..., n_k) & = & h(f(m, n_1, ..., n_k), m, n_1, ..., n_k) \end{array}$$
   allora f è ricorsiva (*Regola di ricorsione*).
4. Se g è ricorsiva e per ogni $n_1, ..., n_k$ esiste p tale che $g(n_1, ..., n_k, p) = 0$, allora è ricorsiva anche la funzione f tale che:
   $$f(n_1, ..., n_k) = min\{p \in \mathbb{N} \mid g(n_1, ..., n_k, p) = 0\}$$
   dove $min\ A$ è il minimo elemento dell'insieme A (*Regola di minimizzazione*).

Ad esempio, possiamo dimostrare che la somma è ricorsiva, in quanto può essere definita per ricorsione a partire da funzioni iniziali.

$$\begin{array}{rcl} sum(0, n) & = & \pi^1_1(n) \\ sum(m + 1, n) & = & succ(\pi^1_3(sum(m, n), m, n)) \end{array}$$

Si noti che ogni funzione ricorsiva è calcolabile: le funzioni iniziali sono calcolabili, la composizione di funzioni calcolabili è certamente calcolabile, la regola di ricorsione corrisponde alla definizione ricorsiva di una funzione in un linguaggio di programmazione. Infine se g è calcolabile, la funzione f definita per minimizzazione sulla base di g può essere calcolata in modo iterativo, determinando il valore di $g(n1, ..., nk, 0)$, poi quello di $g(n1, ..., nk, 1)$, ecc. fermandosi appena si trova un valore uguale a 0; se esiste p tale che $g(n_1, ..., n_k, p) = 0$ tale iterazione prima o poi termina fornendo il minimo p con tale proprietà.

Se accettiamo la tesi di Church, vale allora anche l'inverso: ogni funzione calcolabile è ricorsiva, quindi "funzione calcolabile" e "funzione ricorsiva" diventano sinonimi (estensionalmente).

Mediante la nozione di funzione ricorsiva possiamo caratterizzare in modo formale anche il concetto di problema risolubile in modo automatico. Consideriamo qui soltanto *problemi di decisione*, cioè problemi che ammettono risposte del tipo SI/NO. Un problema di decisione è un problema della forma seguente: "dato un oggetto x di un certo insieme C, determinare se x gode della proprietà P oppure no", o, in altri termini: "dato $x \in C$ e $A \subseteq C$ (A è il sottoinsieme di C contenente tutti e soli gli elementi che hanno la proprietà P), determinare se $x \in A$". Indicheremo con $\Pi_{C,A}$ un tale problema di decisione. Una *procedura di decisione* per $\Pi_{C,A}$ è un procedimento automatico che risolve $\Pi_{C,A}$, cioè che, dato un qualunque elemento x della classe C, determina se $x \in A$ oppure no. Un problema Π è *decidibile* se esiste una procedura di decisione per esso.

La nozione di decidibilità si si può rendere rigorosa riconducendo la nozione di "procedimento automatico" (o algoritmo) a quella di funzione calcolabile e sfruttando la tesi di Church.

Una procedura di decisione per un problema $\Pi_{C,A}$ equivale al calcolo della *funzione caratteristica* dell'insieme A:

Definizione 3.5.1 *Se $A \subseteq C$, la funzione caratteristica di A è la funzione f_A tale che per ogni $x \in C$:*

$$f_A(x) = \begin{cases} 1 & \textit{se } x \in A \\ 0 & \textit{altrimenti} \end{cases}$$

Un problema di decisione non numerico $\Pi_{C,A}$ si può facilmente ricondurre a un problema di decisione sui numeri naturali, cioè un problema di decisione in cui C è $\mathbb{N}$, purché sia possibile codificare gli elementi dell'insieme C mediante numeri naturali. Molti domini possono essere codificati in tal modo, come è noto in informatica: diversi tipi di oggetti possono essere rappresentati mediante sequenze di 0 e 1 di lunghezza fissata, riconducendoli così alla rappresentazione binaria di un numero.[5] Di conseguenza, considereremo soltanto problemi di decisione sui numeri naturali.

Se la funzione caratteristica dell'insieme A è calcolabile, allora il problema di decisione determinato da A è chiaramente decidibile: una procedura di decisione consiste nel calcolare il valore di $f_A(n)$; la risposta è SI se $f_A(n) = 1$, NO altrimenti.

La nozione di problema decidibile è allora riconducibile a quella di **insieme ricorsivo**:

Definizione 3.5.2 *Un sottoinsieme A di $\mathbb{N}$ è ricorsivo se la sua funzione caratteristica f_A è ricorsiva.*

Un problema di decisione $\Pi_{\mathbb{N},A}$ sui numeri naturali, consistente nel determinare se $n \in A$, è decidibile se A è un insieme ricorsivo.

[5] Attenzione: non tutti i domini possono essere codificati mediante numeri naturali. Ad esempio, i numeri reali non possono essere rappresentati dai naturali; ciò segue banalmente dalla diversa cardinalità dei due insiemi.

Un problema Π è *indecidibile* se A non è ricorsivo, cioè se la funzione caratteristica f_A non è ricorsiva. Ciò significa (se si accetta la tesi di Church) che non esiste alcun procedimento meccanico per calcolare f_A.

La formalizzazione del concetto di decidibilià ha consentito di dimostrare l'esistenza di problemi indecidibili. Si noti che un problema indecidibile non è semplicemente un problema che non si sa risolvere in modo automatico, ma è un problema per il quale è dimostrata l'impossibilità di soluzione automatica.

Consideriamo ora una proprietà più debole della decidibilità: un problema di decisione $\Pi_{C,A}$ è **semidecidibile** se esiste un procedimento automatico che, dato un qualunque elemento $x \in C$, se $x \in A$, riconosce che $x \in A$; tuttavia, nel caso in cui $x \notin A$, il procedimento può non terminare e non fornire risposta.

Per formalizzare la nozione di semidecidibilità, immaginiamo che la *procedura di semidecisione* per $\Pi_{C,A}$ operi come una macchina M che, quando viene fornita di un $x \in C$, genera ad uno ad uno gli elementi di A e per ciascuno di essi verifica se è uguale a x o no. La macchina si ferma con risposta SI quando genera x. Dunque, se $x \notin A$ possono accadere due cose: M può fermarsi e rispondere NO, se A è un insieme finito, oppure M non si ferma mai. In quest'ultimo caso, ponendosi di fronte a M, non si può sapere se M sta continuando i suoi calcoli perché ancora deve generare x, oppure perché x non verrà mai generato. In qualunque momento si decida di spegnere M e optare per la seconda soluzione, si potrebbe essere in errore: forse, qualche minuto più tardi M avrebbe generato x e risposto SI.

Quindi possiamo dire che un problema $\Pi_{C,A}$ è semidecidibile se esiste un procedimento automatico che genera ad uno ad uno gli elmenti di A. In altri termini, se A può essere *enumerato* in modo automatico: esiste cioè un metodo meccanico che genera il primo elemento di A, poi il secondo elemento di A, e così via.

Se nell'affermazione precedente sostituiamo "procedimento automatico" con "funzione calcolabile" e accettiamo la tesi di Church, otteniamo la seguente caratterizzazione formale della nozione di problema semidecidibile:

Definizione 3.5.3 *Un sottoinsieme A di $\mathbb{N}$ è* ricorsivamente enumerabile *(r.e.) se è vuoto o l'immagine di una funzione ricorsiva.*

Un problema di decisione $\Pi_{\mathbb{N},A}$ è semidecidibile *se A è r.e.*

3.5.2 Semidecidibilità, compattezza, indecidibilità

Consideriamo ora il problema di decisione "A è una conseguenza logica di S?" in una logica qualsiasi.

In generale, se $\mathcal{I}$ è un sistema di inferenza, in cui la proprietà di essere un assioma di $\mathcal{I}$ e la relazione di derivabilità mediante ciascuna regola di inferenza di $\mathcal{I}$ sono nozioni decidibili, e se il linguaggio è numerabile, allora la nozione di derivabilità in $\mathcal{I}$ è semidecidibile, cioè la relazione $S \vdash_{\mathcal{I}} A$ è semidecidibile. Infatti, una banale procedura di semidecisione per $S \vdash_{\mathcal{I}} A$ può essere la seguente: se il linguaggio è numerabile, possiamo enumerare tutte le sequenze finite di simboli del linguaggio e, per ciascuna di esse, determinare se costituisce una derivazione di A da S nel sistema considerato (esiste una procedura di decisione per fare questo, se la proprietà di essere un assioma e la relazione di derivabilità mediante ciascuna regola di inferenza sono decidibili). Se $S \vdash_{\mathcal{I}} A$, prima o poi si troverà una derivazione di A da S e il procedimento termina. Se invece A non è derivabile da S il procedimento non termina.

In particolare, dunque, la nozione di derivabilità nel sistema Hilbertiano è semidecidibile: $S \vdash_H A$ è un problema semidecidibile. Poiché il calcolo è corretto e completo, è semidecidibile anche il problema di determinare se una formula A è una conseguenza logica di un insieme S di formule in logica dei predicati. In particolare l'insieme delle formule valide è ricorsivamente enumerabile. Diciamo allora che *il calcolo dei predicati è semidecidibile.*

Sebbene il calcolo dei predicati sia semidecidibile, esso è indecidibile: l'insieme delle formule valide non è ricorsivo.

In particolare, se $\mathcal{L}$ è un linguaggio con almeno una costante e un simbolo di predicato binario, allora l'insieme delle formule valide di $\mathcal{L}$ non è ricorsivo.

Ci sono tuttavia alcuni casi interessanti di decidibilità:

- calcolo monadico (i predicati sono tutti a un solo argomento);
- formule puramente esistenziali, cioè della forma $\exists x_1 ... \exists x_n A$ dove A è senza quantificatori né simboli funzionali;
- formule puramente universali, cioè della forma $\forall x_1 ... \forall x_n A$ dove A è senza quantificatori né simboli funzionali.

Dal fatto che la nozione di conseguenza logica per la logica dei predicati sia caratterizzabile mediante un sistema di inferenza quale H, seguono altre importanti proprietà, quali la *compattezza* della logica dei predicati del primo ordine:

Compattezza: Se $S \models A$ allora esiste S_0 finito, $S_0 \subseteq S$, tale che $S_0 \models A$.

Infatti, se $S \models A$, allora $S \vdash_H A$ (per la completezza del sistema deduttivo). Poiché una derivazione è un oggetto finito, la derivazione di A da S può utilizzare soltanto un sottoinsieme finito S_0 di S, quindi $S_0 \vdash_H A$ per qualche $S_0 \subseteq S$ finito. Dalla correttezza del sistema deduttivo segue $S_0 \models A$.

La compattezza si può equivalentemente formulare come segue:

Se ogni sottoinsieme finito di S è soddisfacibile, allora S è soddisfacibile.

Dimostriamo l'equivalenza delle due formulazioni per contrapposizione: se S è insoddisfacibile, allora $S \models \bot$. Ciò implica che $S_0 \models \bot$ per qualche sottoinsieme finito $S_0 \subseteq S$ (per la prima formulazione della compattezza), quindi S_0 è insoddisfacibile. Viceversa, se non esiste alcun sottoinsieme finito S_0 di S tale che $S_0 \models A$, allora non esiste alcun sottoinsieme finito S_0 di S tale che $S_0 \cup \{\neg A\}$ è insoddisfacibile, cioè ogni sottoinsieme finito di $S \cup \{\neg A\}$ è soddisfacibile. Dunque (per la seconda formulazione) $S \cup \{\neg A\}$ è soddisfacibile, cioè $S \not\models A$.

Si noti che esistono logiche non compatte. Un esempio è dato dalla "teoria dell'aritmetica del secondo ordine", PA, per la quale vale la seguente proprietà:

$$PA \models A \text{ sse } A \text{ è vera nell'interpretazione standard su } \mathbb{N}$$

Si ha dunque in particolare che per qualsiasi proprietà dei naturali rappresentata da una formula $F(x)$:

$$PA,\ F(0), F(1), F(2), ... \models \forall x F(x)$$

Tuttavia $\forall x F(x)$ non è conseguenza logica di alcun sottoinsieme finito di $PA \cup \{F(0), F(1), F(2), ...\}$.

Un'ulteriore proprietà della logica dei predicati che segue dall'esistenza di sistemi di inferenza corretti e completi per essa è la *monotonicità*: se $S \subseteq S'$ e $S \models A$, allora $S' \models A$. Si noti che il ragionamento di senso comune spesso non è monotono. Ad esempio, se sappiamo che Pippo è un elefante, deduciamo dalle nostre conoscenze sugli elefanti che Pippo è grigio. Ma se alla nostra conoscenza aggiungiamo il fatto che Pippo è albino, non concludiamo più che Pippo è grigio.

3.5.3 Come si dimostra l'indecidibilità della logica dei predicati

In questo paragrafo viene data una presentazione intuitiva del modo in cui si dimostra che la logica dei predicati è indecidibile.

Riformulando in termini semplici la definizione 3.5.2, possiamo dire che, data una proprietà P, il problema $P(x)$ è decidibile se esiste un procedimento effettivo (algoritmo, programma) che, dato qualunque input x, termina con output `true` se vale $P(x)$ e termina con output `false` se non vale $P(x)$.

Una logica è decidibile se il problema di determinare se $S \models A$ è decidibile, cioè se esiste un algoritmo che, dato qualunque insieme S di formule e una formula A, termina con output `true` se $S \models A$ e termina con output `false` se $S \not\models A$.

L'indecidibilità della logica dei predicati si può dimostrare mostrando come ridurre a un problema di conseguenza logica un problema la cui indecidibilità è nota, ad esempio il

Problema della fermata: dato un programma p e un suo input x, l'esecuzione di p con input x termina?

Indichiamo con $p(x)$ l'esecuzione di p con input x.

Il problema della fermata è indecidibile: infatti,

supponiamo per assurdo

che esista un algoritmo (programma) `M` che, dato un qualsiasi programma p e un suo input x, termina con output `true` se $p(x)$ si ferma e termina con output `false` se $p(x)$ non si ferma:

`M(`p, x`) = true` se l'esecuzione di p con input x termina, `false` altrimenti.

Possiamo allora scrivere un programma `K` con un unico input:

`K(`p`) = M(`p, p`)`

Quindi `K(`p`) = true` se l'esecuzione di p con input p termina, `false` altrimenti.

Sia ora `loop` un qualsiasi programma che non termina mai (e.g. `while true do done`), e consideriamo il seguente programma `N` con un unico input:

`N(`p`) = if K(`p`) then loop else false`

Questo programma è tale che, per ogni programma `P`, se l'esecuzione di `P` con input `P` termina (`K(P)=true`), allora `N(P)` non si ferma, altrimenti `N(P)` si ferma.

Cosa succede quando il programma `N` viene eseguito con input `N` stesso?

```
N(N)  = if K(N) then loop else false
      = if N con input N si ferma then loop else false
```

Quindi se `N(N)` si ferma (`K(N)` riporta `true`), allora `N(N)` non si ferma e se `N(N)` non si ferma (`K(N)` riporta `false`), allora `N(N)` si ferma.

Contraddizione

Quindi è assurda l'ipotesi iniziale che esista un algoritmo `M` che, dato un qualsiasi programma p e un suo input x, termina con output `true` se $p(x)$ si ferma e termina con output `false` se $p(x)$ non si ferma.

Riduzione del problema della fermata a un problema di conseguenza logica (sketch): si definisce un procedimento (automatico) che, dato qualsiasi programma p e input x, costruisce un insieme di formule $S_{p,x}$ e una formula $A_{p,x}$ tali che:

$S_{p,x} \models A_{p,x}$ se e solo se l'esecuzione di p con input x termina.

Se il problema di determinare se $S \models A$ fosse decidibile, sarebbe allora decidibile anche il problema della fermata: dati p e x, si costruiscono i corrispondenti $S_{p,x}$ e $A_{p,x}$ e si decide se $S_{p,x} \models A_{p,x}$.

Quindi la logica dei predicati è indecidibile.

3.6 Esercizi

A. Definire (ricorsivamente sui termini) l'insieme $var(t)$ delle variabili che occorrono in un termine t.

B. Per ciascuna delle formule a pagina 32, dimostrare che esse sono effettivamente formule, utilizzando la definizione ricorsiva 3.2.2, e la definizione di termine.

C. Definire ricorsivamente (sulle formule) l'insieme delle variabili libere in una formula.

D. Determinare l'interpretazione della formula

$$\forall x(padre(x) = felix \vee padre(x) = padre(giovanni))$$

nell'interpretazione data a pagina 39.

E. Sia A la formula $\forall x\,(p(x) \to \exists y\,(q(y) \wedge r(x, y)))$. Definire un'interpretazione $\mathcal{M}$ di A in cui A è vera e dimostrare che $\mathcal{M} \models A$, e un'interpretazione $\mathcal{M}'$ di A in cui A è falsa e dimostrare che $\mathcal{M}' \not\models A$.

F. Si considerino gli enunciati seguenti:

(A) $\forall x\,(paese(x) \to \exists y\,(re(y) \wedge governa(y, x)))$;

(B) $\exists y\,(re(y) \wedge \forall x\,(paese(x) \to governa(y, x)))$.

(1) Dare una lettura di A e B in linguaggio naturale ed illustrare le differenze tra le due formule.

(2) È vero o falso che $A \models B$? Dimostrare la propria affermazione.

(3) È vero o falso che $B \models A$? Dimostrare la propria affermazione.

G. Per ciascuno dei seguenti enunciati: a) determinare un linguaggio per esprimerlo, b) scrivere una formula che lo rappresenti, c) determinare un modello e un contromodello.

(1) Giovanni è fratello di Riccardo.

(2) Il re d'Inghilterra è cattivo.

(3) Il padre di Riccardo è sposato con la madre di Giovanni.

(4) Giovanni ha governato l'Inghilterra nel 1200.

(5) Giovanni è fratello di Riccardo se Riccardo è fratello di Giovanni.

(6) Giovanni non è fratello di Robin Hood.

(7) Se la madre di Giovanni è diversa dalla madre di Robin Hood e il padre di Giovanni è diverso dal padre di Robin Hood, allora Giovanni e Robin Hood non sono fratelli.

(8) Se x è positivo, allora anche x^2 è positivo.

(9) Condizione sufficiente affinché x sia dispari è che x sia primo e diverso da 2.

(10) Condizione necessaria affinché una successione s sia convergente è che s sia limitata.

(11) Tutte le balene sono mammiferi.

(12) Tutti gli inglesi amano la regina d'Inghilterra.

(13) Qualche inglese ama il Re d'Inghilterra.

(14) Can che abbaia non morde.

(15) Alcuni sciatori odiano la montagna.
(16) Riccardo ha un fratello che è matto.
(17) Nessun uomo è sia dottore sia pescivendolo.
(18) Qualche mandriano non è abbronzato.
(19) Chi guida un aereo è un aviatore.
(20) Ogni aereo è guidato da qualche aviatore.
(21) Esiste un aviatore che guida tutti gli aerei.
(22) Ognuno ha un genitore.
(23) Qualcuno è genitore di tutti.
(24) Qualcuno ha come genitori tutti.
(25) Qualche inglese ama i re di tutti i paesi.
(26) Ognuno ama qualcuno e nessuno ama tutti, oppure qualcuno ama chiunque e qualcuno non ama nessuno.
(27) Ogni numero ha un successore.
(28) Ogni numero diverso da 0 è il successore di qualche numero.
(29) Nessun numero è il successore di tutti i numeri.
(30) Ogni ragazzo ama una certa ragazza.
(31) A tutti i cani, eccetto i chihuahua, piace il freddo.
(32) Se due insiemi hanno gli stessi elementi, allora sono uguali.
(33) Ogni oggetto appartiene a un insieme.
(34) Giovanni non è fratello di nessun fratello di Robin Hood.
(35) Ognuno ha un padre e una madre.
(36) Nessuno può vincere tutti.
(37) Qualcuno non vince nessuno.
(38) Qualcuno è migliore di tutti.
(39) Tutti i tedeschi parlano la stessa lingua.
(40) Giovanni non ama ogni ragazza.
(41) Giovanni è più giovane di un suo fratello.
(42) Giovanni e Robin Hood sono fratelli se hanno un genitore in comune.
(43) Solo gli uomini che mangiano legumi sono vegetariani.
(44) Se è vero che ogni avo di un avo di un individuo è anche avo dello stesso individuo, allora deve esistere una persona che non ha avi.
(45) Tutte le città in cui mi sono fermato avevano un'ottima pinacoteca.
(46) Mi sono fermato in una città che aveva un'ottima pinacoteca.
(47) Se a è divisibile per b e b è divisibile per c, allora a è divisibile per c.
(48) Esiste un barbiere a Marsiglia che rade tutte e solo le persone che non si radono da sole.
(49) A marsiglia c'è un barbiere che rade solo quelli che non si radono da soli
(50) Non solo i liberi professionisti possono evadere le tasse
(51) Solo Antonio non evade le tasse
(52) Ad Antonio piacciono solo i film di Fellini

(53) L'unico film che piace a Antonio è Il Corvo

H. Dimostrare le proprietà enunciate a pagina 38.

I. Dimostrare che una formula A è valida sse $\neg A$ non è soddisfacibile.

J. Dimostrare il Lemma 3.3.9 e completare la dimostrazione del Lemma 3.3.10.

K. Dimostrare che se t è un termine chiuso, allora per ogni assegnazione s e s', $s(t) = s'(t)$.

L. Dimostrare la validità delle formule a pagina 42.

M. Completare le dimostrazioni della non validità delle formule a pagina 43.

N. Rappresentare i seguenti enunciati in un linguaggio del primo ordine e determinare se sono validi oppure no.

(1) Se qualcuno beve, allora tutti bevono.
(2) Esiste qualcuno tale che se lui beve, allora tutti bevono.
(3) Esiste qualcuno tale che, quando lui beve, tutti bevono.
(4) Se esiste un numero maggiore di tutti gli altri, allora per ogni numero ne esiste uno maggiore.
(5) Se per ogni numero ne esiste uno minore, allora esiste un numero minore di tutti.
(6) Se tutte le città in cui mi sono fermato avevano un'ottima pinacoteca, allora almeno una città ha un'ottima pinacoteca.
(7) Se tutti sono aggressivi con qualcuno, allora qualcuno è aggressivo con tutti.

O. Le seguenti argomentazioni sono corrette?

(1) Tutti i ladri sono ricchi. B è ricco. Quindi B è un ladro.
(2) Non esiste nessun numero naturale minore di zero. Quindi tutti i numeri naturali minori di zero sono dispari.
(3) Alcuni sciatori odiano la montagna. Alberto Tomba è uno sciatore. Quindi Alberto Tomba odia la montagna.
(4) Jacques è avaro. Nessun francese è avaro. Quindi Jacques non è francese.
(5) Tutti i francesi eccetto i parigini sono gentili. Jacques è parigino. Quindi Jacques non è gentile oppure non è francese.
(6) Tutti i francesi eccetto i parigini sono gentili. Jacques è francese e non è gentile. Quindi Jacques è parigino.
(7) Guglielmo non è francese. Tutti i francesi sono avari. Quindi Guglielmo non è avaro.
(8) Guglielmo non è francese. Solo i francesi sono avari. Quindi Guglielmo non è avaro.
(9) Nessuna montagna è raggiungibile. Alcune colline sono montagne. Quindi alcune colline non sono raggiungibili.
(10) Ogni uomo sano di mente può capire la matematica. Nessuno dei figli di Hegel capisce la matematica. Nessun pazzo può votare. Quindi nessun figlio di Hegel può votare.
(11) Tutti i portoghesi sono socievoli. Qualche lattaio è portoghese. Quindi qualche lattaio è socievole.

(12) Se ogni paese verde confina solo con paesi rossi e ogni paese rosso confina solo con paesi blu e ogni paese confina con qualche paese, allora nessun paese è verde.

(13) Per ogni insieme X esiste un insieme Y di cardinalità maggiore di quella si X. Se X è incluso in Y, la cardinalità di X non è maggiore di quella di Y. Ogni insieme è incluso in V. Quindi V non è un insieme.

(14) Tutti i cavalli sono animali, quindi tutte le teste di cavallo sono teste di animali.

(15) Alcuni ragazzi amano tutte le ragazze. Nessun ragazzo ama un topo di biblioteca. Quindi nessuna ragazza è un topo di biblioteca.

(16) Alcuni botanici sono eccentrici. Alcuni botanici non amano cose eccentriche. Quindi alcuni botanici non sono amati da tutti i botanici.

(17) Se uno parla con qualcuno allora vi è qualcun altro che li ha presentati. Nessuno presenta uno a qualcuno se non conosce entrambi. Tutti parlano con Mario. Quindi tutti sono stati presentati a Mario da qualcun altro che lo conosce.

(18) Se piove nessun uccello è felice. Se nevica alcuni uccelli sono felici. Quindi se piove non nevica.

(19) Tutti i cammelli preferiscono un bravo cammelliere. Alcuni cammelli non preferiscono Mohammed. Quindi Mohammed non è bravo.

(20) Tutti i cammelli preferiscono un bravo cammelliere. Alcuni cammelli non preferiscono Mohammed. Quindi Mohammed non è bravo oppure non è un cammelliere.

(21) Ogni barbiere di Marsiglia rade esattamente quelle persone che non si radono da sole. Quindi non esiste alcun barbiere a Marsiglia.

P. Esprimere in un linguaggio con eguaglianza:

(1) esiste un unico gatto

(2) esistono esattamente due gatti

(3) esistono esattamente n gatti

(4) esistono almeno due gatti

(5) esistono almeno n gatti

Q. Formalizzare il seguente ragionamento e dire che cosa c'è di sbagliato:

> Per ogni numero naturale, c'è qualche numero più grande. Pertanto, sia x un numero arbitrario: x deve essere inferiore a qualche altro numero. Sia y un tale numero. Si ha $x < y$. Poiché x è arbitrario, si ha che ogni numero è più piccolo di y. Perciò esiste un numero più grande di tutti.

R. Consideriamo in questo esercizio il **Paradosso di Russell**. L'assioma di comprensione (non ristretto), inizialmente proposto da Frege per formalizzare la teoria degli insiemi, afferma che per ogni proprietà o concetto P esiste l'insieme K di tutti gli oggetti che godono di P. Utilizzando l'abituale notazione insiemistica, si può formulare come segue: per ogni proprietà P:

$$\exists k \forall y (y \in k \equiv P(y))$$

Consideriamo ora la proprietà $P(y) =_{def} \neg(y \in y)$. Per l'assioma di comprensione si avrebbe:

$$\exists k \forall y (y \in k \equiv \neg(y \in y)) \tag{3.8}$$

Dimostrare che sia $k \in k$ che $\neg(k \in k)$ sono conseguenze logiche di $\forall y(y \in k \equiv \neg(y \in y))$, quindi che l'assioma di comprensione è contraddittorio.

3.7 Soluzione di alcuni esercizi

(E) La formula è un'affermazione universale: il modo più semplice di renderla vera è interpretare p con l'insieme vuoto. Quindi è sufficiente stabilire che $\mathcal{M}(p) = \emptyset$ (dimostrare in dettaglio che tale interpretazione è effettivamente un modello della formula data, utilizzando la definizione 3.3.5; per verificare la correttezza dei passaggi si utilizzi il programma `models` che si può scaricare da [7].

Per definire un contromodello occorre avere almeno un oggetto che soddisfa l'antecedente, sebbene il conseguente sia falso. Il modo più semplice di falsificare il conseguente è quello di rendere q sempre falso. Se dunque $\mathcal{M}$ ha dominio $D = \{0\}$ con un solo oggetto, $\mathcal{M}(p) = \{0\}$ e $\mathcal{M}(q) = \emptyset$, allora $\mathcal{M} \not\models A$ (sviluppare in dettaglio i passaggi della dimostrazione, utilizzando la definizione di $\not\models$ derivata dalla definizione 3.3.5; verificare i dettagli utilizzando il programma `models` [7].

(F) La formula A si può leggere "ogni paese ha un re che lo governa"; la B "esiste un re che governa tutti i paesi". Le differenze fondamentali sono due: (a) in A il re dipende dal paese, quindi due paesi possono avere re differenti, mentre la B richiede che il re sia lo stesso per tutti i paesi; (b) nel caso in cui non esista alcun paese, la (a) è vera anche se non esiste nemmeno alcun re, mentre la (b) richiede comunque l'esistenza di un re.

$A \not\models B$: per dimostrarlo occorre costruire un modello di A che non sia un modello di B. Alla luce di quanto detto prima, possiamo costruire una tale interpretazione stabilendo che $\mathcal{M}(paese) = \emptyset$ e $\mathcal{M}(re) = \emptyset$. Oppure, possiamo stabilire che il dominio sia $D = \{0, 1\}$, che $\mathcal{M}(paese) = \mathcal{M}(re) = D$ e $\mathcal{M}(governa) = \{\langle 0, 0\rangle, \langle 1, 1\rangle$ (sviluppare in dettaglio i passaggi della dimostrazione del fatto che $\mathcal{M} \models A$ e $\mathcal{M} \not\models B$, in entrambi i casi; verificare i passaggi utilizzando il programma `models` [7]).

Per dimostrare che $B \models A$, occorre fare un ragionamento generale. Assumiamo dunque che $\mathcal{M}$ sia qualsiasi interpretazione tale che $\mathcal{M} \models B$, quindi esiste $d_0 \in D$ tale che:

(a) $d_0 \in \mathcal{M}(re)$ e

(b) per ogni $d \in D$, o $d \notin \mathcal{M}(paese)$, oppure $\langle d_0, d\rangle \in \mathcal{M}(governa)$

(sviluppare in dettaglio i passaggi che portano a questa affermazione). Supponiamo ora per assurdo che $\mathcal{M} \not\models A$, quindi esiste d_1 tale che:

(c) $d_1 \in \mathcal{M}(paese)$ e

(d) per ogni $d \in D$, o $d \notin \mathcal{M}(re)$, oppure $\langle d, d_1\rangle \notin \mathcal{M}(governa)$

(sviluppare in dettaglio i passaggi che portano a questa affermazione). La (d) vale in particolare anche per $d = d_0$, quindi $d_0 \notin \mathcal{M}(re)$, oppure $\langle d_0, d_1\rangle \notin \mathcal{M}(governa)$. Ma per la (a) la prima alternativa è falsa, dunque:

(e) $\langle d_0, d_1\rangle \notin \mathcal{M}(governa)$.

Analogamente, la (b) vale in particolare anche quando $d = d_1$, quindi $d_1 \notin \mathcal{M}(paese)$, oppure $\langle d_0, d_1\rangle \in \mathcal{M}(governa)$. Per la (c) la prima alternativa è falsa, dunque:

(f) $\langle d_0, d_1\rangle \in \mathcal{M}(governa)$,

contraddicendo la (e). Quindi $B \models A$.

(G) Nella soluzione degli esercizi di questo gruppo indichiamo soltanto la rappresentazione degli enunciati e non la costruzione di modello e contromodello. Per verificare la correttezza da voi proposta per quel che riguarda modello e contromodello, quando questi hanno dominio finito, si può utilizzare il programma `models` [7].

Inoltre, si utilizzeranno nomi significativi per costanti, simboli funzionali e predicati, lasciando spesso sottinteso il loro significato. Si noti che ciascun esercizio potrebbe essere risolto in modi diversi da quello qui indicato, per diverse ragioni. Innanzitutto il linguaggio naturale è spesso ambiguo. In secondo luogo la scelta del linguaggio non è univoca: uno stesso enunciato può essere analizzato a diversi livelli di dettaglio. Ad esempio, non c'è niente di sbagliato, in sé, a rappresentare "il papà di Gennaro batte a scopone i papà di tutti i bambini del quartiere di Santa Lucia" come una proprietà di Gennaro: $p(gennaro)$. Il giusto livello di dettaglio dipende dall'uso che si vuole fare della formalizzazione, e questo, in questa classe di esercizi, non è indicato. Adotteremo spesso comunque nel seguito ipotesi semplificative sul dominio, assumendo, quando è possibile, che esso consista soltanto di elementi di un certo insieme, evitando così di far uso di predicati che identificano un certo tipo di oggetti. Infine, si deve tener conto del fatto che un enunciato può essere rappresentato da formule sintatticamente diverse ma semanticamente equivalenti. Ad esempio "nessun professore è simpatico" può essere rappresentato indifferentemente da $\neg\exists x(prof(x) \wedge simpatico(x))$, $\forall x(prof(x) \rightarrow \neg simpatico(x))$ o $\forall x(simpatico(x) \rightarrow \neg prof(x))$: le formule sono logicamente equivalenti.

(G7) Utilizziamo un linguaggio con uguaglianza contenente le costanti *giovanni* e *robin*, i simboli funzionali unari m (madre di) e p (padre di, assumiamo dunque che il dominio sia costituito da esseri umani e che tutti abbiano esattamente un padre e una madre) e il simbolo di predicato binario $fratello$.

$$\begin{array}{l}\neg(m(giovanni) = m(robin)) \wedge \neg(p(giovanni) = p(robin)) \\ \qquad \rightarrow \neg fratello(giovanni, robin)\end{array}$$

Se invece vogliamo utilizzare simboli di predicato a due posti *madre* e *padre* per la relazione di "essere madre/padre di", l'enunciato va riformulato come: se non esiste alcun x che è madre di Giovanni e madre di Robin Hood e non esiste alcun x che è padre sia di Giovanni che di Robin Hood allora Giovanni e Robin Hood non sono fratelli":

$$\begin{array}{c}\neg\exists x(madre(x, giovanni) \wedge madre(x, robin)) \\ \wedge\neg\exists x(padre(x, giovanni) \wedge padre(x, robin)) \\ \rightarrow\neg\, fratello(giovanni, robin)\end{array}$$

(G13) Se utilizziamo una costante per indicare il re d'Inghilterra (un simbolo funzionale sarebbe poco opportuno: si dovrebbe assumere che tutti gli oggetti del dominio abbiano un re):

$$\exists x(inglese(x) \wedge ama(x, re\_inghilterra))$$

Se invece vogliamo utilizzare una costante, i, per l'Inghilterra e il simbolo di predicato binario re, l'affermazione si riformula in: esiste un inglese x ed esiste un re y d'inghilterra tale che x ama y:

$$\exists x(inglese(x) \wedge \exists y(re(y, i) \wedge ama(x, y)))$$

o, equivalentemente: $\exists x \exists y(inglese(x) \wedge re(y, i) \wedge ama(x, y))$

(G14) Anche se espressa al singolare, si tratta di un'affermazione universale su tutti i cani: $\forall x(cane(x) \rightarrow \neg morde(x))$.

(G17) $\neg\exists x(dottore(x) \wedge pescivendolo(x))$.

(G19) Utilizziamo un linguaggio con i simboli di predicato av, $aereo$, $guida$ ($av(x)$: x è un aviatore, $aereo(x)$: x è un aereo, $guida(x,y)$: x guida y). Si tratta di un'affermazione universale su tutti coloro che guidano un aereo, cioè su tutti gli x tali che $\exists y(aereo(y) \wedge guida(x,y))$. Dunque:

$$\forall x(\exists y(aereo(y) \wedge guida(x,y)) \rightarrow aviatore(x))$$

Questa formula è equivalente a:

$$\forall x \forall y(aereo(y) \wedge guida(x,y) \rightarrow aviatore(x))$$

(G20) Questa è un'affermazione universale sugli aerei:

$$\forall x(aereo(x) \rightarrow \exists y(aviatore(y) \wedge guida(y,x)))$$

(G21) Questa è un'affermazione di esistenza di un aviatore con la proprietà di guidare ogni aereo:

$$\exists x(aviatore(x) \wedge \forall y(aereo(y) \rightarrow guida(x,y)))$$

(G22) Se $genitore(x,y)$ sta per "x è genitore di y": $\forall x \exists y\, genitore(y,x)$.

(G23) $\exists y \forall x\, genitore(y,x)$. Si noti l'inversione dei quantificatori rispetto alla formula precedente.

(G24) $\exists x \forall y\, genitore(y,x)$.

(G25) $\exists x(inglese(x) \wedge \forall y\,(\exists z(paese(z) \wedge re(y,z)) \rightarrow ama(x,y)))$. Si noti il quantificatore esistenziale su z: anche se l'enunciato afferma "i re di *tutti* i paesi", quel che si vuole dire è che "se y è un re (di *qualche* paese), allora x ama y". La formula è comunque equivalente a $\exists x(inglese(x) \wedge \forall y \forall z\,(paese(z) \wedge re(y,z) \rightarrow ama(x,y)))$, dove il quantificatore esistenziale è "diventato" universale.

Se scrivessimo invece $\exists x(inglese(x) \wedge \forall y\,(\forall z(paese(z) \wedge re(y,z)) \rightarrow ama(x,y)))$, ciò significherebbe "esiste un inglese che ama y se y è il re di tutti i paesi": è sufficiente che per ogni y esista un paese di cui y non è re per garantire la verità di $\forall y\,(\forall z(paese(z) \wedge re(y,z)) \rightarrow ama(x,y))$. Quindi la formula è vera in tutte le interpretazioni in cui esiste almeno un inglese e non esiste un re di tutto il mondo. Infatti la sottoformula considerata equivale a $\forall y \exists z\,(paese(z) \wedge re(y,z) \rightarrow ama(x,y))$: basta che esista un oggetto z che rende falso l'antecedente e tutta la formula è vera.

(G27) Assumendo che il dominio sia costituito soltanto da numeri: $\forall x \exists y\, succ(x) = y$.

(G28) $\forall x(\neg(x = 0) \rightarrow \exists y\, x = succ(y))$.

(G29) $\neg \exists x \forall y\, x = succ(y)$.

(G30) La frase è ambigua: se i ragazzi amano tutti la stessa ragazza, $\exists y \forall x\, ama(x,y)$, altrimenti $\forall x \exists y\, ama(x,y)$.

(G31) Possiamo dare due interpretazioni di "eccetto di": una più liberale secondo la quale l'affermazione si intende un'affermazione universale sull'insieme dei cani esclusi i chiuaua, che quindi sui chiuaua non dice nulla:

$$\forall x(cane(x) \wedge \neg chiuaua(x) \rightarrow piace\_freddo(x))$$

Ed una più forte, in cui si intende: "per ogni cane x, se x non è un chiuaua allora a x piace il freddo e, viceversa, se x è un chiuaua allora a x non piace il freddo":

$$\forall x(cane(x) \rightarrow (\neg chiuaua(x) \equiv piace\_freddo(x)))$$

(G32) Utilizziamo l'abituale notazione insiemistica per l'appartenenza. Che due insiemi x e y abbiano gli stessi elementi significa che per ogni z, $z \in x$ se e solo se $z \in y$. Dunque: $\forall x \forall y\, (\forall z(z \in x \equiv z \in y) \rightarrow x = y)$.

(G39) Attenzione: la lingua è la stessa per tutti, quindi "esiste una lingua parlata da tutti i tedeschi": $\exists x(lingua(x) \wedge \forall y(tedesco(y) \rightarrow parla(y, x)))$.

(G43) $\forall x(uomo(x) \wedge vegetariano(x) \rightarrow mangia\_legumi(x))$. L'enunciato non esclude, evidentemente, che anche un non-vegetariano mangi legumi. Se scrivessimo invece $\forall x(uomo(x) \wedge mangia\_legumi(x) \rightarrow vegetariano(x))$ significherebbe che tutti gli uomini che mangiano legumi sono vegetariani (anche quelli che mangiano i fagioli con le cotiche).

(G44) La forma è $A \rightarrow \exists x \forall y \neg avo(y, x)$, dove A rappresenta "ogni avo di un avo di un individuo è anche avo dello stesso individuo", dunque il fatto che la relazione di essere avo è transitiva: $A = \forall x \forall y \forall z(avo(x, y) \wedge avo(y, z) \rightarrow avo(x, z))$.

(G48) $\exists x(barbiere(x) \wedge \forall y(rade(x, y) \equiv \neg rade(y, y)))$. La formula non ha modelli. Essa infatti implica $\exists x(rade(x, x) \equiv \neg rade(x, x))$.

(G49) $\exists x(barbiere(x) \wedge \forall y(rade(x, y) \rightarrow \neg rade(y, y)))$. La contraddizione in questo caso non cè: semplicemente, il barbiere di Marsiglia non si rade da solo.

(G50) $\neg \forall x(evade(x) \rightarrow libero\_prof(x))$. O, equivalentemente: $\exists x(evade(x) \wedge \neg libero\_prof(x))$.

(G51) Per esprimere "solo l'oggetto X ha la proprietà P" occorre l'uguaglianza: $\forall x(\neg evade(x) \rightarrow x = antonio)$.

(G52) Se intendiamo "gli unici film che piacciono ad Antonio sono quelli di Fellini", allora: $\forall x(film(x) \wedge piace(x, antonio) \rightarrow autore(x, fellini))$. Se scriviamo invece $\forall x(piace(x, antonio) \rightarrow film(x) \wedge autore(x, fellini))$, significa che a Antonio non piace nemmeno la cioccolata. La frase di fatto può essere considerata ambigua.

(G53) $\forall x(film(x) \wedge piace(x, antonio) \rightarrow x = il\_corvo)$.

(H) (3) Assumiamo che per ogni assegnazione s: $(\mathcal{M}, s) \models A \rightarrow B$, dunque per ogni s, $(\mathcal{M}, s) \not\models A$ oppure $(\mathcal{M}, s) \models B$. Ma se è anche anche vero che, per ogni s, $(\mathcal{M}, s) \models A$, allora comunque si scelga s deve essere vero $(\mathcal{M}, s) \models B$, quindi B è vera in $\mathcal{M}$.

(5) $\mathcal{M} \models A$ se e solo se, per ogni assegnazione s, $(\mathcal{M}, s) \models A$. Ciò è vero se e solo se ogni x-variante di ogni successione s soddisfa A: per ogni s e ogni $d \in D$, $(\mathcal{M}, s[d/x]) \models A$, che è vero se e solo se, per ogni s, $(\mathcal{M}, s) \models \forall x A$, cioè $\mathcal{M} \models A$.

(L) Diamo qui di seguito soltanto alcuni esempi di dimostrazioni di validità.

- Dimostriamo che $\models \forall x A \equiv \neg \exists x \neg A$, mostrando prima che $\models \forall x A \rightarrow \neg \exists x \neg A$ e poi l'inverso. Assumiamo dunque che $(\mathcal{M}, s) \models \forall x A$, dunque:

  (a) per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \models A$.

  Assumiamo ora per assurdo che $(\mathcal{M}, s) \not\models \neg \exists x \neg A$, quindi $(\mathcal{M}, s) \models \exists x \neg A$. Da ciò segue che esiste $d_0 \in D$ tale che $(\mathcal{M}, s[d_0/x]) \models \neg A$, cioè:

  (b) $(\mathcal{M}, s[d_0/x]) \not\models A$,

  contraddicendo (a). Il ragionamento per mostrare $\models \neg \exists x \neg A \rightarrow \forall x A$ è analogo.

- Dimostriamo la validità della prima regola di distribuzione. Assumiamo dunque che $(\mathcal{M}, s) \models \forall x(A \rightarrow B)$. Da ciò segue che:

(a) per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \not\models A$ oppure $(\mathcal{M}, s[d/x]) \models B$.

Se, per assurdo, $(\mathcal{M}, s) \not\models \forall x A \to \forall x B$, allora $(\mathcal{M}, s) \models \forall x A$ e $(\mathcal{M}, s) \not\models \forall x B$, dunque:

(b) per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \models A$, e

(c) esiste $d_0 \in D$ tale che $(\mathcal{M}, s[d_0/x]) \not\models B$

La (a) vale in particolare anche per $d = d_0$, quindi $(\mathcal{M}, s[d_0/x]) \not\models A$ oppure $(\mathcal{M}, s[d_0/x]) \models B$. La prima alternativa è falsa per (b), quindi deve essere $(\mathcal{M}, s[d_0/x]) \models B$, ma questo contraddice (c).

(N2) $\exists x(\mathit{beve}(x) \to \forall y\, \mathit{beve}(y))$. La formula è equivalente a $\forall x\, \mathit{beve}(x) \to \forall y\, \mathit{beve}(y)$.

(N3) In questo caso occorre un predicato a due posti $\mathit{beve}(x, y)$: x beve al tempo y. Quindi : $\exists x \forall y\, (\mathit{beve}(x, y) \to \forall z\, \mathit{beve}(z, y))$. La formula non è valida: sia $\mathcal{M}$ con dominio $D = \{0, 1\}$ e $\mathcal{M}(\mathit{beve}) = \{\langle 0, 1\rangle, \langle 1, 0\rangle\}$. Allora:

$(\mathcal{M}, s) \models \exists x \forall y\, (\mathit{beve}(x, y) \to \forall z\, \mathit{beve}(z, y))$
sse esiste $d \in D$ tale che $(\mathcal{M}, s[d/x]) \models \forall y\, (\mathit{beve}(x, y) \to \forall z\, \mathit{beve}(z, y))$
sse esiste $d \in D$ tale che per ogni $d' \in D$:
$(\mathcal{M}, s[d/x, d'/y]) \models \mathit{beve}(x, y) \to \forall z\, \mathit{beve}(z, y)$
sse esiste $d \in D$ tale che per ogni $d' \in D$:
$(\mathcal{M}, s[d/x, d'/y]) \not\models \mathit{beve}(x, y)$ oppure
per ogni $d'' \in D$, $(\mathcal{M}, s[d/x, d'/y, d''/z]) \models \mathit{beve}(z, y)$
sse esiste $d \in D$ tale che per ogni $d' \in D$:
$\langle s[d/x, d'/y](x), s[d/x, d'/y](y)\rangle \notin \mathcal{M}(\mathit{beve})$ oppure
per ogni $d'' \in D$, $\langle s[d/x, d'/y, d''/z](z), s[d/x, d'/y, d''/z](y) \in \mathcal{M}(\mathit{beve})$
sse esiste $d \in D$ tale che per ogni $d' \in D$:
$\langle d, d'\rangle \notin \mathcal{M}(\mathit{beve})$ oppure per ogni $d'' \in D$, $\langle d'', d'\rangle \in \mathcal{M}(\mathit{beve})$

Verifichiamo se l'affermazione è vera per $d = 0$, cioè che per ogni $d' \in D$, $\langle d, d'\rangle \notin \mathcal{M}(\mathit{beve})$ oppure per ogni $d'' \in D$, $\langle d'', d'\rangle \in \mathcal{M}(\mathit{beve})$. Dobbiamo dunque verificare per i due casi $d' = 0$ (caso a) e $d' = 1$ (caso b):

(a) $\langle 0, 0\rangle \notin \mathcal{M}(\mathit{beve})$ oppure per ogni $d'' \in D$, $\langle d'', 0\rangle \in \mathcal{M}(\mathit{beve})$,

(b) $\langle 0, 1\rangle \notin \mathcal{M}(\mathit{beve})$ oppure per ogni $d'' \in D$, $\langle d'', 1\rangle \in \mathcal{M}(\mathit{beve})$.

La (a) è vera perché $\langle 0, 0\rangle \notin \mathcal{M}(\mathit{beve})$, ma la (b) è falsa perché $\langle 0, 1\rangle \in \mathcal{M}(\mathit{beve})$ e $\langle 1, 1\rangle \notin \mathcal{M}(\mathit{beve})$.

Un ragionamento analogo mostra che anche nel caso in cui $d = 1$ è falso che per ogni $d' \in D$ $\langle d, d'\rangle \notin \mathcal{M}(\mathit{beve})$ oppure per ogni $d'' \in D$, $\langle d'', d'\rangle \in \mathcal{M}(\mathit{beve})$. Quindi la formula è falsa in $\mathcal{M}$.

(N6) Utilizziamo i predicati c (essere una città), f (essermi fermato in) e p (avere un'ottima pinacoteca). Allora: $\forall x(c(x) \wedge f(x) \to p(x)) \to \exists x(c(x) \wedge p(x)))$. La formula non è valida: potrei non essermi fermato in nessuna città. Consideriamo, formalmente, l'interpretazione $\mathcal{M}$ con dominio $D = \{0\}$ e $\mathcal{M}(c) = \mathcal{M}(f) = \mathcal{M}(p) = \emptyset$. Allora: $(\mathcal{M}, s) \not\models \forall x(c(x) \wedge f(x) \to p(x)) \to \exists x(c(x) \wedge p(x)))$ sse $(\mathcal{M}, s) \models \forall x(c(x) \wedge f(x) \to p(x))$ e $(\mathcal{M}, s) \not\models \exists x(c(x) \wedge p(x))$. Verifichiamo le due condizioni:

$(\mathcal{M}, s) \models \forall x(c(x) \wedge f(x) \rightarrow p(x))$

sse	per ogni $d \in D$ (dunque solo per $d = 0$):
	$(\mathcal{M}, s[d/x]) \models c(x) \wedge f(x) \rightarrow p(x)$
sse	$(\mathcal{M}, s[0/x]) \not\models c(x) \wedge f(x)$ oppure $(\mathcal{M}, s[0/x]) \models p(x)$
che vale se:	$(\mathcal{M}, s[0/x]) \not\models c(x) \wedge f(x)$
sse	$(\mathcal{M}, s[0/x]) \not\models c(x)$ oppure $(\mathcal{M}, s[0/x]) \not\models f(x)$
che vale se:	$(\mathcal{M}, s[0/x]) \not\models c(x)$
sse	$s[0/x](x) \notin \mathcal{M}(c)$
sse	$0 \notin \mathcal{M}(c)$

Abbiamo dunque stabilito che *se* $0 \notin \mathcal{M}(c)$, allora $(\mathcal{M}, s) \models \forall x(c(x) \wedge f(x) \rightarrow p(x))$ (le implicazioni vanno dal basso verso l'alto). Poiché effettivamente $0 \notin \mathcal{M}(c)$, dato che $\mathcal{M}(c) = \emptyset$, ne segue che $(\mathcal{M}, s) \models \forall x(c(x) \wedge f(x) \rightarrow p(x))$.

$(\mathcal{M}, s) \not\models \exists x(c(x) \wedge p(x))$

sse	per ogni $d \in D$ (quindi per $d = 0$):
	$(\mathcal{M}, s[d/x]) \not\models c(x) \wedge p(x)$
sse	$(\mathcal{M}, s[0/x]) \not\models c(x)$ oppure $(\mathcal{M}, s[0/x]) \not\models p(x)$
che vale se:	$(\mathcal{M}, s[0/x]) \not\models c(x)$
sse	$s[0/x](x) \notin \mathcal{M}(c)$: vero.

(O2) Scriviamo $m(x)$ per "x è minore di 0". Il ragionamento è corretto se: $\neg\exists x\, m(x) \models \forall x(m(x) \rightarrow dispari(x))$. Verifichiamolo: siano $\mathcal{M}$ un'interpretazione e s un'assegnazione tali che:

$(\mathcal{M}, s) \models \neg\exists x\, m(x)$

sse	$(\mathcal{M}, s) \not\models \exists x\, m(x)$
sse	per ogni oggetto d del dominio D: $(\mathcal{M}, s[d/x]) \not\models m(x)$
sse	per ogni $d \in D$, $s[d/x](x) \notin \mathcal{M}(m)$
sse	(1) per ogni $d \in D$, $d \notin \mathcal{M}(m)$

Supponiamo ora, per assurdo:

$(\mathcal{M}, s) \not\models \forall x(m(x) \rightarrow dispari(x))$

sse	esiste $d \in D$ tale che $(\mathcal{M}, s[d/x]) \not\models m(x) \rightarrow dispari(x)$
sse	esiste $d \in D$ tale che $(\mathcal{M}, s[d/x]) \models m(x)$
	e $(\mathcal{M}, s[d/x]) \not\models dispari(x)$
questo implica che	esiste $d \in D$ tale che $(\mathcal{M}, s[d/x]) \models m(x)$
sse	esiste $d \in D$ tale che $s[d/x](x) \in \mathcal{M}(m)$
sse	(2) esiste $d \in D$ tale che $d \in \mathcal{M}(m)$

Ma (2) contraddice (1).

(O5) Anche qui, come indicato nella soluzione dell'esercizio G31, possiamo dare due interpretazioni di "eccetto di". Se adottiamo quella più debole, allora, intuitivamente, poiché Jacques è parigino, la prima affermazione non dice niente su di lui, quindi non si può concludere nulla sulla sua città o sulla sua gentilezza. Formalmente, il ragionamento è corretto se:

$$\forall x(f(x) \wedge \neg p(x) \rightarrow g(x)),\ p(j) \models \neg g(j) \vee \neg f(j)$$

Cerchiamo allora di costruirne un contromodello $\mathcal{M}$. Assumiamo che $\mathcal{M}(j) = d^*$. Si deve avere che per ogni assegnazione s:

1. $(\mathcal{M}, s) \models \forall x(f(x) \wedge \neg p(x) \rightarrow g(x))$, quindi che per ogni $d \in D$, o vale $d \notin \mathcal{M}(f)$ oppure $d \in \mathcal{M}(p)$ oppure $d \in \mathcal{M}(g)$ (sviluppare in dettaglio i passaggi che portano a questa conclusione).

2. $(\mathcal{M}, s) \models p(j)$, quindi che $d^* \in \mathcal{M}(p)$.
3. $(\mathcal{M}, s) \not\models \neg g(j) \vee \neg f(j)$, quindi che $d^* \in \mathcal{M}(g)$ e $d^* \in \mathcal{M}(f)$.

Costruiamo allora $\mathcal{M}$ con $D = \{d^*\}$, $\mathcal{M}(p) = \mathcal{M}(g) = \mathcal{M}(f) = \{d^*\}$. La (2) e la (3) sono senz'altro vere, E anche la (1) è vera perché $d^* \in \mathcal{M}(p)$.

Se invece intendiamo "eccetto di" nella sua accezione più forte, allora il ragionamento è corretto se e solo se:

$$\forall x(f(x) \rightarrow (\neg p(x) \rightarrow g(x))),\ p(j) \models \neg g(j) \vee \neg f(j)$$

Cerchiamo, come prima, di costruire un contromodello della conclusione, in cui le premesse siano vere. La condizione (1) diventa ora:

1. $(\mathcal{M}, s) \models \forall x(f(x) \rightarrow (\neg p(x) \equiv g(x)))$, quindi che per ogni $d \in D$, si deve avere uno dei casi seguenti:
   (a) $d \notin \mathcal{M}(f)$, oppure
   (b) $d \in \mathcal{M}(p)$ e $d \notin \mathcal{M}(g)$, oppure
   (c) $d \notin \mathcal{M}(p)$ e $d \in \mathcal{M}(g)$.

Ma le condizioni (2) e (3) contraddicono la (1), per $d = d^*$. Quindi non è possibile construire un'interpretazione con le caratteristiche volute: il ragionamento è corretto.

(O12) Assumiamo che il dominio sia costituito soltanto da paesi, e che c rappresenti la relazione "confina con". Siano:

$$\begin{array}{rcl} A & = & \forall x(verde(x) \rightarrow \forall y(c(x,y) \rightarrow rosso(y))) \\ B & = & \forall x(rosso(x) \rightarrow \forall y(c(x,y) \rightarrow blu(y))) \\ C & = & \forall x \exists y\, c(x,y) \\ D & = & \neg \exists x\, verde(x) \end{array}$$

Il ragionamento è corretto se $A, B, C \models B$. Saremmo portati a pensare che il ragionamento sia corretto in quanto, se esistesse un paese P verde, questo dovrebbe confinare con qualche paese X (per C) e X deve essere rosso per A. Ma allora X confina con P e P dovrebbe essere blu per B, mentre P è verde, dunque non può essere blu. Attenzione però, stiamo utilizzando della conoscenza aggiuntiva su colori e paesi che non è stata dichiarata: che la relazione di "confinare con" sia simmetrica e che un paese non possa essere contemporaneamente di due colori diversi.

Di fatto $A, B, C \not\models B$. Sia $\mathcal{M}$ con dominio $D = \{0\}$, $\mathcal{M}(verde) = \mathcal{M}(rosso) = \mathcal{M}(blu) = \{0\}$ e $\mathcal{M}(c) = \{\langle 0, 0\rangle\}$. Allora:

$\mathcal{M} \models A$ perché per ogni $d \in D$, $d \in \mathcal{M}(rosso)$;
$\mathcal{M} \models B$ perché per ogni $d \in D$, $d \in \mathcal{M}(blu)$;
$\mathcal{M} \models C$ perché $\langle 0, 0\rangle \in \mathcal{M}(c)$;
$\mathcal{M} \not\models D$ perché $0 \in \mathcal{M}(verde)$.

Verificare in dettaglio queste affermazioni (eventualmente utilizzando il programma `models` [7]).

(O13) È possibile formalizzare il problema utilizzando un linguaggio con la costante V, il simbolo funzionale a un posto c (per indicare la cardinalità di un oggetto), e simboli di predicato i^1 (essere un insieme), m^2 (essere maggiore di) e s^2 (essere un sottoinisieme di). Il ragionamento è corretto se e solo se:

$$\begin{array}{l} \forall x(i(x) \rightarrow \exists y(i(y) \wedge m(c(y), c(x)))), \\ \forall x \forall y(s(x,y) \rightarrow \neg m(c(x), c(y))) \\ \forall x\, s(x, V) \ \models\ \neg i(V) \end{array}$$

Assumiamo che il ragionamento non sia corretto. Quindi esistono un'interpretazione $\mathcal{M}$ e un'assegnazione s su $\mathcal{M}$ tali che:

1. $(\mathcal{M}, s) \models \forall x(i(x) \rightarrow \exists y(i(y) \wedge m(c(y), c(x))))$
2. $(\mathcal{M}, s) \models \forall x \forall y(s(x, y) \rightarrow \neg m(c(x), c(y)))$
3. $(\mathcal{M}, s) \models \forall x\, s(x, V)$
4. $(\mathcal{M}, s) \not\models \neg i(V)$

Sia D il dominio di $\mathcal{M}$ e $d_0 \in D$ l'interpretazione di V:

$$\mathcal{M}(V) = d_0$$

Per la 4 si ha che

(a) $d_0 \in \mathcal{M}(i)$

Per la 1 si ha che per ogni $d \in D$:

$$(\mathcal{M}, s[d/x]) \models i(x) \rightarrow \exists y(i(y) \wedge m(c(y), c(x)))$$

quindi in particolare anche

$$(\mathcal{M}, s[d_0/x]) \models i(x) \rightarrow \exists y(i(y) \wedge m(c(y), c(x)))$$

Da ciò segue che $(\mathcal{M}, s[d_0/x]) \not\models i(x)$ oppure $(\mathcal{M}, s[d_0/x]) \models \exists y(i(y) \wedge m(c(y), c(x)))$. Ma il primo caso (equivalente a $d_0 \notin \mathcal{M}(i)$) non è possibile, perché contraddice la (a), quindi si deve avere $(\mathcal{M}, s[d_0/x]) \models \exists y(i(y) \wedge m(c(y), c(x)))$, cioè deve esistere un oggetto $d_1 \in D$ tale che:

(b) $d_1 \in \mathcal{M}(i)$

e $(\mathcal{M}, s[d_0/x, d_1/y]) \models m(c(y), c(x))$. Se $F : D \rightarrow D$ è l'interpretazione di c ($F = \mathcal{M}(c)$), e

$$F(d_0) = d_2 \qquad F(d_1) = d_3$$

si deve allora avere che $\langle s[d_0/x, d_1/y](c(y)), s[d_0/x, d_1/y](c(x)) \rangle \in \mathcal{M}(m)$, cioè

(c) $\langle d_3, d_2 \rangle \in \mathcal{M}(m)$

Consideriamo ora la 3: si deve avere che per ogni $d \in D$, $(\mathcal{M}, s[d/x]) \models s(x, V)$, cioè per ogni $d \in D$ $\langle d, d_0 \rangle \in \mathcal{M}(s)$. In particolare anche:

(d) $\langle d_1, d_0 \rangle \in \mathcal{M}(s)$

Per la 2, si deve avere che per ogni $d, d' \in D$, o $(\mathcal{M}, s[d/x, d'/y]) \not\models s(x, y)$ oppure $(\mathcal{M}, s[d/x, d'/y]) \models \neg m(c(x), c(y))$. Ciò vale anche per $d = d_1$ e $d' = d_0$, quindi deve valere uno dei due casi seguenti:

(e1) $(\mathcal{M}, s[d_1/x, d_0/y]) \not\models s(x, y)$, cioè $\langle d_1, d_0 \rangle \notin \mathcal{M}(s)$;

(e2) $(\mathcal{M}, s[d_1/x, d_0/y]) \models \neg m(c(x), c(y))$, cioè $\langle d_3, d_2 \rangle \notin \mathcal{M}(m)$.

Ma entrambi i casi sono impossibili: (e1) contraddice (d) e (e2) contraddice (c).

Quindi è impossibile che esista un'interpretazione in cui sono vere tutte le premesse e falsa la conclusione: il ragionamento è corretto.

(O16) È possibile formalizzare il problema utilizzando un linguaggio con i simboli di predicato b^1 (essere un botanico), e^1 (essere eccentrico) e a^2 (amare). Il ragionamento è corretto se e solo se:

$$\begin{array}{l} \exists x(b(x) \wedge e(x)), \\ \exists x(b(x) \wedge \forall y(e(y) \rightarrow \neg a(x, y))) \\ \qquad\qquad \models \exists x(b(x) \wedge \neg\forall y(b(y) \rightarrow a(y, x))) \end{array}$$

Quindi il ragionamento non è corretto se e solo se esistono un'interpretazione $\mathcal{M}$ e un'assegnazione s su $\mathcal{M}$ tali che:

1. $(\mathcal{M}, s) \models \exists x(b(x) \wedge e(x))$
2. $(\mathcal{M}, s) \models \exists x(b(x) \wedge \forall y(e(y) \rightarrow \neg a(x, y)))$
3. $(\mathcal{M}, s) \not\models \exists x(b(x) \wedge \neg\forall y(b(y) \rightarrow a(y, x)))$

Assumiamo che ciò sia vero. Si ha che:

1. $(\mathcal{M}, s) \models \exists x(b(x) \wedge e(x))$ se e solo se esiste un oggetto d_0 nel dominio D di $\mathcal{M}$ tale che: $(\mathcal{M}, s[d_0/x]) \models b(x) \wedge e(x)$, cioè tale che:
   (1a) $d_0 \in \mathcal{M}(b)$, e
   (1b) $d_0 \in \mathcal{M}(e)$.
2. $(\mathcal{M}, s) \models \exists x(b(x) \wedge \forall y(e(y) \rightarrow \neg a(x, y)))$ se e solo se esiste $d_1 \in D$ tale che
   (2a) $d_1 \in \mathcal{M}(b)$

   e per ogni $d \in D$, $d \notin \mathcal{M}(e)$ oppure $\langle d_1, d \rangle \notin \mathcal{M}(a)$. Quest'ultima affermazione vale anche per $d = d_0$, dunque, per (1b), si ha che
   (2b) $\langle d_1, d_0 \rangle \notin \mathcal{M}(a)$
3. $(\mathcal{M}, s) \not\models \exists x(b(x) \wedge \neg\forall y(b(y) \rightarrow a(y, x)))$ se e solo se per ogni oggetto $d \in D$, $(\mathcal{M}, s[d/x]) \not\models b(x) \wedge \neg\forall y(b(y) \rightarrow a(y, x))$. In particolare, questo vale anche per $d = d_0$, quindi $(\mathcal{M}, s[d_0/x]) \not\models b(x) \wedge \neg\forall y(b(y) \rightarrow a(y, x))$, cioè $d_0 \notin \mathcal{M}(b)$ oppure $(\mathcal{M}, s[d_0/x]) \not\models \neg\forall y(b(y) \rightarrow a(y, x))$.
   Il primo caso è impossibile, per (1a), quindi si ha che
   $(\mathcal{M}, s[d_0/x]) \models \forall y(b(y) \rightarrow a(y, x))$. Cioè, per ogni $d' \in D$:
   $(\mathcal{M}, s[d_0/x][d'/y]) \models b(y) \rightarrow a(y, x)$. Questo vale anche, in particolare, per $d' = d_1$, quindi $(\mathcal{M}, s[d_0/x][d_1/y]) \models b(y) \rightarrow a(y, x)$. Ciò significa che o $d_1 \notin \mathcal{M}(b)$ oppure $\langle d_1, d_0 \rangle \in \mathcal{M}(a)$. Ma entrambi i casi sono impossibili: il primo contraddice (2a) e il secondo contraddice (2b).

Di conseguenza, l'ipotesi che esistano un'interpretazione $\mathcal{M}$ e un'assegnazione s su $\mathcal{M}$ che soddisfano tutte le premesse e non soddisfano la conclusione è assurda, dunque il ragionamento è corretto.

Se la conclusione fosse interpretata diversamente e fosse rappresentata dalla formula:

$$\exists x(b(x) \wedge \forall y(b(y) \rightarrow \neg a(y, x)))$$

(che è più forte della conclusione considerata sopra), allora il ragionamento non sarebbe corretto. Infatti:

1. $(\mathcal{M}, s) \models \exists x(b(x) \wedge e(x))$ se e solo se esiste $d_0 \in D$ tale che:
   (1a) $d_0 \in \mathcal{M}(b)$, e
   (1b) $d_0 \in \mathcal{M}(e)$.
2. $(\mathcal{M}, s) \models \exists x(b(x) \wedge \forall y(e(y) \rightarrow \neg a(x, y)))$ se e solo se esiste $d_1 \in D$ tale che

(2a) $d_1 \in \mathcal{M}(b)$, e

(2b) per ogni $d \in D$, $d \notin \mathcal{M}(e)$ oppure $\langle d_1, d\rangle \notin \mathcal{M}(a)$.

3. $(\mathcal{M}, s) \not\models \exists x(b(x) \wedge \forall y(b(y) \rightarrow \neg a(y,x)))$ se e solo se per ogni $d \in D$, vale uno dei casi seguenti:

   (3a) $d \notin \mathcal{M}(b)$,

   oppure $(\mathcal{M}, s[d/x]) \not\models \forall y(b(y) \rightarrow \neg a(y,x))$, cioè esiste $d_2 \in D$, tale che: $(\mathcal{M}, s[d/x][d_2/y]) \not\models b(y) \rightarrow \neg a(y,x)$. In altri termini, esiste $d_2 \in D$, tale che $(\mathcal{M}, s[d/x][d_2/y]) \models b(y)$ e $(\mathcal{M}, s[d/x][d_2/y]) \not\models \neg a(y,x)$, cioè:

   (3b) esiste $d_2 \in D$, tale che $d_2 \in \mathcal{M}(b)$ e $\langle d_2, d\rangle \in \mathcal{M}(a)$.

Consideriamo ora l'interpretazione $\mathcal{M}$ con dominio $D = \{0,1\}$, tale che $\mathcal{M}(b) = \{0,1\}$, $\mathcal{M}(e) = \{0\}$ e $\mathcal{M}(a) = \{\langle 0,0\rangle, \langle 1,1\rangle$, e verifichiamo che $\mathcal{M}$ verifica le due premesse e falsifica la conclusione.

Se si considera $d_0 = 0$, si ha che (1a) e (1b) sono vere. Se si considera $d_1 = 1$, (2a) è vera. Inoltre, $\langle 1,0\rangle \notin \mathcal{M}(a)$ e $1 \notin \mathcal{M}(e)$, quindi anche (2b) è vera.

Per dimostrare che la conclusione è falsa in $\mathcal{M}$ mostriamo che (3b) è vera per ogni $d \in D$. Per ogni $d \in D$ si ha infatti che $d \in \mathcal{M}(b)$ e $\langle d,d\rangle \in \mathcal{M}(a)$.

Il ragionamento, dunque, non è corretto.

Capitolo 4

Deduzione Automatica

4.1 Introduzione

Un sistema di deduzione automatica costituisce un apparato per ragionare in modo meccanico. Per automatizzare il ragionamento logico è dunque necessario definire un metodo meccanico per eseguire dimostrazioni. Supponiamo infatti di avere una teoria logica T (un insieme di formule) che rappresenti le conoscenze su un determinato dominio. Come si è visto nel paragrafo 2.6, risolvere un problema nel dominio in oggetto si può spesso ridurre alla questione di determinare se una certa formula A è una conseguenza logica di T, o, equivalentemente, se A è un teorema della teoria T. Perché questo tipo di problemi sia risolubile in modo meccanico occorre dunque avere una procedura automatica che, data una teoria logica T e una formula A, risponda alla domanda "A è un teorema di T? o, in altri termini, "esiste una dimostrazione di A a partire dagli assiomi di T?. Questo tipo di problemi è in generale soltanto semidecidibile nella logica dei predicati, come si è visto nel paragrafo 3.5.2: è dimostrato cioè che non esiste una procedura di tal sorta che sia sempre in grado di dare una risposta; qualsiasi procedura infatti può trovarsi di fronte a qualche A e T che la fanno entrare in un ciclo di calcoli che non termina mai e che dunque non porta a nessuna risposta. Tuttavia, le procedure migliori garantiscono che ciò possa accadere soltanto nel caso in cui A *non* sia un teorema di T.

La forma di ragionamento che ha attirato l'attenzione quasi esclusiva dei logici fino a tempi recentissimi è il ragionamento tipico della matematica, che presuppone l'esistenza di un mondo di oggetti statico (atemporale), del quale si rappresentano proprietà che sono senz'altro o vere o false (mai, ad esempio, solo probabili oppure semplicemente sconosciute) e del quale non interessa altro che dichiarare ciò che è vero. Non si vuole, ad esempio, esprimere che qualche proprietà è "necessariamente vera, oppure che qualcuno crede o conosce qualcosa. Il formalismo che abbiamo introdotto in questo testo è, in effetti, quello relativo a questo tipo di logica, la logica classica. Sebbene questa logica non sia sempre perfettamente adeguata per rappresentare conoscenze di tipo non matematico, essa viene spesso prescelta comunque per via del fatto che è il sistema in cui è più semplice automatizzare il processo di deduzione.

Per la logica classica si possono definire diversi sistemi deduttivi, tra loro equivalenti nel senso che permettono di derivare gli stessi teoremi, tra i quali quello introdotto nel paragrafo 3.4.3. Alcuni di essi sono adatti a essere automatizzati, altri no, perché in essi la costruzione di una dimostrazione è necessariamente guidata da scelte che richiedono intuito e che non possono essere sostituite da un esame esaustivo di tutte le possibilità. Il sistema Hilbertiano rientra in questa categoria.

In questo capitolo esamineremo alcuni metodi di dimostrazione automatizzabili per la logica proposizionale e la logica dei predicati. Si tenga presente comunque che, nel caso della logica dei predicati, nessun metodo (automatizzabile o no) può *decidere* la validità di una formula o la relazione di conseguenza logica, come si è visto nel paragrafo 3.5.2. Il capitolo non contiene le dimostrazioni di correttezza e completezza dei sistemi presentati, per le quali si rimanda a [2, 3].

4.2 Refutazione

L'obiettivo della deduzione automatica è quello di definire algoritmi che consentano di risolvere problemi della forma: dato un insieme S di formule e una formula A, è vero che $S \models A$?

La maggior parte dei metodi di dimostrazione automatica sono *metodi di refutazione*, che, sfruttando il Teorema 3.1, riconducono il problema di determinare se $S \models A$ al problema di determinare se $S \cup \{\neg A\}$ è insoddisfacibile. Un metodo di refutazione è dunque un metodo che consente di dimostrare l'insoddisfacibilità di insiemi di formule.

4.3 Il metodo dei tableaux per la logica proposizionale

In questa sezione presentiamo un semplice metodo di dimostrazione automatica per la logica proposizionale, il metodo dei *tableaux semantici*. Tale metodo consiste essenzialmente in un sistema di ricerca esaustiva di modelli per una formula o insieme di formule.

Assumiamo qui che $\equiv$ sia un simbolo definito, quindi i simboli logici del linguaggio sono $\neg, \wedge, \vee, \rightarrow$.

Un tableau[1] è un albero, rappresentato con la radice in alto, i cui nodi sono etichettati da formule. Un tableau per una formula F, viene inizializzato con un unico nodo, etichettato proprio dalla formula F. Questo è il "tableau iniziale" per F. La costruzione del tableau procede poi aggiungendo nodi, in accordo con determinate *regole di espansione*, che "analizzano il significato" delle formule. La costruzione di un tableau con radice F essenzialmente consiste nella ricerca esaustiva di un modello per F: se un tale modello viene trovato, la formula F è soddisfacibile, altrimenti si è dimostrato che non esistono modelli di F (e quindi la negazione di F è valida).

Le regole di espansione vengono rappresentate in una delle seguenti forme:

$$\frac{A}{A_0} \qquad\qquad \frac{A}{\begin{array}{c}A_0\\A_1\end{array}} \qquad\qquad \frac{B}{B_0 \qquad B_1}$$

L'applicazione di una regola della prima forma aggiunge la formula A_0 in fondo a ciascun ramo del tableau che contiene A. L'applicazione di una regola della seconda forma aggiunge le formule A_0 e A_1 in fondo a ciascun ramo che contiene la formula A. L'applicazione di regole della terza forma crea invece una ramificazione nell'albero: ciascun ramo che contiene la formula B viene espanso aggiungendo i due nodi B_0 e B_1 come figli dell'ultimo nodo del ramo, provocando la biforcazione del ramo stesso.

Le regole della prima e seconda forma sono chiamate α-regole, quelle della terza forma β-regole: Le regole sono le seguenti:

α-regole:	$\dfrac{A \wedge B}{\begin{array}{c}A\\B\end{array}}$	$\dfrac{\neg(A \vee B)}{\begin{array}{c}\neg A\\\neg B\end{array}}$	$\dfrac{\neg(A \rightarrow B)}{\begin{array}{c}A\\\neg B\end{array}}$	$\dfrac{\neg\neg A}{A}$
β-regole:	$\dfrac{A \vee B}{A \qquad B}$	$\dfrac{\neg(A \wedge B)}{\neg A \qquad \neg B}$	$\dfrac{A \rightarrow B}{\neg A \qquad B}$	

[1] Al singolare: "tableau"; al plurale: "tableaux".

L'intuizione dietro queste regole è la seguente. La prima α-regola (in alto a sinistra) stabilisce che qualsiasi interpretazione è un modello di $A \wedge B$ se e solo se è anche un modello di A e un modello di B. La seconda α-regola che qualsiasi interpretazione è un modello di $\neg(A \vee B)$ se e solo se è anche un modello di $\neg A$ e un modello di $\neg B$. La prima β-regola stabilisce che una qualsiasi interpretazione è un modello di $A \vee B$ se e solo se è un modello di A *oppure* un modello di B.

È conveniente, per compattezza, utilizzare una "notazione uniforme" per le regole di espansione, che utilizza la nozione di α e β-formule, con le rispettive α_0 e α_1, o β_0 e β_1, come descritto nelle seguenti tabelle:

α	α_0	α_1
$A \wedge B$	A	B
$\neg(A \vee B)$	$\neg A$	$\neg B$
$\neg(A \rightarrow B)$	A	$\neg B$
$\neg\neg A$	A	

β	β_0	β_1
$A \vee B$	A	B
$\neg(A \wedge B)$	$\neg A$	$\neg B$
$A \rightarrow B$	$\neg A$	B

Le regole di espansione sono allora semplicemente:

$$\alpha\text{-regola}: \quad \frac{\alpha}{\begin{array}{c}\alpha_0\\(\alpha_1)\end{array}} \qquad\qquad \beta\text{-regola}: \quad \frac{\beta}{\beta_0 \qquad \beta_1}$$

Ogni ramo del tableau rappresenta un possibile modello. Se un ramo contiene due formule della forma A e $\neg A$, oppure se contiene $\perp$, esso si dice *chiuso*. Quando, durante la costruzione di un tableau, si genera un ramo chiuso, questo, normalmente, non viene ulteriormente espanso: infatti tutti i rami che sarebbero generati come sua estensione sono rami chiusi, che non possono rappresentare alcuna interpretazione, in quanto A e $\neg A$ dovrebbero essere contemporaneamente vere. Un ramo che non è chiuso si dice *aperto*.

Se è possibile proseguire la costruzione del tableau fino ad ottenere tutti rami chiusi, ottenendo cioè un *tableau chiuso*, allora la formula di partenza non ha modelli, è cioè insoddisfacibile. Altrimenti, se, quando tutte le formule del tableau sono state espanse (ad esse è stata cioè applicata una delle regole applicabili), vi sono ancora rami non chiusi (aperti), allora ciascuno di essi rappresenta un possibile modello (o, in generale, un insieme di modelli) della formula iniziale.

Si noti che, in logica proposizionale, la costruzione di un tableau termina sempre, in quanto le regole di espansione generano formule più semplici della formula espansa.

Come primo esempio, consideriamo l'albero della Figura 4.1. Esso è un tableau per la formula $(p \wedge q \rightarrow r) \wedge (\neg p \rightarrow s) \wedge q$. I primi nodi, nel ramo unico iniziale, sono ottenuti applicando la regola per formule della forma $A \wedge B$. L'espansione del nodo $p \wedge q \rightarrow r$ provoca la prima ramificazione, con $\neg(p \wedge q)$ a sinistra e r a destra. L'espansione di $\neg p \rightarrow s$ aggiunge un'ulteriore ramificazione alla fine di ciascun ramo, con $\neg\neg p$ a sinistra e s a destra. I due nodi $\neg\neg p$ vengono entrambi espansi aggiungendo p in fondo al rispettivo ramo. Infine, l'espansione di $\neg(p \wedge q)$ aggiunge le ultime ramificazioni sulla sinistra, con $\neg p$ e $\neg q$ in ciascun ramo. I rami marcati con una croce, $\times$, sono chiusi: il primo infatti contiene p e $\neg p$, il secondo e il terzo q e $\neg q$. Vi sono tre rami aperti nel tableau, il primo contenente i letterali $q, s, \neg p$, il secondo q, r, p, il terzo q, r, s. Il tableau non può essere ulteriormente espanso, quindi la formula $(p \wedge q \rightarrow r) \wedge ((\neg p \rightarrow s) \wedge q)$ è soddisfacibile e i tre rami aperti rappresentano modelli per essa. Ad esempio, il primo ramo rappresenta l'insieme delle interpretazioni di p, q, r, s in cui q e s sono vere e p è falsa; ciascuna di tali interpretazioni è un modello di $(p \wedge q \rightarrow r) \wedge ((\neg p \rightarrow s) \wedge q)$.

Se si vuole controllare la validità di una formula A, si inizializza il tableau con la formula $\neg A$: se la ricerca sistematica di un modello di $\neg A$ fallisce, vuol dire che $\neg A$ è insoddisfacibile, dunque A è valida.

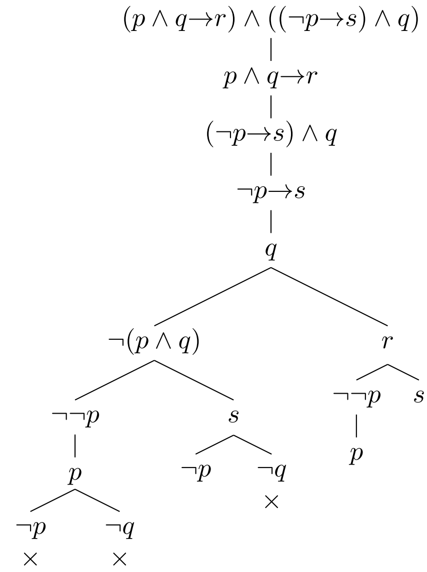


Figura 4.1: Un tableau per la formula $(p \wedge q \rightarrow r) \wedge (\neg p \rightarrow s) \wedge q$

Il metodo dei tableaux si può utilizzare anche per dimostrare che una data formula A è una conseguenza logica di un insieme di formule S: il tableau viene allora inizializzato con l'insieme di formule $S \cup \{\neg A\}$. Ciò significa che il tableau iniziale è un albero costituito da un unico ramo, in cui occorrono tutte le formule di $S \cup \{\neg A\}$. Se $S = \{G_1, ..., G_n\}$, tale tableau ha la forma riportata nella figura 4.2. In generale, se T è un tableau per un insieme di formule S, e T^* si ottiene da T applicando una regola di espansione, allora T^* è ancora un tableaux per S.

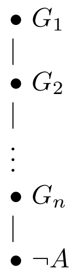


Figura 4.2: Tableau iniziale per l'insieme di formule $\{G_1, ..., G_n, \neg A\}$.

Ad esempio, allo scopo di dimostrare che $A \wedge B \rightarrow C, \neg A \rightarrow D \models B \rightarrow (C \vee D)$, costruiamo il tableau iniziale riportato in Figura 4.3, a sinistra. Se in seguito viene espansa la formula $\neg(B \rightarrow (C \vee D))$, otteniamo il tableau riportato al centro, dove segnamo con un $\surd$ la formula espansa. Espandendo poi l'ultima formula di tale tableau, si ottiene il tableau riportato a destra, sempre in Figura 4.3.

In seguito, l'espansione del primo nodo genera il tableau riportato nella Figura 4.4, a sinistra, e la successiva espansione del nodo $\neg A \rightarrow D$ genera il tableau riportato a destra, sempre in Figura 4.4.

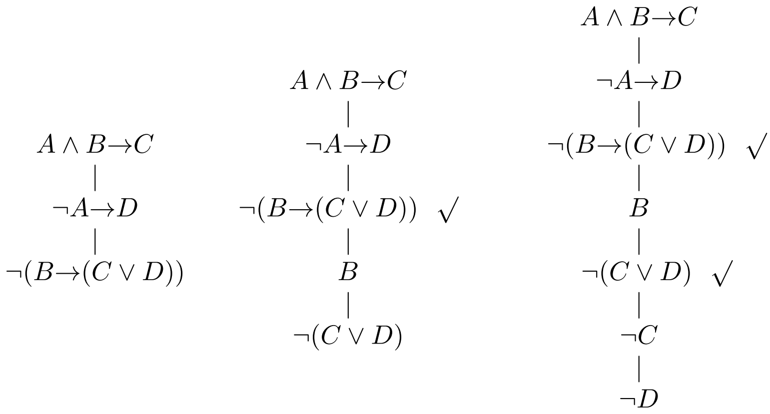


Figura 4.3: Tableaux per $\{A \wedge B \to C,\ \neg A \to D,\ \neg(B \to (C \vee D))\}$.

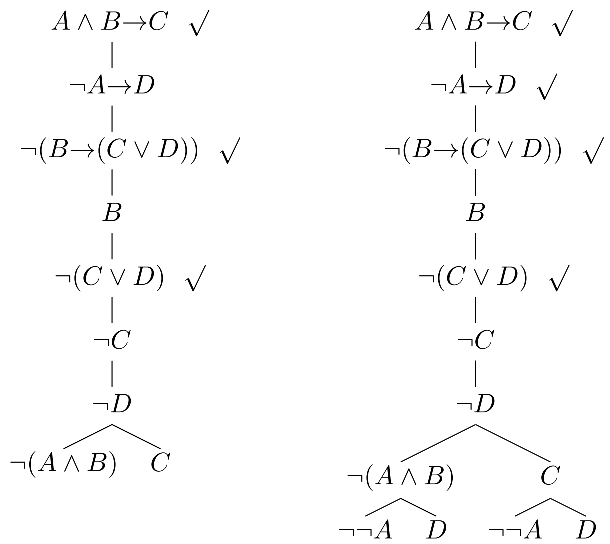


Figura 4.4: Tableaux per $\{A \wedge B \to C,\ \neg A \to D,\ \neg(B \to (C \vee D))\}$.

Infine, l'espansione dei due nodi etichettati con $\neg\neg A$ e quella di $\neg(A \wedge B)$ genera il tableau chiuso riportato in Figura 4.5. Il tableau è completamente espanso: i soli nodi non espansi sono etichettati da letterali. Il primo ramo (a partire da sinistra) è chiuso perché contiene A e $\neg A$, il secondo contiene B e $\neg B$, il terzo contiene D e $\neg D$, gli ultimi due contengono C e $\neg C$.

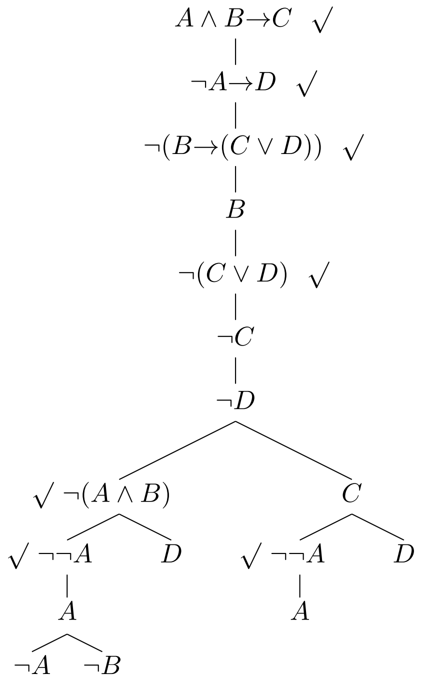


Figura 4.5: Una dimostrazione mediante tableaux di $A \wedge B \rightarrow C, \neg A \rightarrow D \vdash B \rightarrow (C \vee D)$.

Recapitolando:

- Un ramo di un tableau è **chiuso** se esso contiene sia A che $\neg A$ per qualche formula A, oppure contiene $\perp$.
- Un tableau è chiuso se tutti i suoi rami sono chiusi.
- Una dimostrazione mediante tableau di $S \vdash A$ è un tableau chiuso per $S \cup \{\neg A\}$.
- Un tableau si dice *completo* se ogni nodo che possa essere espanso è stato espanso almeno una volta.

Il metodo dei tableaux per la logica proposizionale è corretto e completo:

$$S \models A \text{ sse esiste una dimostrazione mediante tableaux di } S \vdash A.$$

Inoltre, nei tableaux proposizionali:

1. Se S è insoddisfacibile, è sufficiente espandere al massimo una volta ogni formula di un tableau per S per ottenere un tableau chiuso.
2. L'ordine di espansione delle formule è irrilevante per la completezza (*invertibilità delle regole di espansione*).

3. Se A è una formula, T è un tableau completo per A, $\mathcal{B}_1, ..., \mathcal{B}_n$ sono tutti i rami aperti di T, e se $\mathcal{C}(\mathcal{B}_i)$ è la congiunzione dei letterali in $\mathcal{B}_i$, allora:
$$A \;\leftrightarrow\; \mathcal{C}(\mathcal{B}_1) \vee .... \vee \mathcal{C}(\mathcal{B}_n)$$

Come esempio relativo al precedente punto 3, si consideri il tableau riportato in Figura 4.6. Il ramo più a destra è chiuso, quindi i rami aperti sono i primi tre, $\mathcal{B}_1, \mathcal{B}_2$ e $\mathcal{B}_3$, contenenti, rispettivamente i letterali $\{p, \neg q\}$, $\{p, \neg r\}$ e $\{p, q, \neg s\}$. Quindi $\mathcal{C}(\mathcal{B}_1) = p \wedge \neg q$, $\mathcal{C}(\mathcal{B}_2) = p \wedge \neg r$ e $\mathcal{C}(\mathcal{B}_3) = p \wedge q \wedge \neg s$. Quindi $\neg((p \rightarrow (q \wedge r)) \wedge (p \rightarrow (q \rightarrow (s \wedge p))))$ è logicamente equivalente a $(p \wedge \neg q) \vee (p \wedge \neg r) \vee (p \wedge q \wedge \neg s)$.

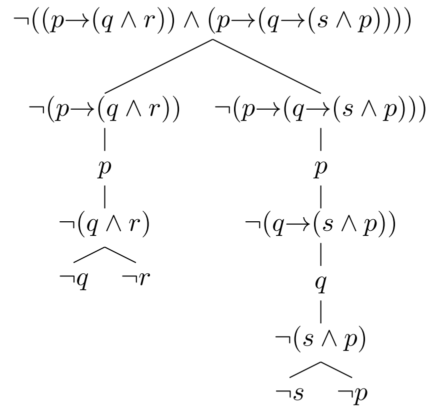


Figura 4.6: Uso dei tableaux per la trasformazione in forma normale

Come si vede, dunque, il metodo dei tableaux si può utilizzare anche per trasformare le formule proposizionali in forma normale disgiuntiva. Come ulteriore esempio, dal tableau riportato in Figura 4.1 si ottiene la formula $(q \wedge s \wedge \neg p) \vee (q \wedge r \wedge p) \vee (q \wedge r \wedge s)$, che è logicamente equivalente alla formula iniziale $(p \wedge q \rightarrow r) \wedge (\neg p \rightarrow s) \wedge q$.

In modo "duale" si può ottenere la forma normale congiuntiva di una formula F: si inizializza il tableau con $\neg F$, si espande il tableau completamente, e si identificano tutti i rami aperti. Per ciascuno di essi si costruisce l'insieme costituito dai *complementi* dei letterali nel ramo (il complemento di un letterale p è $\neg p$ e il complemento di $\neg p$ è p). In tal modo si ottengono insiemi $S_1, ..., S_k$ di letterali. Una forma normale congiuntiva equivalente a F è allora la congiunzione di tutte le disgiunzioni $\bigvee_{\ell \in S_i} \ell$.

In altri termini, per trasformare A in forma normale congiuntiva:

- si costruisce un tableau completo per $\neg A$
- si eliminano i rami chiusi
- si costruisce la congiunzione delle disgiunzioni dei complementi dei letterali nei rami aperti.

Ad esempio, per trasformare in CNF la formula $\neg(p \rightarrow (q \wedge r) \vee (q \rightarrow s))$, costruiamo innanzitutto un tableau completo per $\neg\neg(p \rightarrow (q \wedge r) \vee (q \rightarrow s))$, come riportato in Figura 4.7. Tutti i rami sono aperti e contengono i letterali $\{\neg p\}$ $\{q, r\}$, $\{\neg q\}$, $\{s\}$. I corrispondenti insiemi con i complementi dei letterali sono $\{p\}$ $\{\neg q, \neg r\}$, $\{q\}$, $\{\neg s\}$. Si ottiene quindi la formula in CNF $p \wedge (\neg q \vee \neg r) \wedge q \wedge \neg s$.

Come ulteriore esempio, dal tableau della figura 4.6 si ottengono, dai tre rami aperti, i letterali (già complementati) $\{\neg p, q\}$, $\{\neg p, r\}$ e $\{\neg p, \neg q, s\}$. Quindi $(p \rightarrow (q \wedge r)) \wedge (p \rightarrow (q \rightarrow (s \wedge p)))$ è logicamente equivalente a $(\neg p \vee q) \wedge (\neg p \vee r) \wedge (\neg p \vee \neg q \vee s)$.

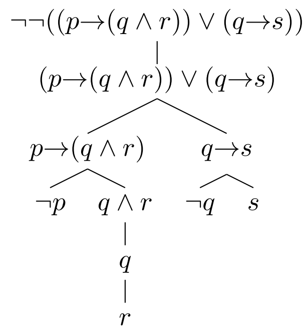


Figura 4.7: Tableau per $\neg\neg((p\to(q\wedge r))\vee(q\to s))$

Il fatto che la CNF di una formula si possa ottenere in questo modo deriva dalla "dualità" tra CNF e DNF: il tableau della figura 4.7 può servire a costruire, secondo il metodo prima visto, la forma normale disgiuntiva di $\neg\neg((p\to(q\wedge r))\vee(q\to s))$: otteniamo $\neg p\vee(q\wedge r)\vee\neg q\vee s$, quindi

$$\neg\neg(p\to(q\wedge r))\vee(q\to s) \;\leftrightarrow\; \neg p\vee(q\wedge r)\vee\neg q\vee s$$

Dunque:

$$\begin{array}{rcl}\neg((p\to(q\wedge r))\vee(q\to s)) & \leftrightarrow & \neg(\neg p\vee(q\wedge r)\vee\neg q\vee s)\\ & \leftrightarrow & p\wedge(\neg q\vee\neg r)\wedge q\wedge\neg s\end{array}$$

L'ultima equivalenza si ottiene applicando le leggi di De Morgan.

Si noti, infine, che, dato che la costruzione di un tableau proposizionale termina sempre, il metodo dei tableaux costituisce anche un metodo per dimostrare che una formula proposizionale A *non* è valida: si costruisce un tableau completo per $\neg A$; se tale tableau ha un ramo aperto, esso rappresenta un contromodello di A e dunque A non è valida.

4.4 Unificazione

La maggior parte dei metodi di deduzione automatica per la logica dei predicati del primo ordine utilizza la nozione di unificazione e la costruzione di "unificatori più generali" di un insieme di espressioni. Introduciamo queste nozioni in questo paragrafo, assieme ad un algoritmo per calcolare l'unificatore più generale di una coppia di espressioni. Un'implementazione OCaml di quanto presentato qui si può trovare nel programma `fol.ml` [7].

4.4.1 Sostituzioni

Una sostituzione è un insieme finito della forma:

$$\{t_1/x_1, ..., t_n/x_n\}$$

dove:

- $x_1, ..., x_n$ sono variabili
- $t_1, ..., t_n$ sono termini
- per ogni i, $t_i \neq x_i$
- per ogni i, j se $i \neq j$ allora $x_i \neq x_j$

Se E è un'espressione e $\theta = \{t_1/x_1, ..., t_n/x_n\}$ una sostituzione, l'applicazione di θ a E, indicata con $E\theta$, è l'*istanza* di E che si ottiene sostituendo simultaneamente ogni occorrenza di x_i con t_i (per $1 \leq i \leq n$).

Ad esempio, l'applicazione della sostituzione $\{f(z,z)/x, c/z\}$ alla formula $p(f(x,y), x, g(z))$ è:

$$p(f(x,y), x, g(z))\{f(z,z)/x, c/z\} = p(f(f(z,z), y), f(z,z), g(c))$$

4.4.2 Composizione di sostituzioni

Siano $\theta = \{t_1/x_1, ..., t_n/x_n\}$, e $\sigma = \{u_1/y_1, ..., u_m/y_m\}$, due sostituizioni. La composizione di θ con σ, denotata da

$$\theta \circ \sigma$$

è la sostituzione che si ottiene da

$$\{t_1\sigma/x_1, ..., t_n\sigma/x_n, u_1/y_1, ..., u_m/y_m\}$$

eliminando tutti gli elementi:

(1) $t_j\sigma/x_j$ tali che $t_j\sigma = x_j$

(2) u_ii/y_i tali che $y_i \in \{x_1, ..., x_n\}$

Ad esempio, se $\theta = \{f(y)/x, z/y\}$ e $\sigma = \{a/x, b/y, y/z\}$, allora dall'insieme $\{f(y)\sigma/x, z\sigma/y, a/x, b/y, y/z\} = \{f(b)/x, y/y, a/x, b/y, y/z\}$ eliminiamo l'elemento y/y per (1), l'elemento a/x per (2) e l'elemento b/y per (2), ottenendo $\theta \circ \sigma = \{f(b)/x, y/z\}$.

La proprietà fondamentale della composizione di sostituzioni è la seguente: per ogni espressione E

$$E(\theta \circ \sigma) \;=\; (E\theta)\sigma$$

Ad esempio: se θ e σ sono come sopra, si ha:

$$h(x, g(y), z)(\theta \circ \sigma) = h(x, g(y), z)\{f(b)/x, y/z\} = h(f(b), g(y), y)$$

$$(h(x, g(y), z)\theta)\sigma = h(f(y), g(z), z)\sigma = h(f(b), g(y), y)$$

4.4.3 Unificazione

Una sostituzione θ è un *unificatore* per l'insieme di espressioni $\{E_1, ..., E_k\}$ sse:

$$E_1\theta = E_2\theta = ... = E_k\theta$$

L'insieme $\{E_1, ..., E_k\}$ è *unificabile* sse esiste un unificatore per esso.

Un unificatore σ per $\{E_1, ..., E_k\}$ è un *unificatore più generale* (mgu = most general unifier) sse per ogni unificatore θ di $\{E_1, ..., E_k\}$ esiste una sostituzione λ tale che $\theta = \sigma \circ \lambda$.

In termini intuitivi, un unificatore più generale per $\{E_1, ..., E_k\}$ è un unificatore che non sostituisce (istanzia) più del necessario. Se σ è un mgu per $\{E_1, ..., E_k\}$, vuol dire che:

- σ è un unificatore: $E_1\sigma = E_2\sigma = ... = E_k\sigma$
- è più generale: se $E_1\theta = E_2\theta = ... = E_k\theta$, allora per qualche λ

$$E_i\theta = (E_i\sigma)\lambda$$

(θ è meno generale: istanzia più del necessario)

Ad esempio, $\sigma = \{f(a)/x, a/y\}$ è un unificatore per $\{p(x), p(f(y))\}$ ma non è un mgu. Infatti se $\theta = \{f(y)/x\}$, non esiste alcuna sostituzione λ tale che $\theta = \sigma \circ \lambda$. θ è invece un mgu per $\{p(x), p(f(y))\}$.

4.4.4 Algoritmo di unificazione di Robinson per due espressioni

Siano A e B due espressioni e sia k la posizione più a sinistra in cui le due sequenze di simboli differiscono. L'insieme $\{e_1, e_2\}$ delle due sottoespressioni che iniziano alla posizione k in A e B si chiama il *disagreement set* di A e B.

Algoritmo: calcola $mgu(A, B)$

- Inizializzazione:

$$\begin{array}{c} A_0 = A \\ B_0 = B \\ \sigma_0 = \emptyset \\ i = 0 \end{array}$$

- Ciclo:
  - Se $A_i = B_i$: uscire dal ciclo, con risultato σ_i, altrimenti proseguire
  - Sia $\{e, e'\}$ il *disagreement set* di A_i e B_i. Consideriamo i seguenti casi:
    1. se un elemento di $\{e, e'\}$ è una variabile x_i e l'altro un'espressione e_i in cui non occorre x_i, allora porre:

    $$\begin{array}{c} \sigma_{i+1} = \sigma_i \circ \{e_i/x_i\} \\ A_{i+1} = A_i\{e_i/x_i\} \\ B_{i+1} = B_i\{e_i/x_i\} \end{array}$$

       e proseguire nel ciclo.
    2. altrimenti uscire dal ciclo e riportare un fallimento: $\{e, e'\}$ non è unificabile

Si noti che il caso in cui un elemento del *disagreement set* è una variabile richiede di controllare l'occorrenza di una variabile in un'espressione. Tale controllo viene chiamato *occur check*.

4.5 Il metodo di risoluzione

4.5.1 Il caso proposizionale

Il metodo di risoluzione è un metodo di dimostrazione automatica che si basa su un sistema di inferenza con una sola regola: la regola di risoluzione. Questa si applica a formule di forma particolare, chiamate *clausole*. Una clausola è una disgiunzione di letterali:

$$L_1 \vee L_2 \vee .... \vee L_k$$

dove ciascun L_i è un letterale, cioè un atomo o la negazione di un atomo. Dato che, dal punto di vista semantico, in una disgiunzione non sono importanti né l'ordine dei disgiunti né eventuali occorrenze multiple degli stessi, una clausola si può semplicemente rappresentare mediante l'insieme dei suoi letterali: la clausola $L_1 \vee L_2 \vee .... \vee L_k$ può essere rappresentata dall'insieme $\{L_1, L_2, ..., L_k\}$.

Nel caso della logica proposizionale, la regola di risoluzione ha una forma molto semplice:

$$\frac{C_1 \cup \{p\} \ ; \quad C_2 \cup \{\neg p\}}{C_1 \cup C_2}$$

$C_1 \cup C_2$ è chiamato il *risolvente* di $C_1 \cup \{p\}$ e $C_2 \cup \{\neg p\}$. I letterali p e $\neg p$ si dicono *letterali complementari*. Le clausole $C_1 \cup \{p\}$ e $C_2 \cup \{\neg p\}$ sono le *clausole genitrici*.

Consideriamo ad esempio, la seguente derivazione, che mostra che $\neg p \vee q, p \vee r, \neg q \vee s \vdash_{RES} r \vee s$ (indichiamo con $\vdash_{RES}$ la derivabilità nel sistema di risoluzione):

$$\dfrac{\dfrac{\{\neg p, q\} \quad \{p, r\}}{\{q, r\}} \qquad \{\neg q, s\}}{\{r, s\}}$$

La derivazione può anche essere scritta usando esplicitamente la disgiunzione:

$$\dfrac{\dfrac{\neg p \vee q \quad p \vee r}{q \vee r} \qquad \neg q \vee s}{r \vee s}$$

La regola di risoluzione proposizionale è corretta: se C è un risolvente di C_1 e C_2, allora $C_1, C_2 \models C$. Di conseguenza, il sistema di risoluzione proposizionale è corretto:

$$\Gamma \vdash_{RES} C \implies \Gamma \models C$$

Ma il sistema di risoluzione non è completo rispetto alla derivabilità, ad esempio $\not\vdash_{RES} p \vee \neg p$, pur essendo $p \vee \neg p$ una clausola valida.

Il metodo di risoluzione è utilizzato infatti esclusivamente come metodo di refutazione, sfruttando il Teorema 3.1: per dimostrare $S \models A$ si *refuta* $S \cup \{\neg A\}$, cioè si dimostra che $S \cup \{\neg A\} \vdash \bot$. Il falso, $\bot$, è rappresentato dalla *clausola vuota*, per la quale si usa la notazione $\square$.

Il metodo di risoluzione proposizionale consiste dunque in questo: per "dimostrare" per risoluzione $S \models A$:

1. si trasforma $S \cup \{\neg A\}$ in *forma a clausole*. Per far ciò, si trasforma ogni formula B_i di $S \cup \{\neg A\}$ in una forma normale congiuntiva equivalente. Ciascuna disgiunzione è una clausola. Da ogni B_i si ottiene dunque un insieme di clausole:

$$\begin{array}{rcl} B_i & \Longrightarrow & (L_{1,1} \vee ... \vee L_{1,n_1}) \wedge ... \wedge (L_{k,1} \vee ... \vee L_{k,n_k}) \\ & \Longrightarrow & \{L_{1,1} \vee ... \vee L_{1,n_1},\ ...,\ L_{k,1} \vee ... \vee L_{k,n_k}\} \\ & \Longrightarrow & \{\{L_{1,1}, ..., L_{1,n_1}\}, ..., \{L_{k,1}, ..., L_{k,n_k}\}\} \end{array}$$

   La forma a clausole di $S \cup \{\neg A\}$ è l'unione di tutte le clausole che si ottengono da ciascuna formula dell'insieme.

2. Se S' è l'insieme di clausole ottenute, si dimostra che:

$$S' \vdash_{RES} \square$$

Per illustrare il procedimento, consideriamo ad esempio una dimostrazione di $p \wedge q \rightarrow r, p \rightarrow q \models p \rightarrow r$. Ciascuna formula in $\{p \wedge q \rightarrow r, p \rightarrow q, \neg(p \rightarrow r)\}$ viene trasformata in forma a clausole come segue:

$$\begin{array}{lll} a) & p \wedge q \rightarrow r & \Longrightarrow \neg(p \wedge q) \vee r \\ & & \Longrightarrow \neg p \vee \neg q \vee r \\ & & \Longrightarrow \{\neg p \vee \neg q \vee r\} \\ & & \Longrightarrow \{\{\neg p, \neg q, r\}\} \\ b) & p \rightarrow q & \Longrightarrow \neg p \vee q \\ & & \Longrightarrow \{\neg p \vee q\} \\ & & \Longrightarrow \{\{\neg p, q\}\} \\ c) & \neg(p \rightarrow r) & \Longrightarrow p \wedge \neg r \\ & & \Longrightarrow \{p, \neg r\} \\ & & \Longrightarrow \{\{p\}, \{\neg r\}\} \end{array}$$

L'insieme di clausole ottenute è dunque:

$$S = \{\{\neg p, \neg q, r\}, \{\neg p, q\}, \{p\}, \{\neg r\}\}$$

Da questo insieme si deriva la clausola vuota:

$$\dfrac{\dfrac{\dfrac{\neg p, \neg q, r \qquad \neg r}{\neg p, \neg q} \qquad \dfrac{\neg p, q \qquad p}{q}}{\neg p} \qquad p}{\Box}$$

La risoluzione proposizionale, come metodo di refutazione, è corretta e completa: infatti, se S è un insieme di clausole, allora S è insoddisfacibile sse $S \vdash_{RES} \Box$. Quindi, se $S \models A$ allora esiste una derivazione per risoluzione di $\Box$ dalla forma a clausole di $S \cup \{\neg A\}$.

4.5.2 Trasformazione in forma a clausole nella logica dei predicati

Estendiamo ora il metodo di risoluzione alla logica dei predicati. Il primo passo consiste nel definire un metodo per trasformare un insieme di formule in "forma a clausole", cioè in un insieme di disgiunzioni di letterali (formule atomiche o negazioni di formule atomiche). La trasformazione si applica a ciascuna formula dell'insieme. Per trasformare una formula A della logica dei predicati in forma a clausole, si procede per stadi, come illustrato in seguito.

In primo luogo si trasforma A in *forma normale prenessa* (vedi Paragrafo 3.3.5), cioè in una formula della forma:

$$\underbrace{Q_1 x_1 Q_2 x_2 ... Q_n x_n}_{prefisso} \; \underbrace{A}_{matrice}$$

dove $Q_1, Q_2, ...Q_n$ sono quantificatori e la matrice è senza quantificatori. Si può dimostrare che ogni formula è logicamente equivalente a una formula in forma normale prenessa. Si veda il Paragrafo 3.3.5 per le equivalenze logiche utilizzate nella trasformazione in forma prenessa.

Il secondo passaggio nella trasformazione consiste nell'ottenere, dalla forma prenessa di A una formula in *forma normale di Skolem.* Una formula è in forma di Skolem se essa è in forma prenessa e gli unici quantificatori nella matrice sono quantificatori universali. Una forma di Skolem ha dunque la forma:

$$\forall x_1 ... \forall x_n A$$

dove A è senza quantificatori.

Per trasformare una formula dalla forma prenessa a una forma di Skolem, occorre eliminare i quantificatori esistenziali. Ciascun quantificatore esistenziale $\exists y$ nel prefisso viene eliminato e tutte le occorrenze di y nella matrice vengono sostituite da un "termine di Skolem", della forma $f(x_1, ..., x_n)$ dove f è un simbolo funzionale *nuovo* e $\forall x_1, ..., \forall x_n$ sono tutti i quantificatori universali che precedono $\exists y$ nel prefisso:

$$\forall x_1 ... \forall x_n \exists y A \Longrightarrow \forall x_1 ... \forall x_n A[f(x_1, ..., x_n)/y]$$

Il simbolo funzionale nuovo f si chiama *funzione di Skolem.* Se il quantificatore $\exists y$ che si elimina non è preceduto da alcun quantificatore universale nella matrice, la variabile y è sostituita da una costante (sempre nuova).

L'uso di funzioni di Skolem "codifica" la dipendenza di un esistenziale dai quantificatori universali che lo precedono. La trasformazione, ad esempio, di $\forall x \exists y \, p(x, y)$ in $\forall x \, p(x, f(x))$ rappresenta il fatto che y "dipende" da x. Trasformando invece $\exists y \forall x \, p(x, y)$ in $\forall x \, p(x, c)$ si evidenzia il fatto che y non dipende da x.

Ad esempio, la formula $\exists x\forall y\forall z\exists u\forall w\exists v\; p(x,y,z,u,w,v)$ può essere trasformata in forma di Skolem come segue:

$$\begin{array}{ll} & \exists x\forall y\forall z\exists u\forall w\exists v\; p(x,y,z,u,w,v) \\ \Rightarrow & \forall y\forall z\exists u\forall w\exists v\; p(c,y,z,u,w,v) \\ \Rightarrow & \forall y\forall z\forall w\exists v\; p(c,y,z,f(y,z),w,v) \\ \Rightarrow & \forall y\forall z\forall w\; p(c,y,z,f(y,z),w,g(y,z,w)) \end{array}$$

Assumiamo qui che c, f, g siano simboli *nuovi*, cioè che non occorrono né nella formula iniziale, né in alcune delle altre formule dell'insieme che si vuole trasformare in forma a clausole, e che non siano utilizzati come simboli di Skolem nella trasformazione di nessun'altra formula.

Il terzo passo nella trasformazione consiste nell'eliminazione dei quantificatori universali. È implicito che tutte le variabili che occorrono nella formula che si ottiene siano quantificate universalmente. Si ottiene così una formula senza quantificatori

Gli ulteriori passaggi nella trasformazione corrispondono alla trasformazione in forma a clausole delle formule proposizionali: la matrice della formula viene trasformata in forma normale congiuntiva e questa infine in forma a clausole.

Recapitolando dunque, il procedimento di trasformazione di una formula A in forma a clausole consiste nel:

1. Trasformare A in forma normale prenessa $pre(A)$
2. Trasformare $pre(A)$ in forma normale di Skolem $sk(A)$
3. Eliminare i quantificatori universali da $sk(A)$
4. Trasformare la matrice in forma normale congiuntiva
5. Trasformare in insieme di clausole

Ad esempio, per trasformare la formula $\forall x\exists y\exists z(\neg p(x,y) \vee \neg(r(x,y,z)\rightarrow q(x,z)))$ in un insieme di clausole procediamo come segue:

$$\begin{array}{ll} & \forall x\exists y\exists z(\neg p(x,y) \vee \neg(r(x,y,z)\rightarrow q(x,z))) \\ \Rightarrow & \neg p(x,f(x)) \vee \neg(r(x,f(x),g(x))\rightarrow q(x,g(x))) \\ \Rightarrow & \neg p(x,f(x)) \vee (r(x,f(x),g(x)) \wedge \neg q(x,g(x))) \\ \Rightarrow & (\neg p(x,f(x)) \vee r(x,f(x),g(x))) \wedge (\neg p(x,f(x)) \vee \neg q(x,g(x))) \\ \Rightarrow & \{\neg p(x,f(x)) \vee r(x,f(x),g(x)), \neg p(x,f(x)) \vee \neg q(x,g(x))\} \\ \Rightarrow & \{\{\neg p(x,f(x)), r(x,f(x),g(x))\}, \{\neg p(x,f(x)), \neg q(x,g(x))\}\} \end{array}$$

Si ricordi che le variabili libere in una clausola si intendono quantificate universalmente; è una clausola A con le variabili libere $x_1,...,x_n$ sta per la sua "chiusura universale" $\forall x_1,...,\forall x_n A$. Denotiamo con $\forall A$ la chiusura universale di A.

La forma a clausole di un insieme S di formule è l'unione di tutte le clausole che si ottengono dalla trasformazione in forma a clausole di ciascuna formula in S. Si noti che quando si introducono simboli di Skolem, essi devono essere nuovi rispetto a tutto l'insieme S ed nessuno di essi deve essere utilizzato più volte, nella trasformazione di formule diverse.

Chiediamoci ora che relazione c'è tra A e la sua forma a clausole. Alcuni dei passi nella trasformazione conservano il significato delle formule:

1. Se $pre(A)$ è una forma prenessa di A, allora $A \leftrightarrow pre(A)$.
2. Se $FNC(A)$ è una forma normale congiuntiva di A, allora $\forall x_1...\forall x_n A \leftrightarrow \forall x_1...\forall x_n FNC(A)$.
3. Per quel che riguarda l'eliminazione dei quantificatori universali da una formula in forma normale di Skolem, osserviamo semplicemente che A sta per $\forall x_1...\forall x_n A$.

4. Quando trasformiamo una congiunzione in un insieme di clausole, stiamo ancora conservando il significato. Infatti $\forall x_1 ... \forall x_n (C_1 \wedge ... \wedge C_k) \quad \leftrightarrow \quad \forall x_1 ... \forall x_n C_1 \wedge ... \wedge \forall x_1 ... \forall x_n C_k$. Un insieme di formule S sta per la congiunzione delle formule in S, quindi $\forall x_1 ... \forall x_n (C_1 \wedge ... \wedge C_k)$ equivale a $\{\forall x_1 ... \forall x_n C_1, ..., \forall x_1 ... \forall x_n C_k\}$, cioè $\{C_1, ..., C_k\}$.

Ma la forma di Skolem di una formula non è logicamente equivalente alla formula stessa: in generale, se $sk(A)$ è una forma di Skolem di A

$$\not\models A \equiv sk(A)$$

Più precisamente:

$$\models sk(A) \rightarrow A, \qquad \text{ma} \qquad \not\models A \rightarrow sk(A)$$

Dimostriamo qui soltanto che in generale $\not\models A \rightarrow sk(A)$. Consideriamo infatti $A = \exists x p(x)$. In questo caso $sk(A) = p(c)$. Sia $\mathcal{M}$ l'interpretazione con $D = \{1, 2\}$, $\mathcal{M}(c) = 1$, $\mathcal{M}(p) = \{2\}$. Chiaramente: $\mathcal{M} \models \exists x p(x)$, ma $\mathcal{M} \not\models p(c)$.

Tuttavia, se A è soddisfacibile allora $sk(A)$ è soddisfacibile. Quindi la skolemizzazione conserva la *soddisfacibilità*. Poiché si ha anche che se $sk(A)$ è soddisfacibile allora A è soddisfacibile (banalmente, visto che $\models sk(A) \rightarrow A$), ne segue che A è soddisfacibile sse $sk(A)$ è soddisfacibile. Questo è quel che interessa per la risoluzione, che è un metodo di prova per refutazione: $S \models A$ sse $S \cup \{\neg A\}$ è insoddisfacibile, e questo vale se solo se la forma a clausole di $S \cup \{\neg A\}$ è insoddisfacibile.

4.5.3 La risoluzione in logica dei predicati

Anche nel caso della logica dei predicati, il metodo di risoluzione serve a controllare l'insoddisfacibilità di insiemi di clausole, derivando da esse la clausola vuota. La regola di risoluzione è in questo caso un po' più elaborata, ed utilizza l'unificazione:

$$\frac{C_1 \cup \{P\} \ ; \quad C_2 \cup \{\neg Q\}}{C_1\theta \cup C_2\theta} \qquad \text{se } \theta = mgu(P, Q)$$

$C_1\theta \cup C_2\theta$ è un *risolvente binario* di $C_1 \cup \{P\}$ e $C_2 \cup \{\neg Q\}$.

Ad esempio, quella che segue è un'applicazione della regola di risoluzione, con mgu $\theta = \{f(y)/x\}$:

$$\frac{p(x), q(x) \ ; \quad \neg p(f(y)), r(y)}{q(f(y)), r(y)} \qquad \theta = \{f(y)/x\}$$

La regola di risoluzione in logica dei predicati combina l'istanziazione di variabili (universali) con la regola di risoluzione proposizionale. L'applicazione precedente della regola si può considerare ottenuta dalla seguente:

$$\frac{\dfrac{p(x), q(x)}{p(f(y)), q(f(y))}\, IST \qquad \neg p(f(y)), r(y)}{q(f(y)), r(y)}$$

Applicando la regola di risoluzione, si può assumere che le due clausole genitrici non abbiano variabili in comune. Infatti, ogni volta che si utilizza una clausola in una derivazione mediante risoluzione, si rinominano le sue variabili (*standardizing apart*), ottenendo una *variante* della clausola.

Come esempio di applicazione del metodo di risoluzione nella logica dei predicati, consideriamo il seguente ragionamento:

Alcuni funzionari di dogana hanno perquisito tutti coloro che sono entrati nel paese, ad eccezione dei VIP.
Alcuni spacciatori di droga sono entrati nel paese e sono stati perquisiti solo da spacciatori di droga.
Nessuno spacciatore è un VIP.
Quindi alcuni funzionari sono spacciatori di droga.

Rappresentiamo il ragionamento utilizzando il linguaggio contenente i seguenti predicati:

$E(x)$	x è entrato nel paese
$V(x)$	x è un VIP
$P(x,y)$	y ha perquisito x
$F(x)$	x è un funzionario di dogana
$S(x)$	x è uno spacciatore di droga

Le formule che rappresentano le tre premesse e la conclusione del ragionamento sono quindi:

1. $\forall x(E(x) \wedge \neg V(x) \rightarrow \exists y(F(y) \wedge P(x,y)))$
2. $\exists x(S(x) \wedge E(x) \wedge \forall y(P(x,y) \rightarrow S(y)))$
3. $\forall x(S(x) \rightarrow \neg V(x))$
4. $\exists x(F(x) \wedge S(x))$

Trasformiamo ciascuna delle premesse e la negazione della conclusione in un insieme di clausole come segue:

$$\begin{array}{ll}
\forall x(E(x) \wedge \neg V(x) \rightarrow \exists y(F(y) \wedge P(x,y))) & \\
\quad \Rightarrow & \forall x \exists y(\neg(E(x) \wedge \neg V(x)) \vee (F(y) \wedge P(x,y))) \\
\quad \Rightarrow & \forall x(\neg E(x) \vee V(x) \vee (F(f(x)) \wedge P(x,f(x)))) \\
\quad \Rightarrow & (\neg E(x) \vee V(x) \vee F(f(x))) \wedge (\neg E(x) \vee V(x) \vee P(x,f(x))) \\
\quad \Rightarrow & \{\neg E(x) \vee V(x) \vee F(f(x)), \neg E(x) \vee V(x) \vee P(x,f(x))\} \\
\quad \Rightarrow & \{\{\neg E(x), V(x), F(f(x))\}, \{\neg E(x), V(x), P(x,f(x))\}\}
\end{array}$$

$$\begin{array}{ll}
\exists x(S(x) \wedge E(x) \wedge \forall y(P(x,y) \rightarrow S(y))) & \\
\quad \Rightarrow & \exists x \forall y(S(x) \wedge E(x) \wedge (\neg P(x,y) \vee S(y))) \\
\quad \Rightarrow & S(a) \wedge E(a) \wedge (\neg P(a,y) \vee S(y)) \\
\quad \Rightarrow & \{S(a), E(a), \neg P(a,y) \vee S(y)\} \\
\quad \Rightarrow & \{\{S(a)\}, \{E(a)\}, \{\neg P(a,y), S(y)\}\}
\end{array}$$

$$\begin{array}{rl}
\forall x(S(x) \rightarrow \neg V(x)) \Rightarrow & \forall x(S(x) \rightarrow \neg V(x)) \\
\Rightarrow & \{\neg S(x) \vee \neg V(x)\} \\
\Rightarrow & \{\{\neg S(x), \neg V(x)\}\}
\end{array}$$

$$\begin{array}{rl}
\neg \exists x(F(x) \wedge S(x)) \Rightarrow & \forall x \neg(F(x) \wedge S(x)) \\
\Rightarrow & \{\neg F(x) \vee \neg S(x)\} \\
\Rightarrow & \{\{\neg F(x), \neg S(x)\}\}
\end{array}$$

L'insieme di clausole che si ottiene è dunque

$$\begin{array}{rl}
S = & \{\{\neg E(x), V(x), F(f(x))\}, \{\neg E(x), V(x), P(x,f(x))\}, \\
& \{S(a)\}, \{E(a)\}, \{\neg P(a,y), S(y)\}, \\
& \{\neg S(x), \neg V(x)\}, \{\neg F(x), \neg S(x)\}\}
\end{array}$$

La derivazione della figura 4.8 è una dimostrazione per risoluzione di $S \vdash_{RES} \square$, dove, per motivi di spazio, abbiamo incolonnato i letterali di ciascuna clausola.

Quando le dimensioni di una derivazione crescono, è utile anche in questo caso fare ricorso alla scrittura in forma lineare. La derivazione precedente ad esempio può essere rappresentata come mostrato in figura 4.9.

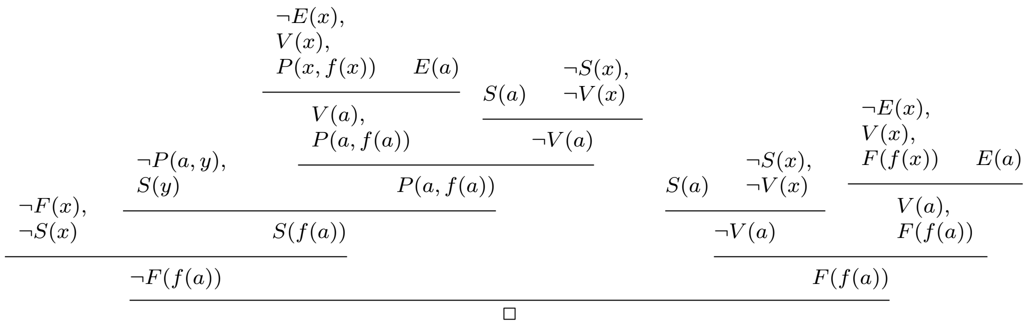


Figura 4.8: Una dimostrazione per risoluzione

1.	$\neg E(x),\ V(x),\ F(f(x))$	*in S*
2.	$\neg E(x),\ V(x),\ P(x, f(x))$	*in S*
3.	$S(a)$	*in S*
4.	$E(a)$	*in S*
5.	$\neg P(a, y),\ S(y)$	*in S*
6.	$\neg S(x),\ \neg V(x)$	*in S*
7.	$\neg F(x),\ \neg S(x)$	*in S*
8.	$V(a),\ P(a, f(a))$	$Res(2, 4), \{a/x\}$
9.	$\neg V(a)$	$Res(3, 6), \{a/x\}$
10.	$P(a, f(a))$	$Res(8, 9)$
11.	$S(f(a))$	$Res(5, 10), \{f(a)/y\}$
12.	$\neg F(f(a))$	$res(7, 11), \{f(a)/x\}$
13.	$V(a),\ F((f(a))$	$Res(1, 4), \{a/x\}$
14.	$F(f(a))$	$Res(9, 13)$
15.	$\Box$	$Res(12, 14)$

Figura 4.9: Rappresentazione lineare della dimostrazione per risoluzione della Figura 4.8

4.5.4 Fattorizzazione

La regola di risoluzione sopra formulata è in realtà una semplificazione della regola necessaria per garantire la completezza del metodo. Si consideri ad esempio l'insieme di clausole:

$$S = \{p(x) \vee p(y),\ \neg p(c) \vee \neg p(z)\}$$

Tale insieme è insoddisfacibile, ma ogni clausola derivabile mediante la regola di risoluzione, come è stata formulata fin qui, contiene due letterali. Quindi la clausola vuota non è derivabile.

Per riformulare adeguatamente la regola di risoluzione occorre definire la nozione di *fattore* di una clausola: se $C = C' \cup D$ e esiste un mgu θ per D (cioè una sostituzione che rende uguali tutti gli elementi di D) allora $C\theta$ è un fattore di C. Ad esempio $p(f(y)) \vee r(f(y), y)$ è un fattore di $p(x) \vee p(f(y)) \vee r(x, y)$. Si noti che una clausola è sempre un fattore (banale) di se stessa.

Un *risolvente* di due clausole C_1 e C_2 è allora un risolvente binario di un fattore di C_1 e di un fattore di C_2.

La regola di risoluzione è infine formulata come segue: se

- $C'_1 \cup \{P\}$ è un fattore di C_1,
- $C'_2 \cup \{\neg Q\}$ è un fattore di C_2,
- θ è un mgu di P e Q,

allora $C'_1\theta \cup C'_2\theta$ è derivabile per risoluzione da C_1 e C_2:

$$\frac{C_1\ ;\quad C_2}{C'_1\theta \cup C'_2\theta}$$

Possiamo allora, ad esempio, derivare la clausola vuota dall'insieme insoddisfacibile di clausole $\{p(x) \vee p(y), \neg p(c) \vee \neg p(z)\}$:

$$\frac{p(x) \vee p(y),\ \ \neg p(c) \vee \neg p(z)}{\Box}$$

Infatti $p(x)$ è un fattore di $p(x) \vee p(y)$ e $\neg p(c)$ è un fattore di $\neg p(c) \vee \neg p(z)$.

Il programma `prove` [7] è un'implementazione del metodo di risuluzione e alcuni suoi raffinamenti, che sono presentati nel seguito.

4.5.5 Raffinamenti della risoluzione

Il metodo di risoluzione fornisce vantaggi drastici rispetto ad altri sistemi di inferenza. Tuttavia l'applicazione non ristretta della risoluzione genera molte clausole inutili. Consideriamo come esempio la generazione esaustiva di risolventi in una "dimostrazione per saturazione di livelli", riportata in Figura 4.10: al primo livello appartengono tutte le clausole dell'insieme iniziale, al secondo i risolventi di coppie di clausole nel primo livello, ... al livello $n + 1$ i risolventi di una clausola del livello n con una clausola in un livello $k \leq n$. Nella colonna di destra della riga i sono indicate le clausole a cui viene applicata la regola di risoluzione per ottenere la riga i. Come si vede, anche in questo semplice caso, vengono generate molte clausole inutili prima della clausola vuota.

In questo paragrafo introduciamo alcune importanti strategie, o *raffinamenti* della risoluzione, che possono ridurre in modo drastico il processo di ricerca.

1.	$P \vee Q$	S
2.	$\neg P \vee R$	S
3.	$\neg Q \vee R$	S
4.	$\neg R$	S
5.	$Q \vee R$	$1, 2$
6.	$P \vee R$	$1, 3$
7.	$\neg P$	$2, 4$
8.	$\neg Q$	$3, 4$
9.	R	$3, 5$
10.	Q	$4, 5$
11.	R	$3, 6$
12	P	$4, 6$
13.	Q	$1, 7$
14.	R	$6, 7$
15.	P	$1, 8$
16.	R	$5, 8$
17.	$\square$	$4, 9$
18.	R	$3, 10$
19.	$\square$	$8, 10$
20.	$\square$	$4, 11$
21.	R	$2, 12$
22.	$\square$	$7, 12$
23.	R	$3, 13$
24.	$\square$	$8, 13$
25.	$\square$	$4, 14$
26.	R	$2, 15$
27.	$\square$	$7, 15$
28.	$\square$	$4, 16$
29.	$\square$	$4, 18$
30.	$\square$	$4, 21$
31.	$\square$	$4, 23$
32.	$\square$	$4, 26$

Figura 4.10: Dimostrazione per saturazione di livelli

La risoluzione lineare

Una derivazione per risoluzione lineare ha la struttura mostrata in Figura 4.11. La clausola C_0 viene detta clausola iniziale, le clausole C_i sono le clausole centrali e le B_i le clausole laterali. Si deve avere che per ogni $i = 0, ..., n-1$: $B_i \in S$, oppure $B_i = C_j$ per qualche $j < i$.

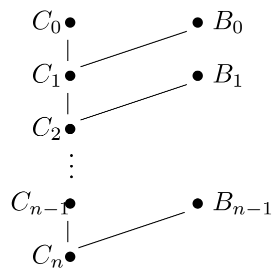


Figura 4.11: La struttura di una derivazione per risoluzione lineare.

Ad esempio, la seguente è una derivazione della clausola vuota da $S = \{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$, per risoluzione lineare.

$$\dfrac{\dfrac{\dfrac{\dfrac{P \vee Q \quad \neg P \vee Q}{Q} \quad P \vee \neg Q}{P} \quad \neg P \vee \neg Q}{\neg Q} \quad Q}{\square}$$

Con la risoluzione lineare, si evita la ridondanza generata dalla risoluzione di conclusioni intermedie con altre conclusioni intermedie: il *focus* è su S e sugli antenati della clausola centrale.

La risoluzione lineare è completa (sempre rispetto alla refutazione): se un insieme S di clausole è insoddisfacibile, allora esiste una refutazione di S per risoluzione lineare.

Nell'impostare una risoluzione lineare può essere di cruciale importanza la scelta della clausola iniziale. La completezza infatti non garantisce che esiste una refutazione lineare *qualsiasi sia la scelta della clausola iniziale*. È importante allora tener presente il seguente risultato: se $S = S' \cup \{C\}$ è un insieme di clausole insoddisfacibile e se S' è soddisfacibile, allora esiste una refutazione lineare di S con C come clausola iniziale.

Risoluzione con clausole unitarie (*unit resolution*)

Una clausola è *unitaria* se essa contiene un solo letterale. Secondo la strategia di risoluzione con clausole unitarie, ogni risolvente è unitario, cioè almeno una delle due clausole genitrici è una clausola unitaria.

Ad esempio, la seguente è una refutazione con clausole unitarie di $S = \{P \vee Q, \neg P \vee R, \neg Q \vee R, \neg R\}$.

1.	$P \vee Q$	S
2.	$\neg P \vee R$	S
3.	$\neg Q \vee R$	S
4.	$\neg R$	S
5.	$\neg P$	$2, 4$
6.	$\neg Q$	$3, 4$
7.	Q	$1, 5$
8.	P	$1, 6$
9.	R	$3, 7$
10.	$\square$	$6, 7$
11.	R	$2, 8$
12.	$\square$	$5, 8$

Questa strategia porta a concludere rapidamente la derivazione (quando possibile), in quanto si ottengono via via clausole con un numero sempre minore di letterali.

La risoluzione con clausole unitarie è efficiente ma non completa: Ad esempio, $\{P \vee Q, \neg P \vee Q, P \vee \neg Q, \neg P \vee \neg Q\}$ è insoddisfacibile ma non ne esiste una refutazione con clausole unitarie.

Tuttavia è completa per una classe particolare di insiemi di clausole, quelli contenenti soltanto *clausole Horn*, dove:

Definizione 4.5.1 *Una clausola Horn è una clausola contenente al massimo un letterale positivo.*

Input Resolution

Questa strategia vincola a generare soltanto *input risolventi*, cioè risolventi in cui almeno una delle due clausole genitrici sia un elemento dell'insieme iniziale (*input database*).

Ad esempio, la derivazione mostrata in figura 4.12 è una refutazione di

$$S = \{p(x, y) \vee q(y, x), \neg p(f(z), z) \vee r(z), \neg q(c, f(c)) \vee r(c), \neg r(x)\}$$

per *input resolution*.

Input risoluzione e risoluzione con clausole unitarie sono equivalenti: esiste una refutazione unitaria da S sse esiste una input-refutazione di S. Di conseguenza, l'input risoluzione non è completa in generale, ma è completa per clausole Horn.

4.5.6 Risoluzione SLD e Programmazione Logica

In questo paragrafo presentiamo i fondamenti su cui si basano i linguaggi di programmazione logica, come il Prolog.

Le clausole Horn, cioè quelle clausole che contengono al massimo un letterale positivo, possono essere classificate in quattro categorie:

Clausole definite: clausole che contengono (esattamente) un letterale positivo. Queste si distinguono a loro volta in:

- **Fatti:** clausole che hanno esattamente un letterale positivo e nessun letterale negativo: A, dove A è un atomo.
- **Regole:** clausole che hanno un letterale positivo e almeno un letterale negativo: $A \vee \neg B_1 \vee ... \vee \neg B_n$. Una formula di questa forma è equivalente a

$$B_1 \wedge ... \wedge B_n \rightarrow A$$

1.	$p(x,y) \vee q(y,x)$	S
2.	$\neg p(f(z),z) \vee r(z)$	S
3.	$\neg q(c,f(c)) \vee r(c)$	S
4.	$\neg r(x)$	S
5.	$q(z,f(z)) \vee r(z)$	$1,2$
6.	$p(f(c),c) \vee r(c)$	$1,3$
7.	$\neg p(f(z),z)$	$2,4$
8.	$\neg q(c,f(c))$	$3,4$
9.	$r(c)$	$3,5$
10.	$q(z,f(z))$	$4,5$
11.	$r(c)$	$2,6$
12.	$p(f(c),c)$	$4,6$
13.	$q(z,f(z))$	$1,7$
14.	$p(f(c),c)$	$1,8$
15.	$\square$	$9,4$

Figura 4.12: Una refutazione per *input resolution*

Nella programmazione logica una clausola di questa forma viene abitualmente scritta utilizzando la seguente notazione

$$A \text{ :- } B_1, ..., B_n$$

(che si legge "A se B_1 e ... e B_n"). A si chiama la *testa* della regola e B_1 e ... e B_n è il *corpo* della regola.

Clausole negative: clausole contenenti soltanto letterali negativi. Queste, a loro volta, si suddividono in:

Obiettivi (goal): clausole negative contenenti almeno un letterale: $\neg B_1 \vee ... \vee \neg B_n$. Una formula di questa forma è equivalente a $\neg(B_1 \wedge ... \wedge B_n)$ o anche $B_1 \wedge ... \wedge B_n \rightarrow \bot$ e la notazione utilizzata nella programmazione logica è

$$?\text{- } B_1, ..., B_n$$

Clausola vuota. La clausola vuota è una clausola negativa, in quanto non contiene alcun letterale positivo.

Un *programma logico* è un insieme finito di clausole definite. Si noti che ogni programma logico è soddisfacibile, in quanto è sufficiente che $(\mathcal{M}, \sigma) \models A$ per ogni formula atomica A perché $(\mathcal{M}, \sigma)$ sia un modello del programma logico: esso infatti è un modello di tutti i fatti e di tutte le teste delle regole del programma.

Ad esempio, il seguente insieme di clausole P_{ama} costituisce un programma logico, che "definisce" la relazione `ama`:

```
ama(maria,birra).
ama(maria,vino).
ama(carlo,maria).
ama(carlo,vino).
ama(X,Y) :- ama(X,Z), ama(Y,Z).
```

Le prime quattro clausole sono fatti, l'ultima è una regola. Si noti che nella notazione della programmazione logica, gli identificatori che iniziano con una lettera maiuscola sono variabili, quelli che iniziano con lettera minuscola sono costanti, simboli funzionali o simboli di predicato,

Si noti che la regola `ama(X,Y) :- ama(X,Z), ama(Y,Z)` corrisponde alla clausola $ama(x,z) \wedge ama(y,z) \rightarrow ama(x,y)$, che sta per

$$\forall x\, \forall y\, \forall z\, (ama(x,z) \wedge ama(y,z) \rightarrow ama(x,y))$$

Quest'ultima formula è logicamente equivalente a

$$\forall x\, \forall y\, (\exists z\, (ama(x,z) \wedge ama(y,z)) \rightarrow ama(x,y))$$

La regola si può dunque leggere così: "X ama Y se esiste Z tale che X ama Z e Y ama Z". In generale, le variabili che occorrono nel corpo di una clausola e non nella testa sono quantificate esistenzialmente nel corpo della clausola.

Un programma logico P viene eseguito a fronte di un *goal* ?- G. L'esecuzione del programma consiste nel tentativo di dimostrare che $P \cup \{\neg G\}$ è insoddisfacibile, cioè di dimostrare che $P \models G$.

Si consideri ad esempio il goal ?- $G =$ `?- ama(maria,carlo)` e il programma logico P_{ama} precedente. Il goal è equivalente alla clausola negativa $\neg\, ama(maria, carlo)$ e dimostrare che $P_{ama} \cup \{\neg\, ama(maria, carlo\}$ è insoddisfacibile equivale a dimostrare che

$$\begin{array}{c}\{ama(maria, birra), ama(maria, vino), ama(carlo, maria), ama(carlo, vino), \\ \forall x\, \forall y\, \forall z\, (ama(x,z) \wedge ama(y,z) \rightarrow ama(x,y))\} \ \models \ ama(maria, carlo)\end{array}$$

Se il goal contiene variabili, esse vanno lette esistenzialmente: se ?- $G =$?- $A_1, ..., A_n$ e le variabili in G sono $x_1, ..., x_n$, allora l'esecuzione del programma logico P a fronte di ?- G è un tentativo di dimostrare che $\forall\, P \cup \{\forall x_1 ... \forall x_n \neg (A_1 \wedge ... \wedge A_n)$ è insoddisfacibile, dove $\forall\, P = \{\forall\, C \mid C \in P\}$ ($\forall\, C$ è la chiusura universale di C). Ciò equivale a dimostrare che $\forall\, P \models \neg \forall x_1 ... \forall x_n \neg (A_1 \wedge ... \wedge A_n)$, cioè che $\forall\, P \models \exists x_1 ... \exists x_n\, (A_1 \wedge ... \wedge A_n)$.

Un sistema di programmazione logica, dato un programma P ed un goal ?- G contenente le variabili $x_1, ..., x_n$, cercherà di dimostrare dunque la derivabilità di $\exists x_1 ... \exists x_n\, G$ da $\forall\, P$. Poiché il metodo di dimostrazione utilizzato è "costruttivo", se il tentativo ha successo il sistema inoltre fornirà esempi di termini $t_1, ..., t_n$ tali che $\forall\, G[t_1/x_1, ..., t_n/x_n]$ è derivabile da P.

Ad esempio, se il programma logico P_{ama} viene eseguito a fronte del goal ?- $ama(maria, X)$, il sistema può fornire come "soluzione" $X = birra$, oppure (eventualmente su richiesta di avere una ulteriore soluzione) $X = vino$, oppure $X = carlo$.

La risoluzione SLD

I sistemi di programmazione logica utilizzano una particolare strategia di risoluzione, chiamata risoluzione SLD (risoluzione Lineare con funzione di Selezione per clausole Definite). Questa combina la risoluzione lineare con una generalizzazione della risoluzione ordinata.

La strategia SLD dipende da una *regola di calcolo*: una regola di calcolo è una funzione R che, applicata a un goal ?- G, riporta un atomo di G (l'atomo "selezionato" in G):

$$R(\text{?- } B_1, ..., B_n) = B_i$$

La regola di risoluzione SLD con regola di calcolo R risolve un goal ?- G ed una clausola di programma (una clausola definita), risolvendo sul letterale selezionato da R in G e derivando un nuovo goal:

$$\frac{?\text{-}\ A_1,...,A_j,...,A_m \qquad\qquad A \ \text{:-}\ B_1,...,B_k}{?\text{-}\ (A_1,...,A_{j-1},B_1,...,B_k,A_{j+1},...,A_m)\theta}$$

dove $R(?\text{-}\ A_1,...,A_j,...,A_m) = A_j$ e θ è un mgu di A_j e A.

Ora, sia P un programma logico, ?- G un goal e R una regola di calcolo. Una derivazione SLD da $P \cup \{?\text{-}\ G\}$ con R è costituita da:

- una sequenza di goal $?\text{-}\ G_0 = ?\text{-}\ G, ?\text{-}\ G_1, ?\text{-}\ G_2, ...$;
- una sequenza $C_0, C_1, C_2, ...$ di varianti di clausole in P (clausole di P, con variabili rinominate);
- una sequenza $\theta_0, \theta_1, \theta_2, ...$ di sostituzioni.

tali che ogni ?- G_{i+1} deriva da ?- G_i e C_i con un'applicazione della regola di risoluzione SLD, con R come regola di calcolo e θ_i come mgu. Una derivazione SLD si può rappresentare come segue:

$$\dfrac{\dfrac{?\text{-}\ G \quad C_0}{?\text{-}\ G_1}\ (\theta_0) \qquad C_1}{?\text{-}\ G_2}\ (\theta_1)$$
$$\vdots$$

Una *dimostrazione* SLD di G da P è una derivazione SLD finita da $P \cup \{?\text{-}\ G\}$ in cui l'ultimo goal è la clausola vuota, $\Box$.

Ad esempio, sia $P = \{q, p \ \text{:-}\ q\}$ e $?\text{-}\ G = ?\text{-}\ q, p$. Allora, se la regola R seleziona il primo letterale della clausola, una dimostrazione SLD di G da P con R è la seguente:

$$\dfrac{\dfrac{\dfrac{?\text{-}\ q,p \quad q}{?\text{-}\ p} \qquad p \ \text{:-}\ q}{?\text{-}\ q} \qquad q}{\Box}$$

Se R' seleziona l'ultimo letterale della clausola, allora la seguente è una dimostrazione SLD di G da P con R':

$$\dfrac{\dfrac{?\text{-}\ q,p \quad p \ \text{:-}\ q}{?\text{-}\ q} \qquad q}{\Box}$$

Supponiamo ora di avere la seguente “base di conoscenza”:

$$\{padre(aldo, piero), \quad padre(piero, ugo),$$
$$\forall x \forall y (\exists z (padre(x, z) \land padre(z, y)) \rightarrow nonno(x, y))\}$$

La forma a clausole di questa teoria è il seguente programma logico P:

```
padre(aldo,piero).
padre(piero,ugo).
nonno(X,Y) :- padre(X,Z), padre(Z,Y).
```

Se vogliamo sapere se dalla base di conoscenza è derivabile $\exists x\, nonno(x, ugo)$, possiamo cercare una derivazione SLD di $\Box$ da $P \cup \{?\text{-}\ nonno(X, ugo)\}$. La derivazione fornirà anche, eventualmente, informazioni su chi sia un nonno di ugo.

Se utilizziamo la regola di calcolo "scelta del primo atomo", possiamo costruire la seguente dimostrazione SLD (dove, per ragioni di spazio, scriviamo p e n al posto di *padre* e *nonno*, rispettivamente):

$$\cfrac{\cfrac{\cfrac{?\text{-}\ n(X, ugo) \quad n(X_1, Y) \text{ :- } p(X_1, Z), p(Z, Y)}{?\text{-}\ p(X, Z), p(Z, ugo)} \qquad p(aldo, piero)}{?\text{-}\ p(piero, ugo)} \qquad p(piero, ugo)}{\Box}$$

La sostituzione applicata nella prima inferenza è $\theta_0 = \{X/X_1, ugo/Y\}$, quella applicata per ottenere ?- $p(piero, ugo)$ è $\theta_1 = \{aldo/X, piero/Z\}$ e l'ultima è la sostituzione vuota $\theta_2 = \emptyset$. Sappiamo duqne che $\exists x\, nonno(x, ugo)$ è derivabile dalla base di consocenza considerata.

Ma sappiamo in realtà anche qualcosa di più: possiamo infatti trarre informazioni utili dalle sostituzioni applicate nella SLD-refutazione. Componendo le due sostituzioni applicate nella derivazione otteniamo:

$$\theta = \theta_0 \circ \theta_1 \circ \theta_2 = \{aldo/X_1, ugo/Y, aldo/X, piero/Z\}$$

Quel che interessa qui è il valore sostituito ad X, che è l'unica variabile che occorre libera nel goal iniziale: $X = aldo$ è una "risposta" al problema risolto dal sistema. Quindi la composizione delle sostituzioni utilizzate nella SLD-refutazione fornisce una risposta alla domanda: "chi è un nonno di ugo?". In altri termini, la soluzione che sarà fornita in questo caso da un sistema di programmazione logica informerà non soltanto del fatto che $P \models \exists x\, nonno(x, ugo)$, ma anche che $P \models nonno(X, ugo)\theta$ cioè $P \models nonno(aldo, ugo)$.

Un sistema di programmazione logica serve dunque per calcolare dei valori e non solo per rispondere "sì è derivabile" o "no, non è derivabile".

Completezza e correttezza della risoluzione SLD

La completezza della risoluzione SLD è indipendente dalla regola di calcolo: se P è un programma logico, G un goal e R una (qualsiasi) regola di calcolo, e se $P \cup \{?\text{-}\ G\}$ è insoddisfacibile, allora esiste una SLD-refutazione di $P \cup \{?\text{-}\ G\}$ con R. La regola di calcolo può influenzare l'efficienza del sistema ma non può compromettere la sua completezza.

La correttezza della risoluzione SLD stabilisce non soltanto che se un insieme è refutabile allora esso è inconsistente, ma anche che la "soluzione" (la sostituzione) fornita dal sistema è effettivamente corretta, nel senso seguente:

Definizione 4.5.2 *Sia P un programma logico, ?- $G =$?- $A_1, ..., A_n$ e siano $X = \{x_1, ..., x_k\}$ tutte le variabili in G.*

Una sostituzione θ è una sostituzione di risposta corretta *(*answer substitution*) per $P \cup \{$?- $G\}$ se:*

- *$\forall P \models \forall((A_1 \wedge ... \wedge A_n)\theta)$;*
- *per ogni $t/y \in \theta$, $y \in X$.*

La seconda clausola nella definizione precedente serve ad eliminare dalla risposta le informazioni relative a variabili "irrilevanti", che non occorrono nel goal ?- G.

Supponiamo ora che $P \cup \{?\text{-}\ G\}$ sia insoddisfacibile. Sia R una regola di calcolo e sia $\theta_0, \theta_1, ..., \theta_n$ la sequenza di mgu usata in una SLD-refutazione di $P \cup \{?\text{-}\ G\}$. La sostituzione di risposta calcolata da R è la restrizione di $\theta_0 \circ \theta_1 \circ ... \circ \theta_n$ alle variabili di G (ottenuta cioè eliminando tutte le coppie t/x tali che x non occorre in G).

Teorema 4.1 (Correttezza della risoluzione SLD) *Siano P un programma logico, ?- G un goal e R una regola di calcolo. Ogni sostituzione di risposta calcolata da R per $P \cup \{$?- $G\}$ è una sostituzione di risposta corretta.*

Consideriamo ancora come esempio il seguente programma logico P_{col}:

```
(1) colore(X,Y) :- vicino(X,Z), colore(Z,Y)
(2) colore(c1,blu)
(3) colore(c2,rosso)
(4) vicino(c,c1)
(5) vicino(c,c2)
```

La seguente derivazione è una dimostrazione di $colore(c,Y)$ da P_{col}, dove, per motivi di spazio non abbiamo riportato per intero la regola (1) ma soltanto il suo numero:

$$\cfrac{\cfrac{\cfrac{\text{?- } colore(c,Y) \quad (1)}{\text{?- } vicino(c,Z), colore(Z,Y)} \quad vicino(c,c1)}{\text{?- } colore(c1,Y)} \qquad colore(c1,blu)}{\Box}$$

Le sostituzioni applicate sono, nell'ordine: $\theta_0 = \{c/X\}$, $\theta_1 = \{c1/Z\}$ $\theta_2 = \{blu/Y\}$. La loro composizione è:

$$\theta = \theta_0 \circ \theta_1 \circ \theta_2 = \{c/X, c1/Z, blu/Y\}$$

La restrizione di θ alle variabili che occorrono in $colore(c,Y)$ è $\{blu/Y\}$. Questa è dunque la sostituzione di risposta estratta dalla derivazione. Si noti che una dimostrazione diversa avrebbe potuto fornire come risposta la sostituzione $\{rosso/Y\}$.

Si noti che, se un programma logico potesse contenere anche clausole "disgiuntive", cioè clausole in cui occorre più di un letterale positivo, non sempre esisterebbe una sostituzione di risposta corretta. Consideriamo ad esempio il programma disgiuntivo

$$P = \{colore(c1, blu) \vee colore(c1, rosso)\}$$

ed il goal

$$\text{?- } G = \text{?- } colore(c1, X)$$

$P \models \exists X colore(c1, X)$, ma non esiste alcuna sostituzione θ tale che $P \models colore(c1, X)\theta$. La derivazione seguente, ad esempio, dimostra che $P \cup \{\text{?- } G\}$ è insoddisfacibile:

$$\cfrac{\cfrac{\neg colore(c1,X) \quad colore(c1,blu) \vee colore(c1,rosso)}{colore(c1,rosso)} \qquad \neg colore(c1,X)}{\Box}$$

Le sostituzioni applicate sono, nell'ordine: $\theta_0 = \{blu/X\}$ e $\theta_1 = \{rosso/X\}$. La composizione di θ_0 e θ_1 è uguale a θ_0 stessa, e questa non è una sostituzione di risposta corretta. Infatti $colore(c1, blu)$ non è una conseguenza logica di P, come non lo è $colore(c1, rosso)$.

Sistemi di programmazione logica

Un sistema di programmazione logica è un sistema di derivazione basato sulla risoluzione SLD, che dipende dalla scelta di:

- una regola di calcolo;

- una strategia di ricerca della refutazione, cioè una strategia per gestire la scelta delle clausole di programma da utilizzare.

Come abbiamo visto, la scelta della regola di calcolo può influenzare drasticamente la dimensione e la struttura dell'albero di derivazione, ma non la completezza del sistema. Al contrario, la strategia di ricerca può compromettere la completezza.

Il linguaggio Prolog è un sistema di programmazione logica in cui:

- il letterale selezionato dalla regola di calcolo è il primo letterale del goal: $R(?\text{-}\ A_1, ..., A_n) = A_1$;

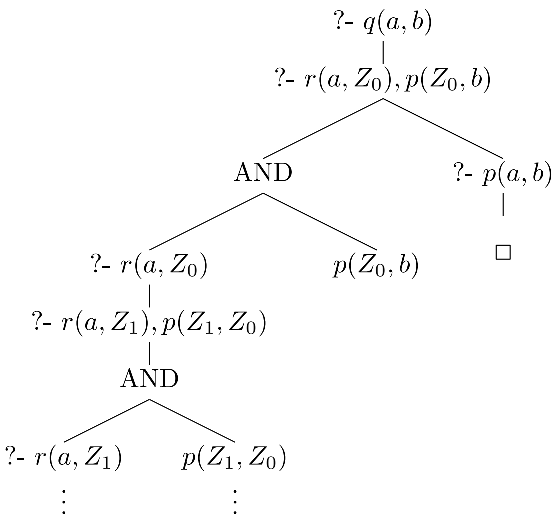
- la strategia di ricerca adottata è la ricerca in profondità.

La ricerca in profondità è una strategia efficiente, ma determina l'incompletezza del sistema.

Consideriamo ad esempio il seguente programma logico:

```
p(a,b).
r(X,Y) :- r(X,Z), p(Z,Y).
r(X,Y) :- p(X,Y).
```

ed il goal ?- $r(a, b)$. Il goal può essere risolto con due clausole di programma: la seconda o la terza. Ciò genera un'alternativa. Nel primo caso si ottiene il nuovo goal ?- $r(a, Z_0), p(Z_0, b)$ – considerando una variante della seconda clausola di programma; nel secondo caso si ottiene il goal ?- $p(a, b)$. Quest'ultimo, risolto con la prima clausola di programma, produce la clausola vuota ed il processo termina. Ma se invece si sceglie la prima alternativa, e la regola di calcolo produce il primo letterale, siamo di nuovo di fronte a una scelta: utilizzare la seconda o la terza clausola di programma. Possiamo rappresentare lo spazio di ricerca mediante il seguente albero, dove i nodi marcati con "AND" indicano che per trovare una soluzione per il genitore, occorre trovare soluzioni per tutti i figli. Gli altri sono invece "nodi OR": è sufficiente trovare una soluzione per un figlio, per avere una soluzione per il padre.



In ogni ramificazione OR, il figlio di sinistra corrisponde alla scelta della seconda clausola di programma, quello di destra alla terza. Come si vede, se la ricerca prosegue in profondità lungo il ramo di sinistra, non termina, e non viene dunque trovata la soluzione immediata sul ramo di destra.

Il linguaggio Prolog costituisce un sistema di programmazione logica incompleto, come si è appena visto, ma anche non corretto. Infatti, sempre per motivi di efficienza, l'algoritmo di unificazione è implementato senza effettuare l'*occur check*, che è un test piuttosto costoso.

4.6 Esercizi

1. Dimostrare mediante tableaux:
   (a) $p \to (q \to r), p \vee s \vdash (q \to r) \vee s$
   (b) $p, q \to \neg p \vdash \neg q$
   (c) $\neg q \to q, p \to q \vdash q$
   (d) $(p \to q) \to p \vdash p$
   (e) $p \to q, q \to p, p \vee q \vdash p \wedge q$
   (f) $p \vee q, p \vee (q \wedge r) \vdash p \vee r$
2. Dimostrare, utilizzando i tableaux per la ricerca di (contro)modelli, che:
   (a) $p \to q \not\models \neg p \to \neg q$
   (b) $p \vee q \not\models p$
   (c) $p \wedge (q \vee r) \not\models \neg(q \to r)$
3. È vero o falso che $\models (p \to q) \vee (q \to p)$? Risolvere il problema utilizzando i tableaux.
4. Formulare regole di espansione per $A \equiv B$ e $\neg(A \equiv B)$, che siano "regole derivate" nel sistema dei tableaux.
5. Dimostrare la validità delle leggi di De Morgan, utilizzando il metodo dei tableaux con la regola per $\equiv$ formulata al punto precedente.
6. Dimostrare mediante il metodo dei tableaux che le formule a pagina 15 sono tautologie e che le coppie di formule introdotte a pagina 16 sono logicamente equivalenti.
7. Si mostrino l'esecuzione, stadio per stadio, ed il risultato dell'algoritmo di unificazione di Robinson sulle seguenti coppie di espressioni:
   (a) $A = p(x)$, $B = p(f(y))$
   (b) $A = p(x, f(a))$, $B = p(c, f(x))$
   (c) $A = p(x, f(a))$, $B = p(c, f(y))$
   (d) $A = p(x, x)$, $B = p(f(y), y)$
   (e) $A = p(x, f(c), x)$, $B = p(f(y), y, z)$
   (f) $A = p(x, f(c), x)$, $B = p(f(y), y, c)$
8. Dimostrare per risoluzione la validità dei seguenti ragionamenti:
   (a) Il professore è contento se a tutti i suoi studenti piace la matematica. Quindi è contento se non ha studenti.
   (b) Ad alcuni pazienti piacciono tutti i medici. A nessun paziente piace alcun ciarlatano. Quindi nessun medico è un ciarlatano.
   (c) Chiunque risparmia denaro guadagna interesse. Dunque se non ci fosse interesse nessuno risparmierebbe denaro.
   Utilizzare un linguaggio con i predicati seguenti:

$R(x, y)$	x risparmia y
$D(x)$	x è denaro
$I(x)$	x è interesse
$G(x, y)$	x guadagna y.

(d) Ogni cittadino è un risparmiatore e un lavoratore. Esiste un cittadino onesto. Quindi esiste un risparmiatore onesto.

(e) Se un corso è facile, alcuni studenti sono felici. Se in un corso si parla di logica, nessuno studente è felice. Quindi nessun corso in cui si parla di logica è facile.

(f) Antonio è padre di Biagio. Biagio è padre di Carlo. Carlo è padre di Dario. Il padre di qualcuno è un suo antenato. Ogni antenato del padre di qualcuno è anche un antenato di quest'ultimo. Quindi Antonio è antenato di Dario.

(g) Se una relazione R è simmetrica, transitiva e totale, allora è anche riflessiva.

9. Dimostrare mediante risoluzione la validità delle formule a pagina 42.

10. Si costruiscano dimostrazioni per risoluzione che dimostrino la correttezza dei ragionamenti corretti tra quelli dell'esercizio O a pagina 67.

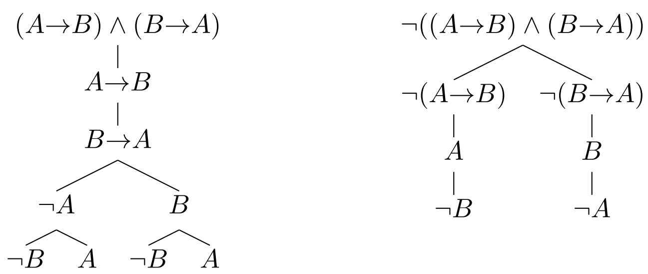
11. Per ciascuno degli esercizi dei gruppi 8–10, determinare se l'insieme di clausole ottenute è un insieme di clausole Horn oppure no. Determinare, di conseguenza, quali strategie di risoluzione, tra quelle esaminate in questo capitolo, possono essere adottate senza perdere la completezza. Determinare inoltre se l'insieme di clausole ottenuto può essere visto come un programma logico e un goal. In caso positivo, risolvere il problema mediante risoluzione SLD e calcolare la sostituzione di risposta corrispondente.

12. Fornire una dimostrazione SLD di $ama(maria, X)$ dal programma logico P_{ama} riportato a pagina 99 che fornisca la soluzione $X = carlo$ (cioè la sostituzione di risposta $\{carlo/X\}$).

4.7 Soluzione di alcuni esercizi

(1),(2),(3) Per controllare la soluzione di questi esercizi si può utilizzare il programma `tableau` che si può scaricare da [7].

(4) Consideriamo la definizione di $A \equiv B$ come $(A \to B) \wedge (B \to A)$ ed i seguenti tableaux:



Poiché il secondo e terzo ramo del tableau a sinistra sono chiusi, essi dimostrano che le espansioni:

$$\begin{array}{cc} \multicolumn{2}{c}{A \equiv B} \\ \hline \neg A & B \\ \neg B & A \end{array} \qquad \begin{array}{cc} \multicolumn{2}{c}{\neg(A \equiv B)} \\ \hline A & B \\ \neg B & \neg A \end{array}$$

sono regole derivate.

(8a) Possiamo rappresentare il ragionamento analizzandolo come: "per ogni professore x, se a tutti gli studenti di x piace la matematica ($pmat$), allora x è contento; quindi per ogni professore x, se x non ha studenti, allora x è contento":

$$\forall x(prof(x) \to (\forall y(studente(y,x) \to pmat(y)) \to contento(x)))$$
$$\models \forall x(prof(x) \to (\neg\exists y\, studente(y,x) \to contento(x)))$$

Premessa e negazione della conclusione sono trasformate in forma a clausole come segue:

$$\begin{array}{ll}
\multicolumn{2}{l}{\forall x(prof(x) \to (\forall y(studente(y,x) \to pmat(y)) \to contento(x)))} \\
\Rightarrow & \forall x(prof(x) \to \exists y((studente(y,x) \to pmat(y)) \to contento(x))) \\
\Rightarrow & \forall x \exists y(prof(x) \to ((studente(y,x) \to pmat(y)) \to contento(x))) \\
\Rightarrow & \forall x(prof(x) \to ((studente(f(x),x) \to pmat(f(x))) \to contento(x))) \\
\Rightarrow & prof(x) \to ((studente(f(x),x) \to pmat(f(x))) \to contento(x)) \\
\Rightarrow & \neg prof(x) \vee (\neg(studente(f(x),x) \to pmat(f(x))) \vee contento(x)) \\
\Rightarrow & \neg prof(x) \vee (studente(f(x),x) \wedge \neg pmat(f(x))) \vee contento(x) \\
\Rightarrow & (\neg prof(x) \vee studente(f(x),x)) \wedge (\neg prof(x) \vee \neg pmat(f(x))) \vee contento(x) \\
\Rightarrow & (\neg prof(x) \vee studente(f(x),x) \vee contento(x)) \\
 & \quad \wedge(\neg prof(x) \vee \neg pmat(f(x)) \vee contento(x)) \\
\Rightarrow & \{\neg prof(x) \vee studente(f(x),x) \vee contento(x), \\
 & \quad \neg prof(x) \vee \neg pmat(f(x)) \vee contento(x)\}
\end{array}$$

$$\begin{array}{ll}
\multicolumn{2}{l}{\neg\forall x(prof(x) \to (\neg\exists y\, studente(y,x) \to contento(x)))} \\
\Rightarrow & \exists x \neg(prof(x) \to (\forall y \neg studente(y,x) \to contento(x))) \\
\Rightarrow & \exists x \neg(prof(x) \to \exists y(\neg studente(y,x) \to contento(x))) \\
\Rightarrow & \exists x \neg\exists y(prof(x) \to (\neg studente(y,x) \to contento(x))) \\
\Rightarrow & \exists x \forall y \neg(prof(x) \to (\neg studente(y,x) \to contento(x))) \\
\Rightarrow & \forall y \neg(prof(c) \to (\neg studente(y,c) \to contento(c))) \\
\Rightarrow & \neg(prof(c) \to (\neg studente(y,c) \to contento(c))) \\
\Rightarrow & prof(c) \wedge \neg(\neg studente(y,c) \to contento(c)) \\
\Rightarrow & prof(c) \wedge (\neg studente(y,c) \wedge \neg contento(c)) \\
\Rightarrow & \{prof(c),\ \neg studente(y,c),\ \neg contento(c)\}
\end{array}$$

Ed una refutazione delle clausole dell'insieme S ottenuto è la seguente (a destra di ciascun risolvente sono indicate le clausole genitrici e la sostituzione applicata):

$$\begin{array}{rll}
1. & \neg prof(x) \vee studente(f(x),x) \vee contento(x) & \textit{in } S \\
2. & \neg prof(x) \vee \neg pmat(f(x)) \vee contento(x) & \textit{in } S \\
3. & prof(c) & \textit{in } S \\
4. & \neg studente(y,c) & \textit{in } S \\
5. & \neg contento(c) & \textit{in } S \\
6. & studente(f(c),c) \vee contento(c) & 1,3\ \{c/x\} \\
7. & contento(c) & 4,6\ \{f(c)/y\} \\
8. & \square & 5,7
\end{array}$$

(8b) Si deve mostrare che:

$$\begin{array}{l}
\exists x(paz(x) \wedge \forall y(med(y) \to piace(x,y))), \\
\neg\exists x(paz(x) \wedge \exists y(ciar(y) \wedge piace(x,y))) \models \forall x(med(x) \to \neg ciarl(x))
\end{array}$$

Le forme a clausole della prima e seconda premessa sono, rispettivamente:

$$\{paz(c),\ \neg med(y) \vee piace(c,y)\}$$
$$\{\neg paz(x) \vee \neg ciar(y) \vee \neg piace(x,y)\}$$

Dalla negazione della conclusione si ottiene:

$$\{med(d),\ ciarl(d)\}$$

Una refutazione per risoluzione dell'unione S dei tre insiemi è la seguente:

1.	$paz(c)$	*in* S
2.	$\neg med(y_0) \vee piace(c, y_0)$	*in* S
3.	$\neg paz(x) \vee \neg ciar(y) \vee \neg piace(x, y)$	*in* S
4.	$med(d)$	*in* S
5.	$ciarl(d)$	*in* S
6.	$\neg ciar(y) \vee \neg piace(c, y)$	$1, 3\ \{c/x\}$
7.	$\neg piace(c, d)$	$5, 6\ \{d/y\}$
8.	$\neg med(d)$	$2, 7\ \{d/y_0\}$
9.	$\Box$	$4, 8$

Si noti che l'insieme di clausole S è un insieme di clausole Horn, che può essere visto come un programma logico e un goal:

```
paz(c).
piace(c,y) :- med(y).
med(d).
ciarl(d).
?- paz(x), ciar(y), piace(x,y)
```

Dal programma logico è dunque dimostrabile che esiste un paziente al quale piace un ciarlatano. La dimostrazione sopra riportata segue in effetti la strategia SLD, con la regola di calcolo "scelta del primo atomo". La sostituzione di risposta calcolata sulla base di tale dimostrazione è $\{c/x, d/y\}$.

Bibliografia

[1] G. Boolos and R. Jeffrey. *Computability and Logic*. Cambridge University Press, 1974.

[2] C.L. Chang and Lee R.C.T. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, 1973. trad. it. Tecniche Nuove.

[3] M. Fitting. *First-Order Logic and Automated Theorem Proving. Second Edition*. Springer, 1996.

[4] S.C. Kleene. *Introduction to Metamathematics*. North Holland, 1952.

[5] E. Mendelson. *Introduction to Mathematical Logic*. Van Nostrand, 1964. Trad. it. Boringhieri.

[6] M. Mondadori and M. D'Agostino. *Logica*. Bruno Mondadori, 1997.

[7] Programmi di utilità e sorgenti OCaml di alcuni algoritmi di logica. `http://cialdea.dia.uniroma3.it/teaching/logica/materiale.php`.

[8] H. Rogers. *Theory of Recursive Functions and Effective Computability*. MIT Press, 1987.

[9] J.R. Shoenfield. *Mathematical Logic*. Addison-Wesley, 1967. Trad. it. Boringhieri.

[10] R. Smullyan. *What is the name of this book?* Prentice Hall, 1978. trad. it. Zanichelli.

Indice analitico