Università di Udine
FACOLTA' DI SCIENZE MM.FF.NN.
Corso di Laurea in Informatica

dispense di

Elementi di Logica Matematica

Alberto Marcone

e-mail: alberto.marcone@dimi.uniud.it
pagina web: http://users.dimi.uniud.it/~alberto.marcone/ElLogica.html

Versione finale (2 dicembre 2008)

Anno Accademico 2008–2009

Indice

Presentazione

Queste dispense raccolgono il materiale (teoria ed esercizi) sviluppato nell'anno accademico 2008-2009 nel corso di Elementi di Logica Matematica del Corso di Laurea in Informatica triennale dell'Università di Udine. Rispetto alla versione sviluppata per l'anno accademico precedente, il materiale è restato sostanzialmente invariato: i cambiamenti si limitano a piccole correzioni e a qualche riscrittura per rendere più chiara qualche frase.

Ringrazio quanti hanno contribuito o contribuiranno, con le loro segnalazioni di errori, imprecisioni ed omissioni, a migliorare queste dispense.

Alberto Marcone

Introduzione

Sia la logica che l'informatica si occupano di problemi legati alla formalizzazione, elaborazione e comunicazione della conoscenza. Alcune delle radici dell'informatica, che è una disciplina relativamente nuova, sono proprio nella logica. Da un lato la logica si è occupata, soprattutto a partire dal lavoro di Alan Turing intorno al 1935, di studiare il concetto astratto di computer (prima ancora che i computers venissero effettivamente realizzati), dall'altro l'informatica ha l'esigenza di un linguaggio formale e sintatticamente preciso. La logica è la disciplina che da sempre si è occupata dello studio delle leggi del ragionamento. A partire dalla fine del XIX secolo, e con sempre maggior successo fino ad oggi, questo si è sviluppato attraverso la costruzione e lo studio con strumenti matematici dei linguaggi formali.

Per capire le problematiche che affronteremo iniziamo con il considerare tre semplici esempi di "deduzione", in cui da due affermazioni di partenza se ne deduce una terza (nella colonna di destra abbiamo riportato le traduzioni in un opportuno linguaggio formale delle affermazioni della colonna di sinistra: rendere precise queste traduzioni è tra gli scopi di queste dispense).

Ogni cane è un mammifero	$\forall x(c(x) \to m(x))$
Esistono cani non bianchi	$\exists x(c(x) \wedge \neg b(x))$
Non ogni mammifero è bianco	$\neg\forall x(m(x) \to b(x))$

Alcuni studenti sono biondi	$\exists x(s(x) \wedge b(x))$
Alcuni biondi hanno gli occhi azzurri	$\exists x(b(x) \wedge a(x))$
Qualche studente biondo ha gli occhi azzurri	$\exists x(s(x) \wedge b(x) \wedge a(x))$

Ogni computer è una mucca	$\forall x(c(x) \to m(x))$
Esistono computer non biondi	$\exists x(c(x) \wedge \neg b(x))$
Non ogni mucca è bionda	$\neg\forall x(m(x) \to b(x))$

Il primo e il terzo esempio, pur parlando di argomenti assai diversi, e facendo affermazioni più o meno sensate, hanno la stessa forma, e infatti le formule che compaiono nella colonna di destra risultano essere identiche. Queste due deduzioni sono (in un senso che renderemo preciso più avanti) argomenti corretti, e ciò dipende solo dalla loro forma, cioè dalle formule della colonna di destra.

Il secondo esempio, pur facendo asserzioni presumibilmente vere e deducendone una conclusione presumibilmente vera, non è un argomento corretto: possiamo concepire una situazione in cui tutti gli studenti biondi hanno gli occhi neri, pur esistendo studenti biondi e persone bionde con gli occhi azzurri. Invece, per quanto possa essere assurda la situazione descritta, **se** siamo in una situazione in cui ogni computer è una mucca ed esistono computer non biondi, **allora** necessariamente non tutte le mucche sono bionde.

Gli esempi precedenti hanno lo scopo di evidenziare come il nostro interesse si concentri sulla correttezza degli argomenti indipendentemente dal modo in cui le asserzioni contenute in essi vengono "interpretate", cioè tenendo conto di tutte le

possibili interpretazioni di queste asserzioni. Perciò dal punto di vista della logica il primo e il terzo esempio sono del tutto equivalenti.

In questo corso ci concentreremo quindi sulla manipolazione di formule di linguaggi formali, allo scopo di individuare precisamente cosa significa affermare che a partire da un insieme di formule possiamo dedurre logicamente una formula e come possiamo fare per riconoscere se ciò avviene o meno.

Il nostro oggetto di studio saranno quindi le formule e gli insiemi di formule. Questo comporta un'analisi del linguaggio formale che utilizziamo.

L'analisi di un linguaggio formale ha essenzialmente due aspetti:

- la SINTASSI: le "regole" per la costruzione di un linguaggio formale e delle sue espressioni (*termini* e *formule*);
- la SEMANTICA: il "significato" dei termini e delle formule di un linguaggio formale, analizzato attraverso le nozioni di *interpretazione* e di *soddisfazione*.

Inizieremo lo studio di sintassi e semantica in un caso piuttosto semplice, ma che permette di acquisire familiarità con le idee principali della materia, quello della *logica proposizionale*. Ad essa sono dedicati i primi quattro capitoli di queste dispense.

- Nel capitolo 1 definiremo la sintassi della logica proposizionale, cioè le regole che ci permettono di costruire formule in questa logica.
- Il secondo capitolo è dedicato alla definizione della semantica della logica proposizionale, che conduce alle importanti nozioni di *conseguenza logica* (che è la definizione matematicamente precisa della nozione intuitiva di "deduzione corretta") e di *formula valida*. Nell'ultima sezione di questo capitolo vedremo alcuni esempi di *traduzione* di asserzioni da un linguaggio naturale (quale l'italiano) ad un linguaggio formale (quale appunto la logica proposizionale).
- Nel capitolo 3 vengono presentate alcune procedure che permettono di *trasformare* (in modo puramente sintattico) una formula della logica proposizionale in una formula ad essa logicamente equivalente (una nozione semantica) che abbia una forma (sintattica) particolarmente semplice. Visto che il corso è indirizzato a studenti di informatica, particolare attenzione sarà dedicata agli aspetti algoritmici di queste trasformazioni.
- Infine nel quarto capitolo verra studiato il metodo dei *tableaux*, che è un algoritmo relativamente efficiente per stabilire se una formula della logica proposizionale è valida.

I successivi quattro capitoli sono dedicati alla logica predicativa (detta anche logica del prim'ordine). Rivedremo quanto visto nel caso più semplice del calcolo proposizionale, concentrandoci in particolare sugli aspetti più complessi che caratterizzano la logica predicativa.

- Il quinto capitolo è dedicato alla sintassi della logica predicativa. Diverse sono le differenze con il caso proposizionale: a partire da variabili, simboli di costante e di funzione si arriva dapprima alla nozione di termine, e solo a partire da essa, con l'uso dei quantificatori, si definiscono le formule.
- Nel sesto capitolo viene sviluppata la semantica della logica predicativa, assai più complessa di quella vista nel contesto proposizionale, basata sulle nozioni di assegnazione e di interpretazione. Anch'essa conduce alle nozioni di conseguenza logica e di validità che sono da considerarsi le principali di questo corso.
- Nel capitolo 7 viene affrontata la traduzione di asserzioni da un linguaggio naturale ad un linguaggio formale (in questa sede la logica predicativa).

Il lettore potrà apprezzare la maggior espressività della logica predicativa rispetto al calcolo proposizionale.

- Infine nell'ottavo capitolo viene affrontato, in modo sintetico e senza fornire tutte le dimostrazioni, il metodo dei tableaux nel caso predicativo. Saranno messe in risalto le notevoli, ed intrinseche, differenze con il caso proposizionale, ed in particolare la possibilità che il metodo non termini.

Gli esercizi inseriti nel testo sono parte integrante delle dispense e si invita il lettore a tentare sistematicamente di svolgerli. Gli esercizi o le parti di esercizio contrassegnati con ($\star$) sono più impegnativi, e spesso richiedono una certa "maturità matematica". L'appendice B raccoglie ulteriori esercizi che possono essere utili nella preparazione dell'esame. Il lettore è inoltre invitato a consultare i testi delle prove d'esame, completi di soluzioni, reperibili sulla pagina web del docente.

La durata limitata del corso non permette un'introduzione esaustiva a tutti i numerosi aspetti della logica matematica di interesse per l'informatica: ulteriori approfondimenti sono lasciati al corso di Logica Matematica offerto nel terzo anno del Corso di Laurea.

CAPITOLO 1

Sintassi della logica proposizionale

Nella logica proposizionale partiamo da alcune *affermazioni* che non possono venir scomposte in affermazioni più semplici e di cui supponiamo di poter stabilire verità o falsità. Affermazioni di questo tipo sono ad esempio: "la terra è una sfera", "oggi piove", "Roma è la capitale della Francia", "Claudio e Maria sono amici".

A partire da questi *atomi* costruiamo delle affermazioni più complesse, quali "oggi piove oppure Claudio e Maria sono amici", "la terra non è una sfera", "se Roma è la capitale della Francia allora oggi piove", "se oggi piove allora Claudio e Maria sono amici e Roma non è la capitale della Francia". La costruzione di affermazioni più complesse avviene per mezzo di *connettivi* quali "e", "non", "se ... allora ...", "oppure". La verità o falsità di un'affermazione complessa dipende dalla verità o falsità degli atomi da cui essa è stato costruita. Ad esempio l'affermazione "oggi piove e Claudio e Maria sono amici" è vera se e solo se entrambe le affermazioni atomiche "oggi piove" e "Claudio e Maria sono amici" sono vere.

Il lettore si sarà accorto che un'analisi di questo tipo si applica solo a frasi del linguaggio naturale (nel nostro caso l'italiano) di un certo tipo. Non ha senso stabilire se frasi interrogative come "oggi piove?" sono vere o false, e quindi queste frasi non rientrano nel nostro oggetto di analisi. D'altra parte affermazioni a prima vista complesse come "ogni amico di Angela è amico anche di Bruna" non possono essere scomposte in affermazioni più semplici da cui sono state costruite usando i connettivi. Dal punto di vista della logica proposizionale questa affermazione è dunque un atomo. Vedremo però nella seconda parte del corso che la logica predicativa permette di analizzare assai meglio questo tipo di affermazioni.

1. Formule proposizionali

Per approfondire la nostra analisi dobbiamo definire in modo matematicamente preciso il concetto di affermazione che finora abbiamo discusso con riferimento al linguaggio naturale. Nell'ambito della logica matematica un'affermazione è rappresentata da una formula, nel nostro caso della logica proposizionale.

Definizione 1.1. Sia $\mathcal{P}$ un insieme, i cui elementi verranno chiamati *lettere proposizionali* o *formule proposizionali atomiche*. $\mathcal{P}$ determina un *linguaggio proposizionale*, le cui formule saranno costruite utilizzando gli elementi di $\mathcal{P}$ e i seguenti *simboli logici proposizionali*:

- i *connettivi proposizionali* sono i quattro simboli $\neg$, $\wedge$, $\vee$ e $\rightarrow$, che sono chiamati rispettivamente *negazione*, *congiunzione*, *disgiunzione* e *implicazione*; il primo è un *connettivo unario*, mentre i restanti tre sono *connettivi binari*;
- le *parentesi* sono i due simboli (e).

Convenzione 1.2. Nei primi quattro capitoli le lettere $p, q, r, s, \ldots$ (eventualmente fornite di pedice: $p_0, q_1, p_i, \ldots$) rappresentano sempre lettere proposizionali del linguaggio che stiamo considerando.

Una formula proposizionale è una stringa di lettere proposizionali, connettivi proposizionali e parentesi, costruita secondo le regole specificate dalla prossima definizione.

DEFINIZIONE 1.3. L'insieme delle *formule proposizionali* (o, limitatamente ai primi quattro capitoli, formule) è definito per ricorsione come segue:

- se $p \in \mathcal{P}$ allora p è una formula;
- se F è una formula allora $(\neg F)$ è una formula;
- se F e G sono formule allora $(F \wedge G)$ è una formula;
- se F e G sono formule allora $(F \vee G)$ è una formula;
- se F e G sono formule allora $(F \to G)$ è una formula.

La formula $(\neg F)$ viene letta "non F", la formula $(F \wedge G)$ viene letta "F e G" oppure "F et G", la formula $(F \vee G)$ viene letta "F oppure G" o anche "F vel G" ("vel" è una delle due forme di "oppure" in latino, vedi nota 2.5), la formula $(F \to G)$ viene letta "se F allora G" oppure "F implica G".

ESEMPIO 1.4. La definizione 1.3 ci permette di riconoscere se una certa stringa di simboli è una formula. E' facile verificare che $\wedge p \to$ non è una formula (non è un elemento di $\mathcal{P}$ e non è stata costruita utilizzando nessuna delle altre quattro condizioni). D'altra parte $((\neg p) \vee q)$ è una formula: per verificarlo dettagliatamente si osserva che sono formule p, $(\neg p)$, q ed infine $((\neg p) \vee q)$ stessa (abbiamo utilizzato nell'ordine la prima, la seconda, la prima e la quarta condizione della definizione 1.3).

ESERCIZIO 1.5. Quali delle seguenti stringhe di simboli sono formule? $(\vee pq)$, $(\neg(p \to (q \wedge p)))$, $((p \wedge q) \to (r \vee))$, $((p \wedge (\neg q)) \vee (q \to r))$, $(p \neg r)$, $(\wedge p)$, $p \to q$.

NOTAZIONE 1.6. Nel seguito le lettere maiuscole F, G e H, eventualmente fornite di pedice, indicheranno sempre formule (proposizionali nei primi quattro capitoli, predicative nei seguenti quattro capitoli). Con le lettere maiuscole greche Γ, Δ e Σ indicheremo invece insiemi di formule. Con Γ, F indicheremo l'insieme $\Gamma \cup \{F\}$ e similmente Γ, F, G significa $\Gamma \cup \{F, G\}$.

Il seguente lemma ci permette di classificare le formule a seconda dell'ultimo connettivo usato nella loro costruzione.

LEMMA 1.7. *Ogni formula proposizionale è di uno e uno solo dei seguenti cinque tipi:*

- *una lettera proposizionale;*
- *una* negazione*, cioè una formula del tipo $(\neg F)$ per qualche formula F;*
- *una* congiunzione*, cioè una formula del tipo $(F \wedge G)$ per formule F e G;*
- *una* disgiunzione*, cioè una formula del tipo $(F \vee G)$ per formule F e G;*
- *un'*implicazione*, cioè una formula del tipo $(F \to G)$ per formule F e G.*

ESERCIZIO 1.8. Di che tipo sono la formula $((\neg p) \vee q)$ e le stringhe dell'esercizio 1.5 che sono formule?

La definizione 1.3 è ricorsiva, e questo ci permette di dimostrare proprietà delle formule ragionando per ricorsione. Una dimostrazione di questo tipo è giustificata dal seguente teorema e viene detta *per induzione sulla complessità delle formule.*

TEOREMA 1.9 (Induzione sulla complessità delle formule). *Sia A una proprietà che può valere per le stringhe di simboli. Supponiamo di dimostrare che*

- *$A(p)$ vale per ogni $p \in \mathcal{P}$;*
- *se vale $A(F)$ per una formula F allora vale anche $A((\neg F))$;*
- *se valgono $A(F)$ e $A(G)$ per formule F e G allora vale anche $A((F \wedge G))$;*

- *se valgono* $A(F)$ *e* $A(G)$ *per formule* F *e* G *allora vale anche* $A((F \vee G))$*;*
- *se valgono* $A(F)$ *e* $A(G)$ *per formule* F *e* G *allora vale anche* $A((F \to G))$.

Allora $A(F)$ *vale per ogni formula* F*.*

Similmente è possibile dare definizioni procedendo *per ricorsione sulla complessità delle formule*. Anziché dare una descrizione generale di questo procedimento, preferiamo fornire un esempio del suo utilizzo.

DEFINIZIONE 1.10. Il *grado della formula* F, indicato con $g(F)$, è definito da:

- $g(F) = 0$ se F è una lettera proposizionale;
- $g(\neg F) = g(F) + 1$;
- $g(F \wedge G) = g(F \vee G) = g(F \to G) = g(F) + g(G) + 1$.

ESERCIZIO 1.11. Calcolare il grado delle formule dell'esempio 1.4 e dell'esercizio 1.5.

ESERCIZIO 1.12. ($\star$) Dimostrare per induzione sulla complessità delle formule che il grado di F è il numero di connettivi che compaiono in F.

2. Usare meno parentesi: precedenze tra connettivi

Il lettore che ha svolto con cura l'esercizio 1.5 si sarà accorto che la nostra definizione di formula comporta l'uso di molte parentesi, più di quelle a cui siamo solitamente abituati. Si considerino ad esempio le stringhe $(\neg((\neg p) \to (q \wedge (\neg r))))$, $(\neg((\neg p) \to (q \wedge (\neg r))))$ e $p \wedge q$: secondo la nostra definizione la prima è una formula, la seconda no (perché una parentesi aperta non si "chiude") e la terza neppure, a causa della mancanza di parentesi. La differenza tra la prima e la seconda stringa è però difficile da cogliere, almeno all'occhio umano, proprio a causa dell'alto numero di parentesi. La terza stringa è invece di facile lettura, e forse l'abbiamo già incontrata come formalizzazione di "p e q". Secondo le nostre regole la formula che si legge "p e q" è $(p \wedge q)$, appesantita dalle parentesi esterne.

Per semplificare la lettura introdurremo ora alcune regole di precedenza tra i connettivi. Per capire ciò che intendiamo fare è bene richiamare le regole sull'uso delle parentesi in aritmetica, con cui siamo talmente familiari da non accorgerci neppure più del loro utilizzo. In quel contesto la scrittura $a \cdot b + c$ va intesa come $(a \cdot b) + c$ e non come $a \cdot (b + c)$: questo perché alla moltiplicazione è stata assegnata (per convenzione universalmente accettata) precedenza sull'addizione. Similmente $a \cdot b^2$ significa $a \cdot (b^2)$ e non $(a \cdot b)^2$: l'operazione di elevamento a potenza ha precedenza sulla moltiplicazione. Naturalmente ci sono dei casi in cui l'uso delle parentesi è inevitabile, per sovvertire l'ordine di precedenza delle operazioni aritmetiche fissato: è il caso di $((a + b) \cdot c)^2$, in cui vanno eseguite nell'ordine addizione, prodotto e elevamento a potenza.

CONVENZIONE 1.13. Nella scrittura delle formule adotteremo le seguenti *convenzioni*:

(1) si omettono le parentesi più esterne;
(2) $\neg$ ha la precedenza su $\wedge$, $\vee$ e $\to$, così che $\neg F \vee G$ abbrevia $((\neg F) \vee G)$;
(3) $\wedge$ e $\vee$ hanno la precedenza su $\to$, così che $F \wedge G \to H$ e $F \to G \vee H$ abbreviano rispettivamente $((F \wedge G) \to H)$ e $(F \to (G \vee H))$;
(4) ulteriori parentesi eventualmente omesse in formule costruite con più di una $\wedge$ o $\vee$ si appoggiano a sinistra, così che $F \wedge G \wedge H$ e $F \vee G \vee H$ abbreviano rispettivamente $((F \wedge G) \wedge H)$ e $((F \vee G) \vee H)$.

ESEMPIO 1.14. Consideriamo nuovamente la formula $(\neg((\neg p) \to (q \wedge (\neg r))))$: utilizzando la convenzione 1.13 essa viene abbreviata da $\neg(\neg p \to q \wedge \neg r)$, che è assai più leggibile.

Esercizio 1.15. Stabilite qual è la differenza tra le formule (scritte adottando la convenzione 1.13) $\neg p \to \neg q \wedge r$ e $\neg(p \to \neg q) \wedge r$ analizzando i passaggi attraverso cui sono state costruite a partire dalle lettere proposizionali.

D'ora in poi utilizzeremo sistematicamente, e senza più richiamarla esplicitamente, la convenzione 1.13 per scrivere le formule.

3. Sottoformule

Definizione 1.16. Se F è una formula, diciamo che G è una *sottoformula* di F se G è una formula che è una sottostringa di F. G è una *sottoformula propria* di F se è diversa da F.

La definizione precedente va applicata tenendo a mente la definizione 1.3 di formula, anche quando si utilizza la convenzione 1.13.

Esempio 1.17. Se F è $p \to q \vee \neg r$, allora $q \vee \neg r$ è una sottoformula di F, mentre $p \to q$ non lo è. Infatti, inserendo alcune delle parentesi omesse in base alla convenzione 1.13, F è $p \to (q \vee \neg r)$. In effetti $q \vee \neg r$ è una delle formule utilizzate nella costruzione di F, mentre $p \to q$ non lo è.

Esercizio 1.18. Elencate tutte le sottoformule della F dell'esempio precedente (sono sei, di cui cinque proprie).

Per dare una definizione precisa di sottoformula possiamo procedere per ricorsione sulla complessità delle formule.

Definizione 1.19. Definiamo per ricorsione sulla complessità della formula F quali sono le *sottoformule* di F:

- se F è una lettera proposizionale, F è la sua unica sottoformula;
- se F è $\neg G$ allora le sottoformule di F sono le sottoformule di G e F stessa;
- se F è $G \wedge H$, $G \vee H$ oppure $G \to H$ allora le sottoformule di F sono le sottoformule di G, le sottoformule di H e F stessa.

CAPITOLO 2

Semantica della logica proposizionale

Nel capitolo 1 abbiamo introdotto la nozione di formula, ora il nostro obiettivo è stabilire quando una formula risulta essere vera. Anche in questo caso è utile ricorrere all'analogia con l'aritmetica: l'uguaglianza $a + b^2 = c \cdot a$ non è né vera né falsa, almeno sino a quando non assegnamo dei valori a a, b e c. Se $a = 8$, $b = 4$ e $c = 3$ essa risulterà essere vera, mentre per $a = 3$, $b = 4$ e $c = 7$ essa risulterà falsa. In modo analogo la verità o falsità delle formule proposizionali dipende dalla verità o falsità delle lettere proposizionali che sono state utilizzate nella loro costruzione.

1. Valutazioni e interpretazioni

DEFINIZIONE 2.1. I *valori di verità* sono $\mathbf{V}$ e $\mathbf{F}$: il valore $\mathbf{V}$ corrisponde a *vero*, il valore $\mathbf{F}$ corrisponde a *falso*.

DEFINIZIONE 2.2. Una *valutazione* per il linguaggio proposizionale $\mathcal{P}$ è una funzione $v : \mathcal{P} \to \{\mathbf{V}, \mathbf{F}\}$ che associa da ogni lettera proposizionale un valore di verità.

Una valutazione può venir estesa ad una funzione che associa un valore di verità ad ogni formula nel modo seguente.

DEFINIZIONE 2.3. Sia $v : \mathcal{P} \to \{\mathbf{V}, \mathbf{F}\}$ una valutazione. L'*interpretazione* $\bar{v}$ associa ad ogni formula F un valore di verità ed è definita per ricorsione sulla complessità delle formule nel modo seguente:

- se F è un $p \in \mathcal{P}$ allora $\bar{v}(F) = v(p)$.
- se F è $\neg G$ allora $\bar{v}(F) = \begin{cases} \mathbf{V}, & \text{se } \bar{v}(G) = \mathbf{F}; \\ \mathbf{F}, & \text{se } \bar{v}(G) = \mathbf{V}. \end{cases}$
- se F è $G \wedge H$ allora $\bar{v}(F) = \begin{cases} \mathbf{V}, & \text{se } \bar{v}(G) = \mathbf{V} \text{ e } \bar{v}(H) = \mathbf{V}; \\ \mathbf{F}, & \text{altrimenti.} \end{cases}$
- se F è $G \vee H$ allora $\bar{v}(F) = \begin{cases} \mathbf{F}, & \text{se } \bar{v}(G) = \mathbf{F} \text{ e } \bar{v}(H) = \mathbf{F}; \\ \mathbf{V}, & \text{altrimenti.} \end{cases}$
- se F è $G \to H$ allora $\bar{v}(F) = \begin{cases} \mathbf{F}, & \text{se } \bar{v}(G) = \mathbf{V} \text{ e } \bar{v}(H) = \mathbf{F}; \\ \mathbf{V}, & \text{altrimenti.} \end{cases}$

La definizione 2.3 va vista come un'usuale definizione matematica in cui viene dato per acquisito il senso di "se" e "e".

NOTA 2.4. La seguente tabella riassume alcune parti della definizione precedente. In ogni riga sono indicati i valori di verità di $F \wedge G$, $F \vee G$ e $F \to G$ in corrispondenza dei valori di verità di F e G indicati nelle prime due colonne.

F	G	$F \wedge G$	$F \vee G$	$F \to G$
$\mathbf{V}$	$\mathbf{V}$	$\mathbf{V}$	$\mathbf{V}$	$\mathbf{V}$
$\mathbf{V}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{V}$	$\mathbf{F}$
$\mathbf{F}$	$\mathbf{V}$	$\mathbf{F}$	$\mathbf{V}$	$\mathbf{V}$
$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{F}$	$\mathbf{V}$

NOTA 2.5. Notiamo che $F \vee G$ è vera se e solo se F è vera oppure G è vera. Qui intendiamo "oppure" nel senso inclusivo del "vel" latino (da cui fra l'altro proviene il simbolo $\vee$) per cui è ammesso che entrambe le asserzioni siano vere. In latino esiste anche l'oppure esclusivo "aut" (di cui resta traccia nell'espressione "aut aut"), per affermare che una e una sola delle due alternative è vera. È possibile introdurre un ulteriore connettivo logico per designare l'aut, ma è più semplice osservare che "F aut G" può venir scritto[1] come $(F \vee G) \wedge \neg(F \wedge G)$.

NOTA 2.6. La nostra intenzione è che $F \to G$ sia vera se e solo se è vera l'affermazione "se F allora G". Qui "se ... allora ..." va inteso nel senso dell'implicazione materiale: se F è vero deve esserlo anche G, se F è falso allora l'implicazione è vera, indipendentemente dalla verità di G. Perciò $F \to G$ è vero se e solo se F è falso oppure G è vero. Questo giustifica la nostra definizione: perché $F \to G$ sia falso è necessario che F sia vero e G falso.

ESEMPIO 2.7. Consideriamo la valutazione v definita da $v(p) = \mathbf{F}$, $v(q) = \mathbf{V}$, $v(r) = \mathbf{V}$ e supponiamo di voler calcolare $\bar{v}(F)$ dove F è $\neg(\neg p \to q \wedge \neg r)$. Applicando le varie parti della definizione di interpretazione troveremo: $\bar{v}(\neg r) = \mathbf{F}$, $\bar{v}(q \wedge \neg r) = \mathbf{F}$, $\bar{v}(\neg p) = \mathbf{V}$, $\bar{v}(\neg p \to q \wedge \neg r) = \mathbf{F}$, ed infine $\bar{v}(F) = \mathbf{V}$. La formula F risulta dunque essere vera nell'interpretazione $\bar{v}$.

Continuando ad utilizzare la stessa valutazione notiamo che $\bar{v}(p \to (q \to p)) = \mathbf{V}$, $\bar{v}((p \to q) \to p) = \mathbf{F}$, $\bar{v}((p \wedge q) \vee r) = \mathbf{V}$ e $\bar{v}(p \wedge (q \vee r)) = \mathbf{F}$. Questo mostra che le scritture $p \to q \to p$ e $p \wedge q \vee r$ sono ambigue (la convenzione 1.13 non stabilisce che queste stringhe abbreviano delle formule).

Il prossimo lemma è la formulazione matematicamente precisa di un'osservazione intuitiva che abbiamo utilizzato implicitamente nell'esempio 2.7: il valore di verità di una formula secondo l'interpretazione $\bar{v}$ dipende solo dai valori assunti da v sulle lettere proposizionali che compaiono nella formula.

LEMMA 2.8. *Sia F una formula e sia $\mathcal{P}' \subseteq \mathcal{P}$ l'insieme delle lettere proposizionali che compaiono in F. Siano v_1 e v_2 due valutazioni tali che $v_1(p) = v_2(p)$ per ogni $p \in \mathcal{P}'$. Allora $\bar{v}_1(F) = \bar{v}_2(F)$.*

DIMOSTRAZIONE. Per induzione sulla complessità di F. □

NOTAZIONE 2.9. Per semplificare la notazione d'ora in poi scriveremo v anche per l'interpretazione, confondendola così con la valutazione da cui siamo partiti per definirla.

DEFINIZIONE 2.10. Se v è un'interpretazione e F una formula, diciamo che v *soddisfa* F o F *è soddisfatta da* v se $v(F) = \mathbf{V}$. Se Γ è un insieme di formule, diciamo che v *soddisfa* Γ se v soddisfa ogni formula di Γ.

ESERCIZIO 2.11. Stabilire se la valutazione v dell'esempio 2.7 soddisfa le formule $(p \to \neg q) \vee \neg(r \wedge q)$ e $\neg(\neg p \to q) \wedge r$.

2. Equivalenza e conseguenza logica

Siano F e G le formule $p \vee q$ e $q \vee p$: è facile convincersi che per qualunque valutazione v si ha $v(F) = v(G)$. F e G sono dunque, in qualche modo, equivalenti.

DEFINIZIONE 2.12. Diciamo che le formule F e G sono *logicamente equivalenti* (in simboli $F \equiv G$) se per ogni interpretazione v si ha $v(F) = v(G)$.

[1] L'espressione "può venir scritto" viene resa precisa come segue. Sia $\oplus$ un nuovo connettivo binario per l'aut, con la regola $\bar{v}(F \oplus G) = \mathbf{V}$ se e solo se $\bar{v}(F) = \mathbf{V}$ e $\bar{v}(G) = \mathbf{F}$ oppure $\bar{v}(F) = \mathbf{F}$ e $\bar{v}(G) = \mathbf{V}$. Allora $F \oplus G \equiv (F \vee G) \wedge \neg(F \wedge G)$, dove il simbolo $\equiv$ di equivalenza logica verrà introdotto nella definizione 2.12.

NOTA 2.13. L'equivalenza logica è (come suggerisce il nome) una relazione di equivalenza sull'insieme delle formule: $F \equiv F$, da $F \equiv G$ segue $G \equiv F$, e se $F \equiv G$ e $G \equiv H$ allora $F \equiv H$.

ESEMPIO 2.14. Siano F e G le formule $p \to \neg p$ e $\neg p$. Sia v una valutazione qualsiasi. Ci sono due possibilità: se $v(p) = \mathbf{V}$ allora si verifica che $v(F) = v(G) = \mathbf{F}$; se $v(p) = \mathbf{F}$ invece $v(F) = v(G) = \mathbf{V}$. In ogni caso $v(F) = v(G)$, e perciò $F \equiv G$.

Siano ora F e G le formule $p \wedge q$ e p. F e G non sono logicamente equivalenti: basta considerare una valutazione con $v(p) = \mathbf{V}$ e $v(q) = \mathbf{F}$. Possiamo però notare che se un'interpretazione soddisfa F allora soddisfa anche G: la verità di F ha come conseguenza la verità di G. Esiste dunque un legame tra le due formule.

DEFINIZIONE 2.15. Siano F e G due formule. Diciamo che G è *conseguenza logica* di F (in simboli $F \models G$) se ogni interpretazione che soddisfa F soddisfa anche G.

NOTA 2.16. La relazione di conseguenza logica è riflessiva e transitiva: è immediato verificare che $F \models F$, e da $F \models G$ e $G \models H$ segue $F \models H$.

ESEMPIO 2.17. Siano F e G le formule $(p \to q) \wedge p$ e $p \vee q$ rispettivamente.

$F \models G$: infatti se v è una valutazione che soddisfa F allora in particolare deve essere $v(p) = \mathbf{V}$, da cui segue che v soddisfa G.

$G \not\models F$: si consideri ad esempio v tale che $v(p) = \mathbf{F}$ e $v(q) = \mathbf{V}$; v soddisfa G ma non soddisfa F.

Notiamo che abbiamo dimostrato anche che $F \not\equiv G$.

L'esempio precedente evidenzia come per mostrare che $G \not\models F$ sia sufficiente trovare **una** valutazione che soddisfa G e non soddisfa F. Invece per mostrare che $F \models G$ è necessario considerare **tutte** le interpretazioni che soddisfano F.

NOTA 2.18. Notiamo che $F \equiv G$ e $F \models G$ **non** sono formule: infatti $\equiv$ e $\models$ non fanno parte dei connettivi introdotti nel capitolo 1. Le espressioni $F \equiv G$ e $F \models G$ sono abbreviazioni per certe affermazioni che noi facciamo parlando delle formule della logica, in quello che usualmente è chiamato *metalinguaggio* (il linguaggio che utilizziamo per parlare dei linguaggi logici).

Possiamo affermare che gran parte delle presenti dispense riguarda lo studio delle relazioni $\models$ e $\equiv$, le cui definizioni sono quindi le più importanti del corso. Esse sono strettamente collegate tra loro, come evidenziato dal prossimo lemma.

LEMMA 2.19. *Due formule F e G sono logicamente equivalenti se e solo se $F \models G$ e $G \models F$.*

DIMOSTRAZIONE. Se $F \models G$ e $G \models F$ consideriamo una valutazione qualsiasi v. Se $v(F) = \mathbf{V}$ allora $F \models G$ implica $v(G) = \mathbf{V}$. Se $v(F) = \mathbf{F}$ allora non può essere $v(G) = \mathbf{V}$: altrimenti $G \models F$ implicherebbe $v(F) = \mathbf{V}$.

La direzione inversa è lasciata come facile esercizio per il lettore. □

Il prossimo lemma raccoglie alcune equivalenze e conseguenze logiche elementari, che seguono immediatamente dalle definizioni.

LEMMA 2.20. *Se F e G sono formule allora:*

(1) $F \equiv \neg\neg F$;
(2) $F \wedge G \equiv G \wedge F$;
(3) $F \vee G \equiv G \vee F$;
(4) $F \wedge F \equiv F$;
(5) $F \vee F \equiv F$;

(6) $F \models F \vee G$ *e* $G \models F \vee G$;
(7) $F \wedge G \models F$ *e* $F \wedge G \models G$.

DIMOSTRAZIONE. Immediata dalle definizioni 2.3, 2.15 e 2.12. Si invita il lettore a svolgere in dettaglio almeno alcune di queste dimostrazioni. □

Notiamo che ciascuna delle affermazioni del lemma 2.20 è uno *schema* di equivalenze e conseguenze logiche e non una singola equivalenza o conseguenza logica. Ad esempio (2) significa che, tra le altre, valgono le equivalenze logiche $p \wedge q \equiv q \wedge p$, $p \wedge r \equiv r \wedge p$ e $(p \to q \vee \neg r) \wedge (\neg q \vee s) \equiv (\neg q \vee s) \wedge (p \to q \vee \neg r)$. Quasi tutte le nostre affermazioni sulle conseguenze ed equivalenze logiche saranno schemi di questo tipo.

Il seguente lemma è molto utile per dimostrare l'equivalenza logica. Esso asserisce che sostituendo all'interno di una formula una sottoformula con una formula ad essa equivalente, si ottiene una formula equivalente a quella di partenza.

LEMMA 2.21. *Se F è una sottoformula di una formula H e $F \equiv G$ allora $H \equiv H'$ dove H' è la formula ottenuta da H rimpiazzando la sottoformula F con G.*

DIMOSTRAZIONE. Fissate F e G, la dimostrazione è per induzione sulla complessità della formula H di cui F è sottoformula. Il caso più semplice possibile è quello in cui H è F: in questo caso H' è G e il risultato segue da $F \equiv G$.

Se H è una negazione $\neg H_0$ e F non è H allora F è una sottoformula di H_0. Per ipotesi induttiva $H_0 \equiv H_0'$ e quindi $\neg H_0 \equiv \neg H_0'$, cioè $H \equiv H'$.

Se H è del tipo $H_0 \circ H_1$ con $\circ \in \{\wedge, \vee \to\}$ e F non è H allora F è una sottoformula di H_0 oppure F è una sottoformula di H_1. Nel primo caso H' è $H_0' \circ H_1$ e per ipotesi induttiva $H_0 \equiv H_0'$ da cui è facile ricavare che $H_0 \circ H_1 \equiv H_0' \circ H_1$, cioè $H \equiv H'$. Il secondo caso è analogo. □

ESEMPIO 2.22. Utilizzando il lemma 2.21, la transitività di $\equiv$, e alcune delle equivalenze logiche del lemma 2.20 è facile dimostrare che

$$F \wedge G \to (\neg F \vee H) \wedge (H \vee \neg F) \equiv \neg\neg G \wedge F \to H \vee \neg F.$$

LEMMA 2.23. *Se F, G e H sono formule allora:*

(1) $\neg(F \wedge G) \equiv \neg F \vee \neg G$;
(2) $\neg(F \vee G) \equiv \neg F \wedge \neg G$;
(3) $F \to G \equiv \neg F \vee G$;
(4) $\neg(F \to G) \equiv F \wedge \neg G$;
(5) $(F \wedge G) \wedge H \equiv F \wedge (G \wedge H)$;
(6) $(F \vee G) \vee H \equiv F \vee (G \vee H)$;
(7) $F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$;
(8) $(G \vee H) \wedge F \equiv (G \wedge F) \vee (H \wedge F)$;
(9) $F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$;
(10) $(G \wedge H) \vee F \equiv (G \vee F) \wedge (H \vee F)$.

DIMOSTRAZIONE. Spesso utilizzeremo il lemma 2.19, senza citarlo esplicitamente. In tutta la dimostrazione v è un'interpretazione arbitraria.

(1) Se v soddisfa $\neg(F \wedge G)$ consideriamo due possibilità. Se $v(F) = \mathbf{F}$, allora $v(\neg F) = \mathbf{V}$ e quindi $v(\neg F \vee \neg G) = \mathbf{V}$. Se invece $v(F) = \mathbf{V}$, dall'ipotesi che $v(\neg(F \wedge G)) = \mathbf{V}$, si ottiene che $v(G) = \mathbf{F}$, quindi che $v(\neg G) = \mathbf{V}$ e dunque che $v(\neg F \vee \neg G) = \mathbf{V}$. In ogni caso $v(\neg F \vee \neg G) = \mathbf{V}$. Quindi $\neg(F \wedge G) \models \neg F \vee \neg G$.
Se v soddisfa $\neg F \vee \neg G$ allora $v(F) = \mathbf{F}$ oppure $v(G) = \mathbf{F}$. In ogni caso $v(F \wedge G) = \mathbf{F}$, cioè $v(\neg(F \wedge G)) = \mathbf{V}$. Perciò $\neg F \vee \neg G \models \neg(F \wedge G)$.
(2) Tanto $v(\neg(F \vee G)) = \mathbf{V}$ che $v(\neg F \wedge \neg G) = \mathbf{V}$ sono equivalenti a $v(F) = \mathbf{F}$ e $v(G) = \mathbf{F}$.

(3) Se v soddisfa $F \to G$ consideriamo due possibilità. Se $v(F) = \mathbf{V}$ allora $v(G) = \mathbf{V}$ e quindi $v(\neg F \vee G) = \mathbf{V}$. Se invece $v(F) = \mathbf{F}$, allora $v(\neg F) = \mathbf{V}$ e quindi $v(\neg F \vee G) = \mathbf{V}$. In ogni caso v soddisfa $\neg F \vee G$, così che $F \to G \models \neg F \vee G$.

Se v soddisfa $\neg F \vee G$ è immediato constatare che sia nel caso in cui $v(\neg F) = \mathbf{V}$, sia in quello in cui $v(G) = \mathbf{V}$ si ha che $v(F \to G) = \mathbf{V}$. Quindi $\neg F \vee G \models F \to G$.

(4) Questa equivalenza si può ottenere, grazie al lemma 2.21 e alla transitività di $\equiv$, utilizzando nell'ordine la (3), la (2) e il lemma 2.20.1:

$$\neg(F \to G) \equiv \neg(\neg F \vee G) \equiv \neg\neg F \wedge \neg G \equiv F \wedge \neg G.$$

(5) e (6) sono immediate, usando l'associatività di "e" e "oppure" del linguaggio naturale.

(7) Se $v(F \wedge (G \vee H)) = \mathbf{V}$ allora $v(F) = \mathbf{V}$ e $v(G \vee H) = \mathbf{V}$. Se $v(G) = \mathbf{V}$ allora $v(F \wedge G) = \mathbf{V}$, mentre se $v(H) = \mathbf{V}$ allora $v(F \wedge H) = \mathbf{V}$. In ogni caso $v((F \wedge G) \vee (F \wedge H)) = \mathbf{V}$. Perciò $F \wedge (G \vee H) \models (F \wedge G) \vee (F \wedge H)$.

Sia $v((F \wedge G) \vee (F \wedge H)) = \mathbf{V}$. Se $v(F \wedge G) = \mathbf{V}$ allora $v(F) = \mathbf{V}$ e $v(G) = \mathbf{V}$, mentre se $v(F \wedge H) = \mathbf{V}$ allora $v(F) = \mathbf{V}$ e $v(H) = \mathbf{V}$. In ogni caso $v(F) = \mathbf{V}$ e $v(G \vee H) = \mathbf{V}$. Quindi $v(F \wedge (G \vee H)) = \mathbf{V}$ e $(F \wedge G) \vee (F \wedge H) \models F \wedge (G \vee H)$.

(8) segue da (7) utilizzando il lemma 2.20.2 e il lemma 2.21.

(9) Se $v(F \vee (G \wedge H)) = \mathbf{V}$ allora $v(F) = \mathbf{V}$ oppure $v(G \wedge H) = \mathbf{V}$. Se vale $v(F) = \mathbf{V}$ allora $v(F \vee G) = \mathbf{V}$ e $v(F \vee H) = \mathbf{V}$, così che $v((F \vee G) \wedge (F \vee H)) = \mathbf{V}$. Se invece $v(G \wedge H) = \mathbf{V}$ allora $v(G) = \mathbf{V}$ e $v(H) = \mathbf{V}$, e quindi $v((F \vee G) \wedge (F \vee H)) = \mathbf{V}$ anche in questo caso. Abbiamo mostrato che $F \vee (G \wedge H) \models (F \vee G) \wedge (F \vee H)$.

Se $v((F \vee G) \wedge (F \vee H)) = \mathbf{V}$ allora $v(F \vee G) = \mathbf{V}$ e $v(F \vee H) = \mathbf{V}$. Se $v(F) = \mathbf{V}$ allora ovviamente $v(F \vee (G \wedge H)) = \mathbf{V}$. Se invece $v(F) = \mathbf{F}$ allora abbiamo tanto $v(G) = \mathbf{V}$ che $v(H) = \mathbf{V}$: quindi $v(G \wedge H) = \mathbf{V}$ e di conseguenza $v(F \vee (G \wedge H)) = \mathbf{V}$. Perciò $(F \vee G) \wedge (F \vee H) \models F \vee (G \wedge H)$.

(10) segue da (9) utilizzando il lemma 2.20.3 e il lemma 2.21. □

Esercizio 2.24. Dare una dimostrazione diretta usando le interpretazioni di (4) del lemma 2.23.

Esercizio 2.25. Utilizzando i lemmi 2.20 e 2.23 (nonchè il lemma 2.21) dimostrare le seguenti equivalenze logiche:

$$F \to G \equiv \neg G \to \neg F;$$
$$(F \to G) \wedge (H \vee \neg F) \equiv \neg(H \to F) \vee (F \to (G \wedge (F \to H)));$$
$$\neg((F \vee G) \wedge (\neg G \vee H)) \equiv F \vee G \to G \wedge \neg H.$$

In alcune circostanze è utile estendere la nozione di conseguenza logica ad insiemi di formule.

Definizione 2.26. Siano Γ e G un insieme di formule ed una formula. Diciamo che G è *conseguenza logica* di Γ (in simboli $\Gamma \models G$) se ogni interpretazione che soddisfa ogni formula di Γ soddisfa anche G.

Nota 2.27. È immediato che se $F \in \Gamma$ e $F \models G$ allora $\Gamma \models G$. È invece possibile che $\Gamma \models G$ ma che nessuna $F \in \Gamma$ sia tale che $F \models G$: considerare ad esempio $\Gamma = \{p, q\}$, $G = p \wedge q$.

Notazione 2.28. Se $\Gamma = \{F_1, \ldots, F_n\}$ è un insieme finito di formule spesso scriveremo $F_1, \ldots, F_n \models G$ al posto di $\{F_1, \ldots, F_n\} \models G$.

Lemma 2.29. *Se $F_1, \ldots, F_n$ e G sono formule, allora $F_1, \ldots, F_n \models G$ se e solo se $F_1 \wedge \cdots \wedge F_n \models G$.*

Dimostrazione. Immediata dalle definizioni. □

ESERCIZIO 2.30. Siano Γ e Δ insiemi di formule, F, G e H formule. Dimostrare le seguenti proprietà:

(a) $F, F \to G \models G$;
(b) $\neg G, F \to G \models \neg F$;
(c) Se $\Gamma \models F$ e $\Delta \supseteq \Gamma$ allora $\Delta \models F$;
(d) Se $\Gamma \models F$ e $\Delta, F \models G$ allora $\Gamma \cup \Delta \models G$;
(e) $\Gamma, F \models G$ se e solo se $\Gamma \models F \to G$;
(f) $\Gamma, \neg F \models G$ se e solo se $\Gamma \models F \vee G$;
(g) Se $\Gamma \models F$ e $\Gamma \models F \to G$ allora $\Gamma \models G$;
(h) Se $\Gamma \models F$ e $\Gamma, G \models H$ allora $\Gamma, F \to G \models H$.

3. Validità e soddisfacibilità

DEFINIZIONE 2.31. Se F è una formula diciamo che

- F è *valida* se F è soddisfatta da ogni interpretazione;
- F è *soddisfacibile* se F è soddisfatta da qualche interpretazione;
- F è *insoddisfacibile* se non esiste un'interpretazione che soddisfa F.

Ovviamente una formula è insoddisfacibile se e solo se non è soddisfacibile.

ESEMPIO 2.32. La formula $p \vee \neg p$ è valida. La formula $p \wedge \neg p$ è insoddisfacibile. La formula $p \to \neg p$ è soddisfacibile ma non valida.

Più in generale, per ogni formula F la formula $F \vee \neg F$ è valida, mentre $F \wedge \neg F$ è insoddisfacibile.

ESERCIZIO 2.33. Dimostrate che ogni lettera proposizionale è soddisfacibile e non valida.

NOTA 2.34. Osserviamo che se $F \equiv G$ e F è valida (soddisfacibile, insoddisfacibile), allora G è valida (soddisfacibile, insoddisfacibile).

Un importante legame tra validità e insoddisfacibilità è contenuto nel prossimo teorema.

TEOREMA 2.35. *Sia F una formula:*

(a) *F è valida se e solo se $\neg F$ è insoddisfacibile;*
(b) *F è insoddisfacibile se e solo se $\neg F$ è valida.*

DIMOSTRAZIONE. (a) F è valida se e solo se per ogni interpretazione v si ha $v(F) = \mathbf{V}$, se e solo se per ogni interpretazione v si ha $v(\neg F) = \mathbf{F}$, se e solo se $\neg F$ è insoddisfacibile.

(b) Si può ragionare come in (a), oppure osservare che per (a) $\neg F$ è valida se e solo se $\neg\neg F$ è insoddisfacibile: per il lemma 2.20.1 $\neg\neg F \equiv F$ e quindi l'ultima condizione è equivalente alla insoddisfacibilità di F. □

ESEMPIO 2.36. Verifichiamo che la formula $(p \to q) \wedge (q \to \neg p)$, che indicheremo con F, è soddisfacibile.

A questo scopo ci basta trovare una valutazione v che soddisfa F. Ponendo $v(p) = \mathbf{F}$ e $v(q) = \mathbf{V}$ è facile verificare che vale $v(F) = \mathbf{V}$.

Notiamo che anche la valutazione v' tale che $v'(p) = \mathbf{F}$ e $v'(q) = \mathbf{F}$ soddisfa F: comunque una sola valutazione è sufficiente a mostrare la soddisfacibilità di F.

ESEMPIO 2.37. Verifichiamo che la formula $(p \to q) \vee (p \wedge (q \to \neg q))$, che indicheremo con G, è valida.

A questo scopo consideriamo una valutazione arbitraria v. Se $v(p \to q) = \mathbf{V}$ allora v soddisfa G. Se v non soddisfa $p \to q$ allora deve essere $v(p) = \mathbf{V}$ e $v(q) = \mathbf{F}$. Allora v soddisfa $q \to \neg q$ e quindi anche $p \wedge (q \to \neg q)$. Ma allora anche in questo

caso $v(G) = \mathbf{V}$. Abbiamo dunque dimostrato che qualunque valutazione soddisfa G, cioè che G è valida.

Per il teorema 2.35 abbiamo anche dimostrato che $\neg G$ è insoddisfacibile.

Gli esempi 2.36 e 2.37 mostrano una differenza essenziale tra le dimostrazioni di soddisfacibilità e quelle di validità: nelle prime è sufficiente costruire **una** interpretazione che soddisfa la formula in esame, nelle seconde si devono considerare **tutte** le possibili interpretazioni e, ragionando più astrattamente, mostrare che la formula è soddisfatta da ognuna di esse. Da questo punto di vista le dimostrazioni di insoddisfacibilità di una formula sono simili a quelle di validità: in questo caso si tratta di dimostrare che la formula non è soddisfatta in tutte le interpretazioni. Questa discussione è molto simile a quella svolta dopo l'esempio 2.17 a proposito delle dimostrazioni di conseguenza logica e di non conseguenza logica. In effetti c'è uno stretto legame tra conseguenza logica e validità/soddisfacibilità, come evidenziato dal prossimo lemma.

LEMMA 2.38. *Siano F e G formule.*

(a) *$F \models G$ se e solo se $F \to G$ è valida.*
(b) *$F \not\models G$ se e solo se $F \wedge \neg G$ è soddisfacibile.*

DIMOSTRAZIONE. (a) Supponiamo $F \models G$ e dimostriamo la validità di $F \to G$. Sia v un'interpretazione qualunque. Se $v(F) = \mathbf{F}$ allora $v(F \to G) = \mathbf{V}$. Se invece $v(F) = \mathbf{V}$ dalla nostra ipotesi segue $v(G) = \mathbf{V}$ e quindi v soddisfa $F \to G$ anche in questo caso.

Per l'implicazione inversa supponiamo che $F \to G$ sia valida e fissiamo un'interpretazione v che soddisfa F. Visto che $v(F \to G) = \mathbf{V}$ deve essere $v(G) = \mathbf{V}$: abbiamo dunque dimostrato che $F \models G$.

(b) Per (a) abbiamo che $F \not\models G$ se e solo se $F \to G$ non è valida. Per il teorema 2.35 quest'ultima condizione è equivalente alla soddisfacibilità di $\neg(F \to G)$. Dato che per il lemma 2.23.4 $\neg(F \to G) \equiv F \wedge \neg G$ la dimostrazione è completa. □

ESERCIZIO 2.39. Alcune delle seguenti affermazioni riguardanti le formule F e G sono corrette, altre no. Dimostrate le prime e trovate un controesempio alle seconde.

(a) Se sia F che G sono soddisfacibili allora esiste un'interpretazione v tale che $v(F \wedge G) = \mathbf{V}$.
(b) $G \not\models \neg F$ se e solo $F \wedge G$ è soddisfacibile.
(c) $F \models \neg G$ se e solo se $F \not\models G$.
(d) Se F non è valido allora $\neg F$ è valido.

Consideriamo ora validità e soddisfacibilità per insiemi di formule.

DEFINIZIONE 2.40. Se Γ è un insieme di formule diciamo che

- Γ è *valido* se ogni interpretazione soddisfa Γ, cioè soddisfa ogni $F \in \Gamma$;
- Γ è *soddisfacibile* se qualche interpretazione soddisfa Γ, cioè soddisfa ogni $F \in \Gamma$;
- Γ è *insoddisfacibile* se ogni interpretazione non soddisfa Γ, cioè non soddisfa qualche $F \in \Gamma$ (F può dipendere dall'interpretazione).

NOTA 2.41. Notiamo che un insieme di formule è valido se e solo se tutti i suoi elementi sono validi. La proprietà analoga non vale però per soddisfacibilità e insoddisfacibilità: l'insieme $\{p, \neg p\}$ è insoddisfacibile, pur essendo ognuno dei suoi elementi soddisfacibile. Se però un insieme di formule è soddisfacibile allora tutti i suoi elementi sono soddisfacibili. D'altra parte se una formula è insoddisfacibile, allora tutti gli insiemi che la contengono sono insoddisfacibili.

ESERCIZIO 2.42. ($\star$) Dimostrate che l'insieme

$$\Gamma = \{\, p_i \to \neg p_{i+1} \,:\, i \in \mathbb{N} \,\} \cup \{\, \neg p_i \to p_{i+1} \,:\, i \in \mathbb{N} \,\}$$

è soddisfacibile e che l'insieme $\Gamma \cup \{p_1, p_4\}$ è insoddisfacibile.

Nel caso in cui Γ è finito ci si può ricondurre al caso di una singola formula.

LEMMA 2.43. *Se $\Gamma = \{F_1, \dots, F_n\}$ è un insieme finito di formule allora la validità (soddisfacibilità, insoddisfacibilità) di Γ è equivalente alla validità (soddisfacibilità, insoddisfacibilità) della formula $F_1 \wedge \cdots \wedge F_n$.*

DIMOSTRAZIONE. Immediata dalle definizioni. □

NOTAZIONE 2.44. $\models F$ sta ad indicare $\emptyset \models F$, dove $\emptyset$ è l'insieme vuoto.

LEMMA 2.45. *Sia F una formula e Γ un insieme di formule.*

(a) *$\models F$ se e solo se F è valida.*
(b) *$\Gamma \models F$ se e solo se $\Gamma, \neg F$ è insoddisfacibile (per la notazione si veda 1.6).*

DIMOSTRAZIONE. (a) $\models F$ significa che per ogni interpretazione che soddisfa tutti gli elementi di $\emptyset$ soddisfa anche F. Ma qualunque interpretazione soddisfa tutti gli elementi di $\emptyset$ (perché non ce ne sono), e perciò questa condizione è equivalente alla validità di F.

(b) $\Gamma, \neg F$ insoddisfacibile significa che nessuna interpretazione soddisfa sia le formule di Γ che $\neg F$, cioè che se un'interpretazione v soddisfa tutte le formule di Γ non può essere $v(\neg F) = \mathbf{V}$. Questo è equivalente a affermare che se un'interpretazione soddisfa tutte le formule di Γ allora soddisfa anche F, cioè $\Gamma \models F$. □

4. Una procedura di decisione: le tavole di verità

Iniziamo specificando cosa intendiamo per procedura di decisione nel nostro contesto.

DEFINIZIONE 2.46. Sia $\mathcal{F}$ un insieme di formule proposizionali. Una *procedura di decisione per $\mathcal{F}$* è un algoritmo che riceve in input una formula F, termina sempre la sua esecuzione e fornisce l'output "sì" se $F \in \mathcal{F}$, l'output "no" se $F \notin \mathcal{F}$.

Un obiettivo della logica è individuare procedure di decisione per gli insiemi delle formule valide e delle formule soddisfacibili. Il teorema 2.35 mostra che una procedura di decisione per uno qualunque di questi insiemi può essere facilmente convertita in una procedura di decisione per l'altro. Ad esempio se abbiamo una procedura di decisione per la validità e vogliamo testare la soddisfacibilità di F basta applicare la procedura di decisione a $\neg F$ e ribaltare la risposta. Grazie al lemma 2.38 un tale algoritmo può essere anche applicato anche per verificare se una formula è conseguenza logica di un'altra. Il lemma 2.43 ci permette di utilizzare una procedura di decisione di questo tipo anche per insiemi finiti di formule.

Nella logica proposizionale esiste una procedura di decisione piuttosto semplice e naturale per la validità e per la soddisfacibilità. Il metodo delle tavole di verità si basa sul lemma 2.8: per stabilire se una valutazione v soddisfa una formula F basta conoscere i valori assunti da v sulle lettere proposizionali che compaiono in F. Dato che in F compaiono solo un numero finito di lettere proposizionali, per stabilire se F è valida è sufficiente considerare tutte le possibili combinazioni di valori di verità assegnati dalle valutazioni a queste lettere.

ALGORITMO 2.47. L'algoritmo delle tavole di verità prende in input una formula proposizionale F e esamina le lettere proposizionali che compaiono in F, che possiamo indicare con $p_1, \dots, p_n$. Si crea una tabella che contiene $n + 1$ colonne,

una per ogni p_i ed una per F, e 2^n righe. Le prime n colonne (quelle delle p_i) contengono **V** o **F** in modo che nelle 2^n righe compaia ogni possibile funzione che associa alle n lettere proposizionali i valori di verità (ciò si ottiene ad esempio se nella prima colonna 2^{n-1} **V** sono seguiti da 2^{n-1} **F**, nella seconda colonna si alternano 4 blocchi di lunghezza 2^{n-2} di **V** e di **F**, e così via fino alla colonna n-esima in cui il valore di verità cambia ad ogni riga). La colonna corrispondente a F contiene **V** o **F** a seconda se l'interpretazione generata dalla valutazione che compare in quella riga soddisfa o meno F.

Se la colonna di F contiene solo **V** allora F è valida, se contiene almeno un **V** allora F è soddisfacibile, se contiene solo **F** allora F è insoddisfacibile.

In pratica è comodo avere a disposizioni colonne supplementari, usualmente posizionate tra quelle delle lettere proposizionali e quella di F, in cui calcolare i valori di verità di opportune sottoformule di F.

ESEMPIO 2.48. Usiamo l'algoritmo delle tavole di verità per verificare che la formula $(p \to q) \vee (p \to \neg q)$ è valida.

p	q	$p \to q$	$\neg q$	$p \to \neg q$	$(p \to q) \vee (p \to \neg q)$
V	**V**	**V**	**F**	**F**	**V**
V	**F**	**F**	**V**	**V**	**V**
F	**V**	**V**	**F**	**V**	**V**
F	**F**	**V**	**V**	**V**	**V**

Come si nota l'ultima colonna è composta interamente da **V**.

ESEMPIO 2.49. Usiamo l'algoritmo delle tavole di verità per verificare che la formula (che per comodità indichiamo con F) $(\neg p \vee q) \wedge (q \to \neg r \wedge \neg p) \wedge (p \vee r)$ è soddisfacibile.

p	q	r	$\neg p \vee q$	$\neg r \wedge \neg p$	$q \to \neg r \wedge \neg p$	$p \vee r$	F
V	**V**	**V**	**V**	**F**	**F**	**V**	**F**
V	**V**	**F**	**V**	**F**	**F**	**V**	**F**
V	**F**	**V**	**F**	**F**	**V**	**V**	**F**
V	**F**	**F**	**F**	**F**	**V**	**V**	**F**
F	**V**	**V**	**V**	**F**	**F**	**V**	**F**
F	**V**	**F**	**V**	**V**	**V**	**F**	**F**
F	**F**	**V**	**V**	**F**	**V**	**V**	**V**
F	**F**	**F**	**V**	**V**	**V**	**F**	**F**

L'unica riga in cui nell'ultima colonna compare **V** corrisponde alla valutazione $v(p) = \mathbf{F}$, $v(q) = \mathbf{F}$, $v(r) = \mathbf{V}$, che definisce dunque un'interpretazione che soddisfa F.

L'algoritmo delle tavole di verità è piuttosto inefficiente: per stabilire se una formula con n lettere proposizionali è valida è necessario compilare 2^n righe. Esistono metodi di decisione per la validità delle formule proposizionali che molto spesso sono più efficienti (ad esempio quello cui è dedicato il capitolo 4), ma per ora non è stato trovato un algoritmo che sia più efficiente in ogni caso. In realtà l'esistenza di un metodo di decisione siffatto è equivalente ad una risposta positiva all'importante problema $P = NP$, di cui si parla in altri corsi.

ESEMPIO 2.50. Usiamo l'algoritmo delle tavole di verità per verificare che la formula $p \wedge \neg q \to p \wedge q$ è conseguenza logica di $\neg p$. Per il lemma 2.38 basta verificare che $\neg p \to (p \wedge \neg q \to p \wedge q)$ è valida. Una lieve semplificazione consiste nel calcolare le tavole di verità delle due formule e verificare che quando $\neg p$ è soddisfatta lo è anche $p \wedge \neg q \to p \wedge q$:

p	q	$\neg p$	$p \wedge \neg q$	$p \wedge q$	$p \wedge \neg q \to p \wedge q$
V	**V**	**F**	**F**	**V**	**V**
V	**F**	**F**	**V**	**F**	**F**
F	**V**	**V**	**F**	**F**	**V**
F	**F**	**V**	**F**	**F**	**V**

Un'ulteriore semplificazione consiste nel non calcolare il valore di verità di $p \wedge \neg q \to p \wedge q$ quando si è verificato che $v(\neg p) = \mathbf{F}$:

p	q	$\neg p$	$p \wedge \neg q$	$p \wedge q$	$p \wedge \neg q \to p \wedge q$
V	**V**	**F**			
V	**F**	**F**			
F	**V**	**V**	**F**	**F**	**V**
F	**F**	**V**	**F**	**F**	**V**

ESEMPIO 2.51. Usiamo l'algoritmo delle tavole di verità per verificare che le formule $p \to (q \wedge \neg q)$ e $\neg p$ sono logicamente equivalenti. Per i lemmi 2.19 e 2.38 basta dimostrare che $(p \to q \wedge \neg q) \to \neg p$ e $\neg p \to (p \to q \wedge \neg q)$ sono entrambe valide. Una strada più breve è calcolare le tavole di verità delle due formule e verificare che $p \to q \wedge \neg q$ e $\neg p$ hanno sempre lo stesso valore di verità:

p	q	$q \wedge \neg q$	$p \to q \wedge \neg q$	$\neg p$
V	**V**	**F**	**F**	**F**
V	**F**	**F**	**F**	**F**
F	**V**	**F**	**V**	**V**
F	**F**	**F**	**V**	**V**

ESERCIZIO 2.52. Dimostrare con le tavole di verità che $p \to (q \to r)$ e $p \wedge q \to r$ sono logicamente equivalenti.

ESERCIZIO 2.53. Stabilite se le seguenti formule sono valide, soddisfacibili, insoddisfacibili (usate sia le tavole di verità che le definizioni):

$$(p \to q) \wedge \neg q \to \neg p;$$
$$(p \to q) \to (p \to \neg q);$$
$$(p \vee q \to r) \vee p \vee q;$$
$$(p \vee q) \wedge (p \to r \wedge q) \wedge (q \to \neg r \wedge p);$$
$$(p \to (q \to r)) \to ((p \to q) \to (p \to r)).$$

ESERCIZIO 2.54. Stabilite se le seguenti conseguenze ed equivalenze logiche sono corrette (usate sia le tavole di verità che le definizioni):

$$p \to q \models \neg p \to \neg q;$$
$$(p \to q) \wedge \neg q \models \neg p;$$
$$(p \vee q) \wedge (\neg p \to \neg q) \equiv q;$$
$$(p \vee q) \wedge (\neg p \to \neg q) \equiv p.$$

5. Traduzioni dal linguaggio naturale

In questa sezione ci occuperemo di tradurre frasi del linguaggio naturale (nel nostro caso, l'italiano) in formule della logica proposizionale e viceversa.

ESEMPIO 2.55. Consideriamo un linguaggio proposizionale in cui p significa "Paola è contenta", q significa "Paola dipinge un quadro" e r significa "Renzo è contento". La formula proposizionale $p \wedge q \to \neg r$ viene interpretata come "se Paola è contenta e dipinge un quadro allora Renzo non è contento". La formula $p \to q$ viene interpretata come "se Paola è contenta allora dipinge un quadro", ma anche come "Paola è contenta soltanto se dipinge un quadro".

Queste due diverse traduzioni della stessa formula mostrano che l'italiano, come qualunque altro linguaggio naturale, consente di aggiungere sfumature che il linguaggio formale non è in grado di esprimere (un ulteriore esempio è fornito dal fatto che dal punto di vista logico non c'è differenza tra "e" e "ma"). D'altro canto il linguaggio formale permette una maggior precisione, ed evita le ambiguità tipiche dei linguaggi naturali.

La traduzione dal linguaggio formale al linguaggio naturale non presenta in genere difficoltà, mentre la direzione inversa è spesso più delicata.

ESEMPIO 2.56. Consideriamo un linguaggio proposizionale in cui p significa "Pietro sarà eletto leader del partito", r significa "Raffaella si dimetterà", m significa "Mario si dimetterà" e v significa "vinceremo le elezioni". La frase "vinceremo le elezioni, se Pietro sarà eletto leader del partito" può venir tradotta dalla formula proposizionale $p \to v$. La frase "solo se Pietro sarà eletto leader del partito vinceremo le elezioni" viene tradotta da $v \to p$ (la frase è equivalente ad affermare che se Pietro non verrà eletto leader del partito le elezioni saranno certamente perse, e infatti $v \to p \equiv \neg p \to \neg v$). La frase "se Pietro non sarà eletto leader del partito, allora Raffaella o Mario si dimetteranno e non vinceremo le elezioni" ha come traduzione $\neg p \to (r \vee m) \wedge \neg v$.

ESEMPIO 2.57. Consideriamo un linguaggio proposizionale in cui p significa "x è primo" e d significa "x è dispari". "Una condizione sufficiente perché x sia primo è che x sia dispari" viene tradotto come $d \to p$. "Una condizione necessaria perché x sia primo è che x sia dispari" viene tradotto come $p \to d$. "Una condizione necessaria e sufficiente perché x sia primo è che x sia dispari" viene tradotto come $(d \to p) \wedge (p \to d)$.

ESERCIZIO 2.58. Introducendo opportuni linguaggi proposizionali, traducete le frasi seguenti:

(a) "Se l'algoritmo termina abbiamo un risultato, e se abbiamo un risultato lo stampiamo", "se l'algoritmo termina abbiamo un risultato e lo stampiamo", "non è possibile che l'algoritmo non termini", "non è possibile che l'algoritmo termini ma non dia un risultato".
(b) "Patrizia va al cinema solo se Roberta ci va", "se Roberta va al cinema, anche Patrizia ci va", "al massimo una tra Roberta e Patrizia va al cinema".
(c) "Se il Signor Rossi è felice, la Signora Rossi è felice", "se il Signor Rossi è infelice, la Signora Rossi è infelice".

La logica proposizionale ha degli evidenti limiti espressivi, già notati all'inizio del capitolo 1. Malgrado questo è possibile utilizzarla per analizzare alcuni tipi di ragionamento e stabilirne la correttezza (o meno).

ESEMPIO 2.59. Supponiamo di sapere che:

- se Paolo è grasso allora Carlo non è biondo oppure Roberta non è alta;
- se Roberta è alta allora Sandra è magra;
- se Sandra è magra e Carlo è biondo allora Paolo è grasso;
- Carlo è biondo.

Possiamo dedurre che Roberta non è alta?

Se indichiamo "Paolo è grasso", "Carlo è biondo", "Roberta è alta" e "Sandra è magra" rispettivamente con p, q, r e s allora dobbiamo verificare se

$$p \to \neg q \vee \neg r, r \to s, s \wedge q \to p, q \models \neg r.$$

Lasciamo al lettore la verifica, utilizzando il ragionamento diretto basato sulle definizioni, oppure le tavole di verità (che richiedono 16 righe).

CAPITOLO 3

Forma normale congiuntiva e disgiuntiva

In questo capitolo ci occuperemo di trasformare una formula proposizionale in una formula ad essa logicamente equivalente che abbia una forma particolarmente semplice dal punto di vista sintattico. Questa trasformazione è di tipo algoritmico, e l'algoritmo che introdurremo è non deterministico (ad ogni passo abbiamo diverse scelte possibili), ma gode della proprietà della terminazione forte (termina qualunque sia la successione delle scelte).

1. Definizione di forma normale congiuntiva e disgiuntiva

DEFINIZIONE 3.1. Un *letterale* è una lettera proposizionale oppure la negazione di una lettera proposizionale.

DEFINIZIONE 3.2. Una formula proposizionale è in *forma normale congiuntiva* se è della forma $F_1 \wedge \cdots \wedge F_m$, dove per $1 \leq i \leq m$, F_i è della forma $G_{i,1} \vee \cdots \vee G_{i,h_i}$, dove per $1 \leq j \leq h_i$, $G_{i,j}$ è un letterale.

Una formula proposizionale è in *forma normale disgiuntiva* se è della forma $F_1 \vee \cdots \vee F_m$, dove per $1 \leq i \leq m$, F_i è della forma $G_{i,1} \wedge \cdots \wedge G_{i,h_i}$, dove per $1 \leq j \leq h_i$, $G_{i,j}$ è un letterale.

ESEMPIO 3.3. La formula $(p \vee \neg q \vee r) \wedge p \wedge \neg s$ è in forma normale congiuntiva con $m = 3$ e $h_1 = 3$, $h_2 = h_3 = 1$.

La formula $\neg p \vee (\neg q \wedge \neg r \wedge s) \vee (p \wedge \neg s)$ è in forma normale disgiuntiva con $m = 3$ e $h_1 = 1$, $h_2 = 3$, $h_3 = 2$.

La formula $p \wedge \neg r$ è sia in forma normale congiuntiva che in forma normale disgiuntiva (nel primo caso $m = 2$ e $h_1 = h_2 = 1$, nel secondo caso $m = 1$ e $h_1 = 2$).

La formula $p \vee \neg\neg q$ non è né in forma normale congiuntiva né in forma normale disgiuntiva (perché $\neg\neg q$ non è un letterale).

La formula $p \wedge \neg(q \vee r)$ non è né in forma normale congiuntiva né in forma normale disgiuntiva.

Ogni formula che contiene $\rightarrow$ non è né in forma normale congiuntiva né in forma normale disgiuntiva.

L'obiettivo di questo capitolo è dimostrare il seguente teorema.

TEOREMA 3.4. *Ogni formula proposizionale F può essere trasformata in due formule G_1 e G_2, la prima in forma normale congiuntiva e la seconda in forma normale disgiuntiva, tali che*

$$F \equiv G_1 \qquad e \qquad F \equiv G_2.$$

L'espressione "può essere trasformata" nell'enunciato del teorema è stata usata per asserire qualcosa di più della semplice esistenza di G_1 e G_2: intendiamo dire che il teorema verrà dimostrato attraverso la descrizione di due algoritmi che preso come input F, forniscono come output rispettivamente G_1 e G_2.

Il teorema 3.4 non asserisce l'unicità di G_1 e G_2. Ad esempio è immediato verificare che $p \equiv (p \vee q) \wedge (p \vee \neg q)$ ed abbiamo due formule in forma normale congiuntiva che sono logicamente equivalenti.

2. Doppie negazioni, α-formule e β-formule

Gli algoritmi che descriveremo per dimostrare il teorema 3.4 sono dovuti a Melvin Fitting e si basano su una classificazione delle formule proposizionali che ci sarà utile anche altrove.

DEFINIZIONE 3.5. Una formula F è una *doppia negazione* se è del tipo $\neg\neg G$ per qualche formula G. In questo caso diciamo che G è il *ridotto* di F.

Usando il lemma 2.23 si può mostrare che ogni formula che non è un letterale è una doppia negazione oppure è equivalente ad una congiunzione o ad una disgiunzione. Per formulare questa osservazione in forma compatta (come faremo nei lemmi 3.7 e 3.8) introduciamo la seguente definizione.

DEFINIZIONE 3.6. Una formula è *una α-formula* se esistono F e G tali che la formula è di uno dei tipi che compaiono nella colonna sinistra della prima delle seguenti tabelle. Una formula è *una β-formula* se esistono F e G tali che la formula è di uno dei tipi che compaiono nella colonna sinistra della seconda delle seguenti tabelle. In entrambi i casi i *ridotti* di una α- o β-formula sono le formule che compaiono nelle due colonne più a destra.

α-formula		
$F \wedge G$	F	G
$\neg(F \vee G)$	$\neg F$	$\neg G$
$\neg(F \to G)$	F	$\neg G$

β-formula		
$F \vee G$	F	G
$\neg(F \wedge G)$	$\neg F$	$\neg G$
$F \to G$	$\neg F$	G

LEMMA 3.7. *Ogni α-formula è logicamente equivalente alla congiunzione dei suoi ridotti. Ogni β-formula è logicamente equivalente alla disgiunzione dei suoi ridotti.*

DIMOSTRAZIONE. Per le formule contenute nelle prime righe di ognuna delle due tabelle l'affermazione è ovvia, per le altre è contenuta nel lemma 2.23. □

LEMMA 3.8. *Una formula proposizionale è di uno e uno solo dei tipi seguenti:*

- *un letterale,*
- *una doppia negazione,*
- *una α-formula,*
- *una β-formula.*

DIMOSTRAZIONE. Per il lemma 1.7 ogni formula F è una lettera proposizionale, una negazione, una congiunzione, una disgiunzione o un'implicazione.

- Se F è una lettera proposizionale allora è un letterale;
- Se F è una negazione $\neg G$ dobbiamo considerare di che tipo è G:
  - se G è una lettera proposizionale allora F è un letterale;
  - se G è una negazione allora F è una doppia negazione;
  - se G è una congiunzione allora F è una β-formula;
  - se G è una disgiunzione oppure un'implicazione allora F è una α-formula.
- Se F è una congiunzione allora è una α-formula.
- Se F è una disgiunzione oppure un'implicazione allora è una β-formula.

□

3. Gli algoritmi di Fitting

Conviene ricordare alcune equivalenze logiche dimostrate nel capitolo 2 (lemmi 2.20 e 2.23), che saranno utili per giustificare i passaggi degli algoritmi di Fitting:

LEMMA 3.9. *Siano F, G e H formule proposizionali qualunque. Valgono le seguenti equivalenze logiche:*

(1) $\neg\neg F \equiv F$;
(2) $F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H)$;
(3) $(G \vee H) \wedge F \equiv (G \wedge F) \vee (H \wedge F)$;
(4) $F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H)$;
(5) $(G \wedge H) \vee F \equiv (G \vee F) \wedge (H \vee F)$.

Per descrivere gli algoritmi di Fitting useremo la seguente notazione.

Convenzione 3.10. Visto che $\wedge$ è sia commutativo che associativo tutte le formule ottenute combinando in un ordine qualunque $G_1, \ldots, G_n$ attraverso $\wedge$ sono logicamente equivalenti a $G_1 \wedge \cdots \wedge G_n$ (che, per la convenzione 1.13.4 sulla scrittura delle formule, è $(((G_1 \wedge G_2) \wedge \ldots G_{n-1}) \wedge G_n)$). Quest'ultima formula è chiamata la *congiunzione generalizzata* di $G_1, \ldots, G_n$ ed è indicata con $\langle G_1, \ldots, G_n \rangle$.

Similmente, a partire da commutatività e associatività di $\vee$, si definisce la *disgiunzione generalizzata* di $G_1, \ldots, G_n$, che è indicata con $[G_1, \ldots, G_n]$.

Usando congiunzioni e disgiunzioni generalizzate, una formula in forma normale congiuntiva ha la forma

$$\langle [G_{1,1}, \ldots, G_{1,h_1}], \ldots, [G_{m,1}, \ldots, G_{m,h_m}] \rangle$$

dove ogni $G_{i,j}$ è un letterale, mentre una formula in forma normale disgiuntiva ha la forma

$$[\langle G_{1,1}, \ldots, G_{1,h_1} \rangle, \ldots, \langle G_{m,1}, \ldots, G_{m,h_m} \rangle]$$

con ogni $G_{i,j}$ letterale.

Algoritmo 3.11. L'algoritmo di Fitting per la trasformazione in forma normale congiuntiva prende in input una formula proposizionale F e la considera come una congiunzione generalizzata di una disgiunzione generalizzata: $\langle [F] \rangle$. Ad ogni passo dell'algoritmo avremo una congiunzione generalizzata di disgiunzioni generalizzate di formule. Se tutti gli elementi di queste disgiunzioni generalizzate sono letterali la formula è in forma normale congiuntiva e l'algoritmo si arresta. Se esistono elementi di queste disgiunzioni generalizzate che non sono letterali se ne sceglie uno, che indichiamo con G. Per il lemma 3.8 ci sono tre possibilità:

(1) G è una doppia negazione con ridotto H: in questo caso si sostituisce G con H nel congiunto in cui appariva G; gli altri congiunti restano immutati.
(2) G è una β-formula e i suoi ridotti sono G_1 e G_2: in questo caso si sostituisce G con G_1, G_2 nel congiunto in cui appariva G; gli altri congiunti restano immutati.
(3) G è una α-formula e i suoi ridotti sono G_1 e G_2: in questo caso si sostituisce il congiunto in cui appariva G con due nuovi congiunti; nel primo congiunto G è sostituito da G_1, nel secondo congiunto G è sostituito da G_2 (e in entrambi i casi gli altri disgiunti restano immutati); gli altri congiunti restano immutati.

La congiunzione generalizzata di disgiunzioni generalizzate di formule così ottenuta è sempre logicamente equivalente a quella precedente: nel primo caso per la (1) del lemma 3.9, nel secondo per il lemma 3.7, nel terzo per il lemma 3.7 e per (4) e (5) del lemma 3.9.

Esempio 3.12. Applichiamo l'algoritmo 3.11 alla formula

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

Sulla sinistra abbiamo la formula (congiunzione generalizzata di disgiunzioni generalizzate di formule) a cui siamo arrivati ad ogni passo dell'applicazione dell'algoritmo, nella colonna centrale indichiamo la formula (che non è un letterale) su cui agiamo per effettuare il passo successivo, nella colonna di sinistra il tipo di formula

(α, β o doppia negazione, indicata da $\neg\neg$).

$\langle[(r \wedge \neg s) \vee \neg(p \to \neg q)]\rangle$	$(r \wedge \neg s) \vee \neg(p \to \neg q)$	β
$\langle[r \wedge \neg s, \neg(p \to \neg q)]\rangle$	$r \wedge \neg s$	α
$\langle[r, \neg(p \to \neg q)], [\neg s, \neg(p \to \neg q)]\rangle$	$\neg(p \to \neg q)$	α
$\langle[r, p], [r, \neg\neg q], [\neg s, \neg(p \to \neg q)]\rangle$	$\neg\neg q$	$\neg\neg$
$\langle[r, p], [r, q], [\neg s, \neg(p \to \neg q)]\rangle$	$\neg(p \to \neg q)$	α
$\langle[r, p], [r, q], [\neg s, p], [\neg s, \neg\neg q]\rangle$	$\neg\neg q$	$\neg\neg$
$\langle[r, p], [r, q], [\neg s, p], [\neg s, q]\rangle$		

La formula di partenza è quindi equivalente alla formula in forma normale congiuntiva $\langle[r, p], [r, q], [\neg s, p], [\neg s, q]\rangle$, cioè a

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

Esempio 3.13. Applichiamo l'algoritmo 3.11 alla formula

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

Questa volta omettiamo l'indicazione della formula su cui si effettua la riduzione.

$\langle[(p \to \neg q) \vee \neg(r \wedge s \to \neg(\neg r \vee t))]\rangle$	β
$\langle[p \to \neg q, \neg(r \wedge s \to \neg(\neg r \vee t))]\rangle$	β
$\langle[\neg p, \neg q, \neg(r \wedge s \to \neg(\neg r \vee t))]\rangle$	α
$\langle[\neg p, \neg q, r \wedge s], [\neg p, \neg q, \neg\neg(\neg r \vee t)]\rangle$	$\neg\neg$
$\langle[\neg p, \neg q, r \wedge s], [\neg p, \neg q, \neg r \vee t]\rangle$	α
$\langle[\neg p, \neg q, r], [\neg p, \neg q, s], [\neg p, \neg q, \neg r \vee t]\rangle$	β
$\langle[\neg p, \neg q, r], [\neg p, \neg q, s], [\neg p, \neg q, \neg r, t]\rangle$	

La formula in forma normale congiuntiva ottenuta è

$$(\neg p \vee \neg q \vee r) \wedge (\neg p \vee \neg q \vee s) \wedge (\neg p \vee \neg q \vee \neg r \vee t).$$

Esercizio 3.14. Applicate l'algoritmo 3.11 alla formula

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

L'algoritmo di Fitting per la trasformazione in forma normale disgiuntiva è duale a quello per la forma normale congiuntiva: congiunzioni generalizzate e disgiunzioni generalizzate sono scambiate e il ruolo delle α-formule e delle β-formule è invertito.

Algoritmo 3.15. L'algoritmo di Fitting per la trasformazione in forma normale disgiuntiva prende in input una formula proposizionale F e la considera come una disgiunzione generalizzata di una congiunzione generalizzata: $[\langle F\rangle]$. Ad ogni passo dell'algoritmo avremo una disgiunzione generalizzata di congiunzioni generalizzate di formule. Se tutti gli elementi di queste congiunzioni generalizzate sono letterali la formula è in forma normale disgiuntiva e l'algoritmo si arresta. Se esistono elementi di queste congiunzioni generalizzate che non sono letterali se ne sceglie uno, che indichiamo con G. Per il lemma 3.8 ci sono tre possibilità:

(1) G è una doppia negazione con ridotto H: in questo caso si sostituisce G con H nel disgiunto in cui appariva G; gli altri disgiunti restano immutati.
(2) G è una α-formula e i suoi ridotti sono G_1 e G_2: in questo caso si sostituisce G con G_1, G_2 nel disgiunto in cui appariva G; gli altri disgiunti restano immutati.
(3) G è una β-formula e i suoi ridotti sono G_1 e G_2: in questo caso si sostituisce il disgiunto in cui appariva G con due nuovi disgiunti; nel primo disgiunto G è sostituito da G_1, nel secondo disgiunto G è sostituito da G_2 (e in entrambi i casi gli altri congiunti restano immutati); gli altri disgiunti restano immutati.

La disgiunzione generalizzata di congiunzioni generalizzate di formule così ottenuta è sempre logicamente equivalente a quella precedente: nel primo caso per la (1) del lemma 3.9, nel secondo per il lemma 3.7, nel terzo per il lemma 3.7 e per (2) e (3) del lemma 3.9.

ESEMPIO 3.16. Applichiamo l'algoritmo 3.15 alla formula dell'esempio 3.12.

$$\begin{array}{c|c|c} [\langle (r \wedge \neg s) \vee \neg(p \to \neg q)\rangle] & (r \wedge \neg s) \vee \neg(p \to \neg q) & \beta \\ [\langle r \wedge \neg s\rangle, \langle \neg(p \to \neg q)\rangle] & r \wedge \neg s & \alpha \\ [\langle r, \neg s\rangle, \langle \neg(p \to \neg q)\rangle] & \neg(p \to \neg q) & \alpha \\ [\langle r, \neg s\rangle, \langle p, \neg\neg q\rangle] & \neg\neg q & \neg\neg \\ [\langle r, \neg s\rangle, \langle p, q\rangle] & & \end{array}$$

La formula di partenza è quindi equivalente alla formula in forma normale disgiuntiva $[\langle r, \neg s\rangle, \langle p, q\rangle]$, cioè a $(r \wedge \neg s) \vee (p \wedge q)$.

ESEMPIO 3.17. Applichiamo l'algoritmo 3.15 alla formula dell'esempio 3.13.

$$\begin{array}{c|c} [\langle (p \to \neg q) \vee \neg(r \wedge s \to \neg(\neg r \vee t))\rangle] & \beta \\ [\langle p \to \neg q\rangle, \langle \neg(r \wedge s \to \neg(\neg r \vee t))\rangle] & \beta \\ [\langle \neg p\rangle, \langle \neg q\rangle, \langle \neg(r \wedge s \to \neg(\neg r \vee t))\rangle] & \alpha \\ [\langle \neg p\rangle, \langle \neg q\rangle, \langle r \wedge s, \neg\neg(\neg r \vee t)\rangle] & \neg\neg \\ [\langle \neg p\rangle, \langle \neg q\rangle, \langle r \wedge s, \neg r \vee t\rangle] & \alpha \\ [\langle \neg p\rangle, \langle \neg q\rangle, \langle r, s, \neg r \vee t\rangle] & \beta \\ [\langle \neg p\rangle, \langle \neg q\rangle, \langle r, s, \neg r\rangle, \langle r, s, t\rangle] & \end{array}$$

La formula in forma normale disgiuntiva ottenuta è

$$\neg p \vee \neg q \vee (r \wedge s \wedge \neg r) \vee (r \wedge s \wedge t).$$

ESERCIZIO 3.18. Applicate l'algoritmo 3.15 alla formula dell'esercizio 3.14.

ESERCIZIO 3.19. Applicate gli algoritmi 3.11 e 3.15 alle formule:

$$\begin{array}{cc} (p \to q) \to (q \to \neg r); & \neg(p \wedge q \wedge (r \to s)); \\ \neg((p \to q) \wedge (q \to p)); & \neg(p \to \neg q) \to (p \wedge q); \\ (\neg q \to p) \vee \neg(s \wedge q \to \neg p); & \neg(\neg(p \to q) \wedge (r \vee s \to \neg t)). \end{array}$$

Entrambi gli algoritmi di Fitting (3.11 e 3.15) sono non deterministici.

ESEMPIO 3.20. La seguente tabella presenta due diverse applicazioni dell'algoritmo 3.11 alla formula $(p \wedge q) \vee \neg(r \wedge s)$:

$$\begin{array}{c|c} \langle [(p \wedge q) \vee \neg(r \wedge s)]\rangle & \langle [(p \wedge q) \vee \neg(r \wedge s)]\rangle \\ \langle [p \wedge q, \neg(r \wedge s)]\rangle & \langle [p \wedge q, \neg(r \wedge s)]\rangle \\ \langle [p \wedge q, \neg r, \neg s]\rangle & \langle [p, \neg(r \wedge s)], [q, \neg(r \wedge s)]\rangle \\ \langle [p, \neg r, \neg s], [q, \neg r, \neg s]\rangle & \langle [p, \neg r, \neg s], [q, \neg(r \wedge s)]\rangle \\ & \langle [p, \neg r, \neg s], [q, \neg r, \neg s]\rangle \end{array}$$

La forma normale congiuntiva ottenuta è la stessa in entrambi i casi ma nella prima colonna al secondo passo si è operato sulla β-formula $\neg(r \wedge s)$, mentre nella seconda colonna si è operato sulla α-formula $p \wedge q$. Questo ha portato ad ottenere il risultato finale in tre e in quattro passi rispettivamente.

ESERCIZIO 3.21. Trovate due diverse applicazioni dell'algoritmo 3.15 alla formula

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

NOTA 3.22. Per ridurre i tempi di esecuzione dell'algoritmo 3.11 per la forma normale congiuntiva è opportuno operare su β-formule ogniqualvolta ciò sia possibile. Dualmente, per ridurre i tempi di esecuzione dell'algoritmo 3.15 per la forma normale disgiuntiva è opportuno operare, ove possibile, su α-formule.

4. Terminazione forte degli algoritmi di Fitting

Abbiamo descritto gli algoritmi di Fitting 3.11 e 3.15, e abbiamo osservato che se arrivano a una formula in forma normale congiuntiva (risp. disgiuntiva), e quindi si fermano, allora la formula finale è logicamente equivalente alla formula da cui sono partiti. Ciò che non abbiamo ancora dimostrato, ma stiamo per fare, è che essi si fermano sempre.

DEFINIZIONE 3.23. Associamo ad ogni formula proposizionale un numero naturale positivo che chiameremo rango, secondo la seguente definizione ricorsiva (le cui clausole comprendono tutte le possibilità, per il lemma 3.8):

- se F è un letterale allora $\mathrm{rg}(F) = 1$;
- se F è una doppia negazione con ridotto G allora $\mathrm{rg}(F) = \mathrm{rg}(G) + 1$;
- se F è una α-formula o una β-formula e ha ridotti F_1 e F_2 allora $\mathrm{rg}(F) = \mathrm{rg}(F_1) + \mathrm{rg}(F_2) + 1$.

LEMMA 3.24. *Gli algoritmi 3.11 e 3.15 godono della proprietà della terminazione forte, cioè terminano qualunque sia la formula su cui si decide di operare ad ogni singolo passo.*

DIMOSTRAZIONE. Dimostreremo la terminazione forte per l'algoritmo 3.11: la dimostrazione per l'algoritmo 3.15 si ottiene con semplici modifiche.

Ad ogni congiunzione generalizzata di disgiunzioni generalizzate

$$H = \langle [F_{1,1}, \dots, F_{1,h_1}], \dots, [F_{m,1}, \dots, F_{m,h_m}] \rangle$$

associamo il numero naturale

$$\rho(H) = 2^{\mathrm{rg}(F_{1,1})+\dots+\mathrm{rg}(F_{1,h_1})} + \dots + 2^{\mathrm{rg}(F_{m,1})+\dots+\mathrm{rg}(F_{m,h_m})}.$$

Dimostreremo ora che, se un'applicazione di una delle tre regole dell'algoritmo 3.11 ci fa passare da una congiunzione generalizzata di disgiunzioni generalizzate H ad un'altra H', allora $\rho(H') < \rho(H)$. Nel caso si agisca su una doppia negazione o su una β-formula la verifica di ciò è immediata. Quando si agisce su una α-formula ciò dipende dalla seguente disuguaglianza, che vale per ogni $a, b > 0$: $2^a + 2^b < 2^{a+b+1}$ (per dimostrare la disuguaglianza supponete $a \geq b$ e osservate che $2^a + 2^b \leq 2^a + 2^a = 2^{a+1} < 2^{a+b+1}$). Perciò se F è una α-formula con ridotti F_1 e F_2, si ha $2^{\mathrm{rg}(F_1)} + 2^{\mathrm{rg}(F_2)} < 2^{\mathrm{rg}(F)}$ e quindi, moltiplicando entrambi i membri della disuguaglianza per 2^n, anche $2^{\mathrm{rg}(F_1)+n} + 2^{\mathrm{rg}(F_2)+n} < 2^{\mathrm{rg}(F)+n}$ per ogni numero naturale n.

Se $H_1, H_2, \dots$ sono le congiunzioni generalizzate di disgiunzioni generalizzate prodotte da un'esecuzione dell'algoritmo 3.11 si ha quindi $\rho(H_1) > \rho(H_2) > \cdots$ e non possono esistere infiniti passi, cioè l'algoritmo termina. □

NOTA 3.25. Notiamo esplicitamente che gli algoritmi di Fitting **non** prevedono la sostituzione di una formula con un'altra logicamente equivalente ad essa. Se introducessimo questa possibilità, l'algoritmo non godrebbe più della proprietà della terminazione forte e sarebbe quindi assai meno utile.

La dimostrazione del lemma 3.24 fornisce un limite superiore al numero di passi necessario all'algoritmo 3.11 per terminare: dato che la prima congiunzione generalizzata di disgiunzioni generalizzate è $\langle [F] \rangle$ e $\rho(\langle [F] \rangle) = 2^{\mathrm{rg}(F)}$, certamente l'algoritmo termina entro $2^{\mathrm{rg}(F)}$ passi. Questo limite superiore è molto grossolano: se F è la formula dell'esempio 3.12 si ha $\mathrm{rg}(F) = 8$ e quindi $\rho(\langle [F] \rangle) = 256$, ma l'algoritmo termina in 6 passi.

ESERCIZIO 3.26. Calcolate $\rho(H)$ dove H rappresenta le varie congiunzioni generalizzate di disgiunzioni generalizzate ottenute nell'esempio 3.12.

Possiamo completare il nostro lavoro sulle trasformazioni in forma normale congiuntiva e disgiuntiva con la:

DIMOSTRAZIONE DEL TEOREMA 3.4. Data una formula F applichiamo a partire da F gli algoritmi 3.11 e 3.15. Per il lemma 3.24 essi terminano, producendo due formule G_1 e G_2, la prima in forma normale congiuntiva, la seconda in forma normale disgiuntiva. Entrambe queste formule sono logicamente equivalenti a F. □

CAPITOLO 4

Il metodo dei tableaux: caso proposizionale

Il metodo dei tableaux proposizionali è una procedura di decisione relativamente efficiente per la soddisfacibilità delle formule proposizionali. Il principio che ispira questo algoritmo è piuttosto semplice: cerchiamo sistematicamente un'interpretazione che soddisfi la formula in esame. Se la ricerca ha successo la formula sarà soddisfacibile, altrimenti la sistematicità della ricerca ci assicura che essa è insoddisfacibile. Come già gli algoritmi del capitolo 3, l'algoritmo è non deterministico e possiede la proprietà della terminazione forte.

1. Coppie complementari

Partiamo da una formula F e, supponendo l'esistenza di un'interpretazione v che la soddisfi, esaminiamo quali altri caratteristiche deve avere v: ad esempio se F è una congiunzione $G \wedge H$ dovremo avere $v(G) = \mathbf{V}$ e $v(H) = \mathbf{V}$. A questo punto esaminiamo quali conseguenze hanno queste prime conseguenze e così via, passando ad esaminare formule via via più semplici fino a raggiungere i letterali (definizione 3.1). Riconosceremo la non esistenza di un'interpretazione con le caratteristiche richieste se ci troveremo ad avere $v(p) = \mathbf{V}$ e $v(\neg p) = \mathbf{V}$ per qualche lettera proposizionale p. Per semplificare la descrizione di questa situazione introduciamo la seguente definizione.

Definizione 4.1. Se p è una lettera proposizionale $\{p, \neg p\}$ è una *coppia complementare di letterali*. Più in generale se F è una formula $\{F, \neg F\}$ è una *coppia complementare*. Diciamo che F e $\neg F$ sono ciascuno il *complemento* dell'altro.

La proprietà fondamentale delle coppie complementari di letterali è contenuta nel seguente lemma.

Lemma 4.2. *Un insieme di letterali è soddisfacibile se e solo se non contiene nessuna coppia complementare.*

Dimostrazione. È ovvio che un insieme soddisfacibile di formule non può contenere coppie complementari.

Viceversa sia Γ un insieme di letterali che non contiene nessuna coppia complementare. Definiamo un'interpretazione v ponendo

$$v(p) = \begin{cases} \mathbf{V} & \text{se } p \in \Gamma; \\ \mathbf{F} & \text{se } p \notin \Gamma. \end{cases}$$

v soddisfa tutte le formule di Γ. Infatti se $\neg p \in \Gamma$ non può essere $p \in \Gamma$ (perché Γ non contiene coppie complementari) e quindi $v(p) = \mathbf{F}$, cioè $v(\neg p) = \mathbf{V}$. □

Nota 4.3. Notiamo che la direzione da destra a sinistra del lemma 4.2 è falsa se a "insieme di letterali" sostituiamo "insieme di formule": $\{p \wedge \neg p\}$ non contiene coppie complementari, ma è insoddisfacibile. La direzione da sinistra a destra è invece vera anche per insiemi di formule (nella dimostrazione di quella direzione non si è fatto alcun uso del fatto che gli elementi dell'insieme fossero tutti letterali).

2. Esempi preliminari

Prima di descrivere l'algoritmo dei tableaux esaminiamo in dettaglio un paio di esempi.

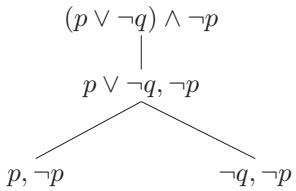
ESEMPIO 4.4. Sia F la formula $(p \vee \neg q) \wedge \neg p$. Se v soddisfa F deve essere $v(p \vee \neg q) = \mathbf{V}$ e $v(\neg p) = \mathbf{V}$. Dalla prima di queste proprietà possiamo dedurre che $v(p) = \mathbf{V}$ oppure $v(\neg q) = \mathbf{V}$. La prima possibilità ci conduce a dover soddisfare la coppia complementare di letterali $\{p, \neg p\}$, che è impossibile. Resta la seconda possibilità, che ci chiede di soddisfare l'insieme di letterali $\{\neg q, \neg p\}$ che non contiene nessuna coppia complementare: ciò è possibile per il lemma 4.2. Dall'insieme soddisfacibile di letterali possiamo "leggere" un'interpretazione che soddisfa F: $v(p) = \mathbf{F}$ e $v(q) = \mathbf{F}$.

ESEMPIO 4.5. Sia F la formula $(p \to \neg q) \wedge (p \wedge q)$. Se v soddisfa F deve essere $v(p \to \neg q) = \mathbf{V}$ e $v(p \wedge q) = \mathbf{V}$. La seconda condizione implica $v(p) = \mathbf{V}$ e $v(q) = \mathbf{V}$. Ricordando che $F \to G \equiv \neg F \vee G$ otteniamo che la prima condizione è soddisfatta se e solo se $v(\neg p) = \mathbf{V}$ oppure $v(\neg q) = \mathbf{V}$. Nel primo caso dovremmo soddisfare l'insieme $\{\neg p, p, q\}$, nel secondo caso l'insieme $\{\neg q, p, q\}$. Entrambi questi insiemi contengono una coppia complementare e sono quindi insoddisfacibili per il lemma 4.2. Perciò F è insoddisfacibile.

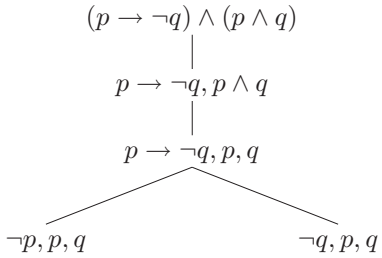
Se abbiamo a che fare con formule più complesse il filo del ragionamento svolto nei due esempi precedenti può diventare piuttosto difficile da seguire. Il metodo dei tableaux rappresenta questi ragionamenti in una forma più facilmente leggibile attraverso una struttura ad albero.

La radice dell'albero è etichettata con la formula che si intende studiare, gli altri nodi con insiemi di formule. Inoltre una foglia (o nodo terminale) dell'albero è etichettata con un insieme di letterali.

ESEMPIO 4.6. Disegniamo il tableau relativo alla formula studiata nell'esempio 4.4:

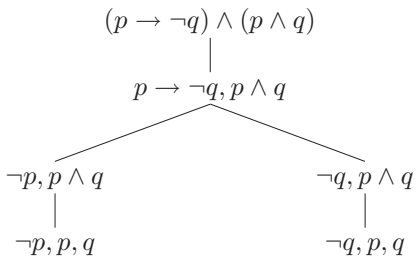


e quello relativo alla formula studiata nell'esempio 4.5:



Il primo tableau contiene una foglia (quella di destra) etichettata con un insieme di letterali che non contiene coppie complementari: questo testimonia la soddisfacibilità della formula originaria. Il secondo tableau contiene solo foglie etichettate con insiemi di letterali che contengono coppie complementari: ciò implica che la formula di partenza è insoddisfacibile.

ESEMPIO 4.7. La costruzione del tableau non è in generale unica. Ecco un altro tableau per la formula dell'esempio 4.5:



Il nuovo tableau corrisponde ad un inversione nell'ordine del ragionamento: si è prima considerato il significato della soddisfazione di $p \to \neg q$ e solo successivamente, per ognuna delle due possibilità, il significato di $p \wedge q$. Notiamo comunque che anche in questo caso il tableau contiene solo foglie etichettate con insiemi di letterali che contengono coppie complementari.

Gli alberi costruiti in precedenza evidenziano una caratteristica dei tableaux: ci sono nodi che hanno un solo figlio e nodi che ne hanno due. I primi corrispondono all'analisi di formule come $p \wedge q$, i secondi all'analisi di $p \vee \neg q$ o $p \to \neg q$. Il metodo più comodo per introdurre precisamente questa distinzione è quello di ricorrere alla distinzione tra α-formule e β-formule utilizzata già nel capitolo precedente (si ricordino la definizione 3.6 e i lemmi 3.7 e 3.8).

3. L'algoritmo

ALGORITMO 4.8. Un tableau per una formula F è un albero in cui ogni nodo è etichettato con un insieme finito di formule. Il tableau è costruito per stadi $T_0, \ldots, T_k$: per ogni i, T_{i+1} è un albero che estende T_i aggiungendo uno o due nodi con le rispettive etichette e lasciando invariate le etichette dei nodi già appartenenti a T_i. L'albero T_k (k ovviamente dipende da F) è il tableau per F. Se n è un nodo di qualche T_i indichiamo con $E(n)$ l'etichetta di n (che, per quanto detto prima, è la stessa per tutti i T_i cui appartiene n).

All'inizio della costruzione T_0 consiste di un solo nodo (la *radice* dell'albero) etichettato con $\{F\}$. Allo stadio i cerchiamo una foglia n dell'albero T_i tale che $E(n)$ non sia un insieme di letterali. Se una tale foglia non esiste siamo al termine della costruzione e il tableau è completo. Altrimenti fissiamo n e scegliamo una formula $G \in E(n)$ che non è un letterale. Per il lemma 3.8 ci sono tre possibilità:

(1) se G è una doppia negazione con ridotto G_1 aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$;
(2) se G è una α-formula con ridotti G_1 e G_2 aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1, G_2\}$;
(3) se G è una β-formula con ridotti G_1 e G_2 aggiungiamo due nodi tra loro inconfrontabili n_1 e n_2 sotto n e poniamo $E(n_1) = (E(n) \setminus \{G\}) \cup \{G_1\}$ e $E(n_2) = (E(n) \setminus \{G\}) \cup \{G_2\}$.

Notiamo che in ogni caso n non è una foglia di T_{i+1}.

L'algoritmo che abbiamo appena descritto è non deterministico perché ad ogni passo scegliamo una foglia n non etichettata solo con letterali e, all'interno di $E(n)$, scegliamo una formula che non sia un letterale.

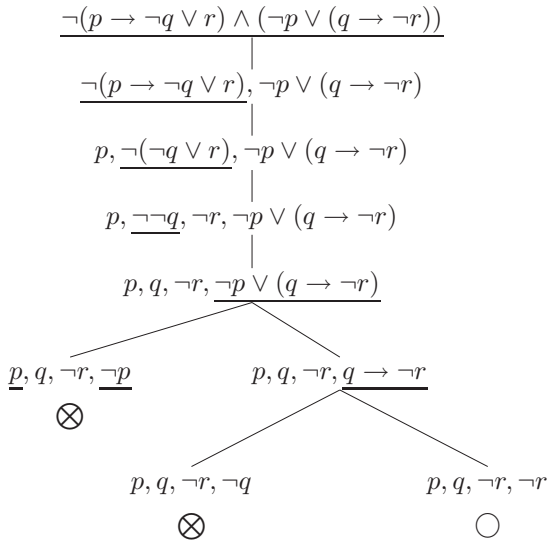
In pratica gli alberi $T_0, \ldots T_k$ non sono rappresentati separatamente: si deve piuttosto pensare che il tableau "cresce" verso la sua forma finale, che usualmente è l'unica che vediamo.

DEFINIZIONE 4.9. Sia n un nodo del tableau che ha successori nel tableau: la *formula su cui si agisce in* n è la G della descrizione dell'algoritmo.

Convenzione 4.10. Per comodità di lettura aggiungeremo sotto le foglie del tableau uno dei simboli $\otimes$ e $\bigcirc$: se l'etichetta della foglia contiene una coppia complementare di letterali useremo $\otimes$, altrimenti $\bigcirc$. Questo ci permette di evidenziare meglio se il tableau contiene solo foglie etichettate con insiemi insoddisfacibili di letterali, oppure se c'è qualche foglia etichettata con un insieme soddisfacibile di letterali (stiamo usando il lemma 4.2).

Inoltre, per alleggerire la notazione, evitiamo di indicare le parentesi { e } intorno agli elementi di $E(n)$.

Esempio 4.11. Costruiamo un tableau per $\neg(p \to \neg q \vee r) \wedge (\neg p \vee (q \to \neg r))$. In ogni nodo sottolineiamo la formula su cui agiamo in quel nodo.



Notate che l'interpretazione definita da $v(p) = \mathbf{V}$, $v(q) = \mathbf{V}$, $v(r) = \mathbf{F}$ soddisfa sia i letterali che etichettano l'unica foglia marcata con $\bigcirc$ che la formula da cui siamo partiti.

Esercizio 4.12. Costruite tableaux per $(q \to \neg p) \wedge (r \to q) \wedge p \wedge r$ e per $\neg(p \wedge q \to \neg r) \vee (p \wedge \neg(q \wedge r))$. Controllate anche, con le tavole di verità o ragionando dalle definizioni, se queste due formule sono soddisfacibili e, se è il caso, trovate un'interpretazione che le soddisfi.

4. Terminazione forte dei tableaux

Il primo risultato che vogliamo dimostrare relativamente al metodo dei tableaux è la sua terminazione forte. La dimostrazione si basa sul seguente lemma, che è un caso particolare del lemma di König. Ricordiamo che un albero binario è un albero tale che ogni nodo ha al più due figli. Gli alberi costruiti dal metodo dei tableaux sono sempre binari.

Lemma 4.13. *Se un albero binario è infinito allora ha un ramo infinito.*

Dimostrazione. Sia T un albero binario e supponiamo che T sia infinito. Vogliamo dimostrare che T ha un ramo infinito. Diciamo che un nodo n di T è buono se il sottoalbero radicato in n è infinito.

Per ipotesi la radice di T è buona (il sottoalbero radicato in essa è T). Sia n un nodo buono: il sottoalbero radicato in n consiste di n e dell'unione dei sottoalberi radicati nei figli di n. Dato che n ha al più due figli, se entrambi questi ultimi fossero finiti (o addirittura n non avesse figli) il sottoalbero radicato in n sarebbe finito, contro la bontà di n. Perciò almeno uno dei figli di n deve essere buono.

Nel paragrafo precedente abbiamo dimostrato che la radice di T è buona e che ogni nodo buono ha un figlio buono. Usando questi due fatti è facile costruire un ramo in T che consiste di nodi buoni ed è infinito. □

Possiamo ora dimostrare il teorema che ci interessa.

TEOREMA 4.14. *L'algoritmo di costruzione dei tableaux gode della proprietà della terminazione forte, cioè termina qualunque siano il nodo e la formula su cui si decide di operare ad ogni singolo passo.*

DIMOSTRAZIONE. Se per assurdo la costruzione di un tableau non terminasse, essa darebbe origine ad un albero binario infinito. Per il lemma 4.13 un tale albero avrebbe un ramo infinito. Per raggiungere una contraddizione assegnamo ad ogni nodo n di T un numero naturale $W(n)$ in modo tale che se n' è un figlio di n si abbia $W(n') < W(n)$. Se T avesse un ramo infinito i valori di W lungo questo ramo sarebbero una successione infinita decrescente di numeri naturali, che è assurdo.

Per definire W useremo il rango rg (definizione 3.23): se n è un nodo di T, $W(n)$ è la somma dei ranghi delle formule in $E(n)$:

$$W(n) = \sum_{G \in E(n)} \mathrm{rg}(G).$$

Sia ora n' un figlio di n. Dobbiamo verificare che si ha sempre $W(n') < W(n)$, cioè $W(n') \leq W(n) - 1$.

Se n' è stato ottenuto agendo su una doppia negazione G con ridotto G_1 allora

$$E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$$

e quindi

$$W(n') = W(n) - \mathrm{rg}(G) + \mathrm{rg}(G_1) = W(n) - (\mathrm{rg}(G_1) + 1) + \mathrm{rg}(G_1) = W(n) - 1.$$

Se n' è stato ottenuto agendo su una α-formula $G \in E(n)$, con ridotti G_1 e G_2 allora

$$E(n') = (E(n) \setminus \{G\}) \cup \{G_1, G_2\}$$

e quindi

$$\begin{aligned} W(n') &= W(n) - \mathrm{rg}(G) + \mathrm{rg}(G_1) + \mathrm{rg}(G_2) = \\ &= W(n) - (\mathrm{rg}(G_1) + \mathrm{rg}(G_2) + 1) + \mathrm{rg}(G_1) + \mathrm{rg}(G_2) = W(n) - 1. \end{aligned}$$

Se invece n' è stato ottenuto agendo su una β-formula $G \in E(n)$, con ridotti G_1 e G_2 allora

$$\begin{aligned} E(n') &= (E(n) \setminus \{G\}) \cup \{G_1\} \quad \text{oppure} \\ E(n') &= (E(n) \setminus \{G\}) \cup \{G_2\}. \end{aligned}$$

Nel primo caso abbiamo

$$\begin{aligned} &W(n') = W(n) - \mathrm{rg}(G) + \mathrm{rg}(G_1) = \\ &= W(n) - (\mathrm{rg}(G_1) + \mathrm{rg}(G_2) + 1) + \mathrm{rg}(G_1) = W(n) - \mathrm{rg}(G_2) - 1 \leq W(n) - 1. \end{aligned}$$

Il secondo caso è del tutto analogo.

La dimostrazione del teorema è così completa. □

ESERCIZIO 4.15. Costruite un tableau per la formula $\neg(p \to q) \to (\neg p \vee q)$ e per ogni nodo n calcolate $W(n)$.

La dimostrazione del teorema 4.14 fornisce un limite superiore all'altezza di un tableau completo: se F è la formula che etichetta la radice l'altezza è minore o uguale a $\mathrm{rg}(F)$.

Esercizio 4.16. Sia F la formula dell'esempio 4.11. Calcolate $\mathrm{rg}(F)$ e confrontatelo con l'altezza del tableau per F.

Nota 4.17. Come notato per gli algoritmi di Fitting nella nota 3.25, anche l'algoritmo di costruzione dei tableaux **non** prevede la sostituzione di una formula con un'altra logicamente equivalente ad essa. Anche in questo caso, se introducessimo questa possibilità, l'algoritmo non godrebbe più della proprietà della terminazione forte e sarebbe quindi assai meno utile.

5. Correttezza e completezza

Dobbiamo ora dimostrare che il metodo dei tableaux è effettivamente una procedura di decisione per la soddisfacibilità delle formule proposizionali. Iniziamo con alcune definizioni che ci permettono di leggere l'output del nostro algoritmo.

Definizione 4.18. Un tableau è *chiuso* se tutte le sue foglie sono etichettate con insiemi di letterali che contengono una coppia complementare. Un tableau è *aperto* se non è chiuso, cioè se almeno una foglia è etichettata con un insieme di letterali che non contiene una coppia complementare.

Un *ramo aperto* di un tableau è un ramo che collega la radice dell'albero con una foglia etichettata con un insieme di letterali che non contiene una coppia complementare.

Il teorema principale che dimostreremo è il seguente.

Teorema 4.19. *Sia F una formula e T un tableau per F. F è insoddisfacibile se e solo se T è chiuso.*

Enunciamo immediatamente due importanti e utili conseguenze del teorema 4.19.

Teorema 4.20. *Sia F una formula e T un tableau per F. F è soddisfacibile se e solo se T è aperto.*

Dimostrazione. Immediata dal teorema 4.19. □

Teorema 4.21. *Sia F una formula e T un tableau per $\neg F$. F è valida se e solo se T è chiuso.*

Dimostrazione. F è valida se e solo se $\neg F$ è insoddisfacibile (teorema 2.35), se e solo se T è chiuso (teorema 4.19). □

I teoremi appena enunciati ci permettono anche di concludere che la chiusura o apertura del tableau dipende solo dalla formula iniziale, e non dalle scelte fatte durante la costruzione del tableau stesso.

Corollario 4.22. *Sia F una formula e T e T' due tableaux per F. T e T' sono entrambi chiusi o entrambi aperti.*

Dimostrazione. Se F è soddisfacibile T e T' sono entrambi aperti per il teorema 4.20. Se F è insoddisfacibile T e T' sono entrambi chiusi per il teorema 4.19. □

Dobbiamo ora dimostrare il teorema 4.19. È bene distinguere le due direzioni di questo teorema. Quella in avanti, cioè da sinistra a destra, è nota come teorema di completezza: ci dice che se F è insoddisfacibile il metodo dei tableaux riesce a scoprirlo, chiudendo tutti i rami di qualsiasi tableau per F. La direzione opposta è invece chiamata teorema di correttezza e asserisce che se il metodo dei tableaux dichiara una formula insoddisfacibile perché il tableau per essa è chiuso, la formula è veramente insoddisfacibile. La distinzione tra completezza e correttezza è utile

per tutte le procedure di decisione, anche in logiche molto più complesse di quella proposizionale. Usualmente è molto più facile dimostrare la correttezza di una procedura di decisione che provarne la completezza. Infatti è probabile che una procedura di decisione consista di passaggi chiaramente corretti, così che la correttezza sia facile da verificare. Più delicato è dimostrare che i passaggi ammessi sono sufficienti a garantire la completezza, cioè che non si è "dimenticato nulla". Il seguente esempio, per quanto paradossale, può essere utile a chiarire questo punto. Consideriamo la procedura di decisione che dichiara ogni formula soddisfacibile: essa è corretta perché non sbaglia mai nel dichiarare una formula insoddisfacibile (non lo fa mai!), ma è ovviamente ben lontana dall'essere completa.

Terremo dunque distinte le due direzioni del teorema 4.19, e le enunceremo e dimostreremo separatamente, iniziando dalla più semplice.

TEOREMA 4.23 (Teorema di correttezza). *Se T è un tableau chiuso per la formula F allora F è insoddisfacibile.*

DIMOSTRAZIONE. Dimostreremo un fatto più generale, che indicheremo con ✠:

> sia T un tableau e n un nodo di T tale che il sottoalbero di T che ha radice in n è chiuso; allora l'insieme di formule $E(n)$ è insoddisfacibile.

Il teorema è il caso particolare di ✠ ottenuto considerando T il tableau chiuso per F e n la radice di T (ricordate che in questo caso $E(n) = \{F\}$).

La dimostrazione di ✠ è per induzione sull'altezza del nodo n in T.

Se l'altezza di n in T è 0 vuol dire che n è una foglia del tableau. Il sottoalbero con radice in n (e che, in questo caso, non ha altri nodi oltre a n) è chiuso per ipotesi: ciò significa che $E(n)$ è un insieme di letterali che contiene una coppia complementare. Per il lemma 4.2 $E(n)$ è insoddisfacibile.

Consideriamo ora il caso in cui l'altezza di n in T è maggiore di 0. Allora n ha uno o due successori in T, che sono stati ottenuti agendo su qualche $G \in E(n)$ e ci sono tre possibilità.

(1) Se G è una doppia negazione con ridotto G_1, n ha un solo successore n' e si ha $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$. n' ha altezza minore di n in T e il sottoalbero di T che ha radice in n' è chiuso (perché lo è quello con radice in n). Per ipotesi induttiva (cioè applicando ✠ a n') $E(n')$ è insoddisfacibile. Sia ora v un'interpretazione qualsiasi: deve essere $v(H) = \mathbf{F}$ per qualche formula $H \in E(n')$ (che può dipendere da v).
   (i) se $H \in E(n) \setminus \{G\}$ allora $H \in E(n)$ e quindi v non soddisfa $E(n)$;
   (ii) se H è G_1 allora $v(H) = \mathbf{F}$ implica $v(G) = \mathbf{F}$ (ricordate che G è $\neg\neg G_1$ e $\neg\neg G_1 \equiv G_1$) ed anche in questo caso v non soddisfa $E(n)$.

   Dato che un'interpretazione arbitraria non soddisfa $E(n)$ concludiamo che $E(n)$ è insoddisfacibile.

(2) se G è una α-formula con ridotti G_1 e G_2, n ha un solo successore n' e si ha $E(n') = (E(n) \setminus \{G\}) \cup \{G_1, G_2\}$. Come nel caso precedente deduciamo che $E(n')$ è insoddisfacibile: fissato v si ha $v(H) = \mathbf{F}$ per qualche $H \in E(n')$.
   (i) se $H \in E(n) \setminus \{G\}$ allora ragioniamo come sopra;
   (ii) se H è G_1 oppure G_2 allora da $v(H) = \mathbf{F}$ segue $v(G_1 \wedge G_2) = \mathbf{F}$ e quindi $v(G) = \mathbf{F}$ perché $G \equiv G_1 \wedge G_2$ per il lemma 3.7. Dunque v non soddisfa $E(n)$.

   Abbiamo dunque dimostrato che $E(n)$ è insoddisfacibile.

(3) se G è una β-formula con ridotti G_1 e G_2, n ha due successori n_1 e n_2 e si ha $E(n_i) = (E(n) \setminus \{G\}) \cup \{G_i\}$ (per $i = 1$ e $i = 2$). Il ragionamento del punto (1) può venir ripetuto per ognuno dei due successori ottenendo che sia $E(n_1)$ che $E(n_2)$ sono insoddisfacibili. Fissiamo nuovamente un'interpretazione v:

(i) se $v(H) = \mathbf{F}$ per qualche $H \in E(n) \setminus \{G\}$ allora v non soddisfa $E(n)$ per le ragioni considerate in precedenza;
(ii) se per ogni $H \in E(n) \setminus \{G\}$ si ha $v(H) = \mathbf{V}$ allora, dato che v non soddisfa $E(n_1)$, deve essere $v(G_1) = \mathbf{F}$. Similmente, dato che v non soddisfa $E(n_2)$, deve essere $v(G_2) = \mathbf{F}$. Allora $v(G_1 \vee G_2) = \mathbf{F}$ e, dato che $G \equiv G_1 \vee G_2$ per il lemma 3.7, si ha $v(G) = \mathbf{F}$. Dunque v non soddisfa $E(n)$.

Perciò $E(n)$ è insoddisfacibile.

Abbiamo dunque dimostrato ✠ e quindi il teorema. □

TEOREMA 4.24 (Teorema di completezza). *Se la formula F è insoddisfacibile allora ogni tableau per F è chiuso.*

SCHEMA DELLA DIMOSTRAZIONE. Dimostreremo che se qualche tableau per F è aperto allora F è soddisfacibile. Sia dunque T un tableau aperto per F e fissiamo un ramo aperto r di T. La dimostrazione si svilupperà in tre passi:

a. definiremo quando un insieme di formule è un insieme di Hintikka (definizione 4.25);
b. dimostreremo che ogni insieme di Hintikka è soddisfacibile (lemma 4.29);
c. dimostreremo che $\bigcup_{n \in r} E(n)$ è un insieme di Hintikka (lemma 4.30).

Dato che $F \in \bigcup_{n \in r} E(n)$ (perché la radice del tableau appartiene a r) questi passi sono sufficienti a completare la dimostrazione. □

DEFINIZIONE 4.25. Un insieme di formule Γ è un *insieme di Hintikka* se soddisfa le seguenti quattro condizioni:

(1) Γ non contiene una coppia complementare di letterali;
(2) se una doppia negazione appartiene a Γ allora il suo ridotto appartiene a Γ;
(3) se una α-formula appartiene a Γ allora entrambi i suoi ridotti appartengono a Γ;
(4) se una β-formula appartiene a Γ allora almeno uno dei suoi ridotti appartiene a Γ.

ESEMPIO 4.26. L'insieme

$$\{p \vee \neg r \to q, \neg(p \vee \neg r), \neg p, \neg\neg r, r\}$$

è un insieme di Hintikka. Infatti non contiene coppie complementari di letterali, contiene uno dei ridotti della β-formula $p \vee \neg r \to q$, entrambi i ridotti della α-formula $\neg(p \vee \neg r)$ e il ridotto della doppia negazione $\neg\neg r$.

ESERCIZIO 4.27. Siano F_1 e F_2 le formule $\neg(p \to q \vee \neg r)$ e $\neg p \vee (r \to q)$.

(i) Definite insiemi di Hintikka Γ_1 e Γ_2 con $F_i \in \Gamma_i$.
(ii) $\Gamma_1 \cup \Gamma_2$ è un insieme di Hintikka?
(iii) ($\star$) Esiste un insieme di Hintikka che contiene sia F_1 che F_2?

ESERCIZIO 4.28. Verificare che l'insieme di formule che compaiono nelle etichette del ramo aperto del tableau dell'esempio 4.11 è un insieme di Hintikka.

LEMMA 4.29 (Lemma di Hintikka). *Ogni insieme di Hintikka è soddisfacibile.*

DIMOSTRAZIONE. Sia Γ un insieme di Hintikka. Definiamo un'interpretazione v come segue:

$$v(p) = \begin{cases} \mathbf{V} & \text{se } p \in \Gamma; \\ \mathbf{F} & \text{se } p \notin \Gamma. \end{cases}$$

Dimostreremo che per ogni $F \in \Gamma$ si ha $v(F) = \mathbf{V}$: questo implica che Γ è soddisfacibile. La dimostrazione è per induzione sul rango di F (definizione 3.23). Se $\mathrm{rg}(F) = 1$ allora F è un letterale:

- se $F \in \Gamma$ è una lettera proposizionale allora $v(F) = \mathbf{V}$ per definizione di v.
- se $F \in \Gamma$ è la negazione di una lettera proposizionale p allora $p \notin \Gamma$ per (1) nella definizione di insieme di Hintikka. Quindi $v(p) = \mathbf{F}$ e perciò $v(F) = \mathbf{V}$.

Se $\mathrm{rg}(F) > 1$ allora F è una doppia negazione, una α-formula o una β-formula:

- Se $F \in \Gamma$ è una doppia negazione con ridotto G allora $G \in \Gamma$ per (2) nella definizione di insieme di Hintikka. Poiché $\mathrm{rg}(G) = \mathrm{rg}(F) - 1$ l'ipotesi induttiva ci dice che $v(G) = \mathbf{V}$. Dato che $F \equiv G$ abbiamo $v(F) = \mathbf{V}$.
- Se $F \in \Gamma$ è una α-formula con ridotti G e H allora $G, H \in \Gamma$ per (3) nella definizione di insieme di Hintikka. Per ipotesi induttiva, dato che $\mathrm{rg}(G) < \mathrm{rg}(F)$ e $\mathrm{rg}(H) < \mathrm{rg}(F)$, possiamo assumere $v(G) = \mathbf{V}$ e $v(H) = \mathbf{V}$. Poiché $F \equiv G \wedge H$ abbiamo $v(F) = \mathbf{V}$.
- Se $F \in \Gamma$ è una β-formula con ridotti G e H allora $G \in \Gamma$ oppure $H \in \Gamma$ per (4) nella definizione di insieme di Hintikka. Se $G \in \Gamma$ per ipotesi induttiva, dato che $\mathrm{rg}(G) < \mathrm{rg}(F)$, $v(G) = \mathbf{V}$ e, dato che $F \equiv G \vee H$, abbiamo $v(F) = \mathbf{V}$. Il caso in cui $H \in \Gamma$ è del tutto analogo. □

LEMMA 4.30. *Se r è un ramo aperto di un tableau allora $\Gamma = \bigcup_{n \in r} E(n)$ è un insieme di Hintikka.*

DIMOSTRAZIONE. Sia f la foglia di r. Dobbiamo verificare che Γ soddisfa le quattro proprietà della definizione di insieme di Hintikka.

(1) Per verificare che Γ non contiene una coppia complementare di letterali notiamo che l'algoritmo dei tableaux non elimina mai i letterali dalle etichette dei nodi. Perciò se un letterale G appartene a qualche $E(n)$ con $n \in r$ allora $G \in E(f)$. Questo significa che se Γ contiene due letterali complementari G e H si ha $G, H \in E(f)$ e r non è aperto, contro la nostra ipotesi.
(2) Se una doppia negazione G con ridotto G_1 appartiene a Γ allora sia n il nodo di r più vicino a f tra quelli tali che $G \in E(n)$. Non può essere $n = f$, perché $E(f)$ è un insieme di letterali, e quindi n ha un successore $n' \in r$. Dato che $G \notin E(n')$ deve essere che G è la formula su cui si agisce in n. Perciò $G_1 \in E(n')$ e quindi $G_1 \in \Gamma$ come volevamo.
(3) Se una α-formula G con ridotti G_1 e G_2 appartiene a Γ allora si può ripetere il ragionamento svolto nel caso della doppia negazione per arrivare a concludere che $G_1, G_2 \in \Gamma$.
(4) se una β-formula G con ridotti G_1 e G_2 appartiene a Γ allora come sopra sia $n \neq f$ il nodo di r più vicino a f tra quelli tali che $G \in E(n)$. Anche in questo caso deve essere che G è la formula su cui si agisce in n e quindi n ha due successori. Esattamente uno di questi successori appartiene a r: indichiamolo con n'. Si ha $G_1 \in E(n')$ oppure $G_2 \in E(n')$ e quindi $G_1 \in \Gamma$ oppure $G_2 \in \Gamma$, come volevamo. □

La dimostrazione del teorema di completezza fornisce informazioni su come trovare un'interpretazione che soddisfi una formula il cui tableau è aperto.

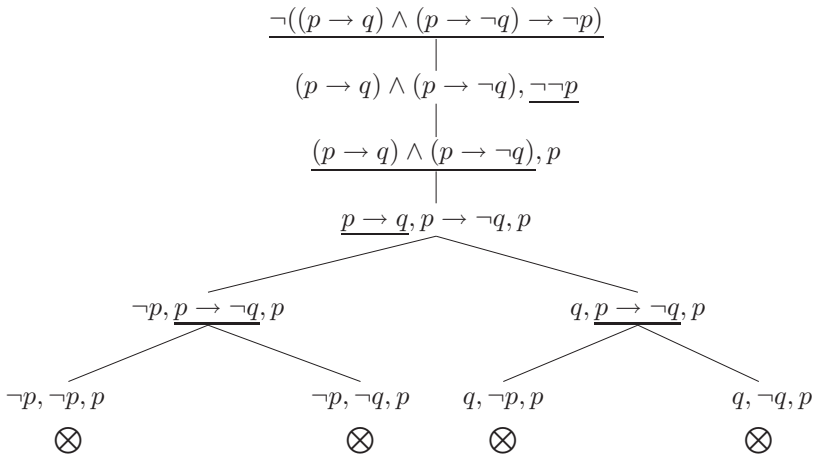
LEMMA 4.31. *Se una foglia f di un tableau per F non contiene coppie complementari allora l'interpretazione v definita da*

$$v(p) = \begin{cases} \mathbf{V} & \text{se } p \in E(f); \\ \mathbf{F} & \text{se } p \notin E(f). \end{cases}$$

soddisfa F.

Un esempio di applicazione di questo lemma è contenuto nell'esempio 4.11.

ESEMPIO 4.32. Utilizziamo il metodo dei tableaux per dimostrare la validità della formula $(p \to q) \wedge (p \to \neg q) \to \neg p$. Per il teorema 4.21 dobbiamo verificare che un tableau per la negazione di questa formula è chiuso. In ogni nodo sottolineiamo la formula su cui agiamo in quel nodo.



6. Semplificare i tableaux

Per scrivere più concisamente i tableaux è utile introdurre alcune abbreviazioni e piccole varianti dell'algoritmo 4.8.

CONVENZIONE 4.33. Da questo punto in poi ogniqualvolta nelle etichette di un nodo di un tableau compare una doppia negazione F con ridotto G scriveremo direttamente G, utilizzando la regola della doppia negazione immediatamente e contraendo due nodi in uno.

Nel tableau dell'esempio 4.32 la precedente convenzione avrebbe permesso di scrivere un nodo in meno (il secondo dalla radice, in cui si è agito su $\neg\neg p$).

CONVENZIONE 4.34. Da questo punto in poi ogniqualvolta nelle etichette di un nodo di un tableau compaiono una formula F con la sua negazione $\neg F$ consideriamo il nodo in questione chiuso e non operiamo più su di esso (secondo l'algoritmo 4.8 dovremmo ancora operare su di esso se l'etichetta contiene formule, compresa eventualmente F, che non sono letterali).

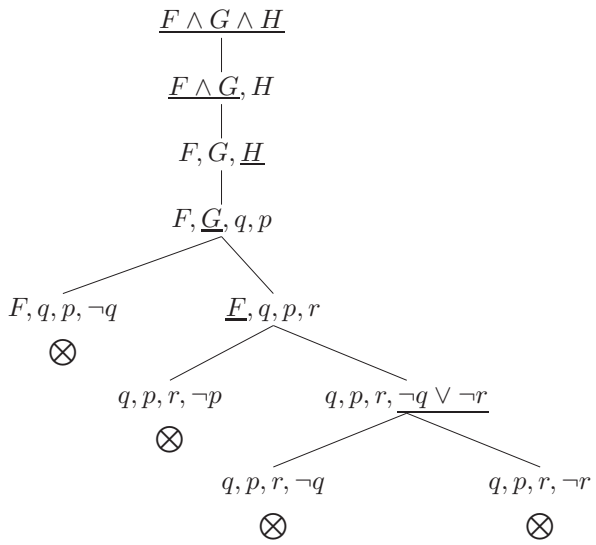
Nel tableau dell'esempio 4.32 la precedente convenzione avrebbe permesso di non scrivere le due foglie più a sinistra, chiudendo il ramo di sinistra dopo il nodo etichettato con $\neg p, p \to \neg q, p$.

NOTA 4.35. La convenzione 4.33 è solo formale (si tratta in pratica di un semplice trucco per scrivere un po' più rapidamente i tableaux). La convenzione 4.34 riguarda invece un mutamento sostanziale ed in realtà sarebbe necessario dimostrare nuovamente il teorema di correttezza per tableaux ottenuti in questo modo (non è necessario dimostrare nuovamente la completezza perché in quel caso la dimostrazione partiva dall'esistenza di rami aperti, mentre la convenzione indica un modo per chiudere rami). Omettiamo questa dimostrazione, che è una semplice modifica della dimostrazione precedente, basata sulla seconda parte della nota 4.3.

ESEMPIO 4.36. Utilizziamo le convenzioni 4.33 e 4.34 per costruire un tableau per la formula

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

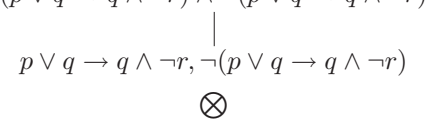
Per semplificare la notazione indichiamo con F, G e H le tre formule $p \to \neg q \vee \neg r$, $q \to r$ e $\neg(q \to \neg p)$. In ogni nodo sottolineiamo la formula su cui agiamo in quel nodo.



La convenzione 4.33 è stata applicata nel nodo ottenuto agendo su H (in cui sarebbe dovuto comparire $\neg\neg p$), la convenzione 4.34 nel nodo chiuso più in alto, in cui compare la formula F che non è un letterale.

Dato che tutti i rami sono chiusi il teorema di correttezza ci permette di concludere che la formula di partenza è insoddisfacibile.

ESEMPIO 4.37. Nel seguente tableaux la semplificazione apportata dalla convenzione 4.34 è particolarmente evidente:



ESERCIZIO 4.38. Sviluppate completamente un tableau per la formula

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

dell'esempio precedente utilizzando l'algoritmo 4.8 senza usare la convenzione 4.34.

Come già notato, l'algoritmo 4.8 è non deterministico: la seguente osservazione ci indica come cercare di ottenere il tableau più semplice possibile.

NOTA 4.39. Per semplificare il più possibile il tableau costruito dall'algoritmo 4.8 è opportuno agire su doppie negazioni o α-formule ogniqualvolta ciò sia possibile. In questo modo si dilaziona il più possibile la creazione di biforcazioni e l'incremento del numero delle foglie.

ESERCIZIO 4.40. Studiate con il metodo del tableaux le formule degli esempi 2.36, 2.37, 2.48 e 2.49 e dell'esercizio 2.53 (verificate che il risultato ottenuto con i tableaux coincida con quello ottenuto in precedenza).

7. I tableaux e la conseguenza logica

Abbiamo presentato il metodo dei tableaux come un metodo per studiare la validità o la soddisfacibilità di una singola formula. Usando il lemma 2.43 possiamo usare i tableaux per studiare la validità o la soddisfacibilità di un insieme finito di formule $\{F_1, \ldots, F_n\}$. Nel caso della soddisfacibilità si dovrebbe costruire un tableau per $F_1 \wedge \cdots \wedge F_n$. I primi passi del tableau conducono ad un nodo etichettato $F_1, \ldots, F_n$: questa semplice osservazione giustifica il seguente algoritmo.

ALGORITMO 4.41. Per stabilire se un insieme finito $\Gamma = \{F_1, \ldots, F_n\}$ di formule è soddisfacibile costruiamo un tableau la cui radice è etichettata con Γ. Se il tableau è aperto Γ è soddisfacibile (e l'etichetta priva di coppie complementari di una foglia ci permette di definire un'interpretazione che soddisfa Γ), se il tableau è chiuso Γ è insoddisfacibile.

Il lemma 2.38 ci permette di usare i tableaux per stabilire la sussistenza della relazione di conseguenza logica. In questo caso per stabilire che $F \models G$ bisogna costruire un tableau chiuso per $\neg(F \to G)$ (la negazione delle formula di cui si vuole stabilire la validità), che ha il primo nodo sotto la radice etichettato da $F, \neg G$. Generalizzando questa osservazione al caso in cui a sinistra del simbolo di conseguenza logica compare un insieme finito di formule (definizione 2.26) si ottiene il seguente algoritmo.

ALGORITMO 4.42. Per stabilire se $F_1, \ldots, F_n \models G$ costruiamo un tableau la cui radice è etichettata con $\{F_1, \ldots, F_n, \neg G\}$. Se il tableau è chiuso $F_1, \ldots, F_n \models G$, se il tableau è aperto $F_1, \ldots, F_n \not\models G$ (e l'etichetta priva di coppie complementari di una foglia ci permette di definire un'interpretazione che soddisfa $F_1, \ldots, F_n$ ma non G).

ESERCIZIO 4.43. Studiate con il metodo del tableaux le conseguenze logiche degli esempi 2.17, 2.50, 2.51 e 2.59 e degli esercizi 2.25, 2.52 e 2.54 (verificate che il risultato ottenuto con i tableaux coincida con quello ottenuto in precedenza). Per i problemi che riguardano un'equivalenza logica bisogna studiare le due conseguenze logiche, usando il lemma 2.19.

CAPITOLO 5

Sintassi della logica predicativa

Le formule predicative sono asserzioni riguardo alle relazioni che intercorrono tra certi oggetti. Esse si differenziano dalle formule proposizionali proprio per la possibilità di riferirsi ad oggetti.

Per poter parlare di oggetti useremo espressioni che non sono formule: i termini. A differenza delle formule, anche dopo lo sviluppo della semantica i termini non saranno né veri né falsi. Anche in questo caso ci può aiutare considerare la situazione dell'aritmetica: $(1 + 1) \cdot 1$, che è un termine in un opportuno linguaggio (quello dell'esempio 5.3 qui sotto), designa un elemento (il numero 2, se tutti i simboli sono interpretati in modo naturale), e non può essere né vero né falso. $0 < (1 + 1) \cdot 1$ è invece una formula del linguaggio dell'esempio 5.3 e risulta essere vera se tutti i simboli sono interpretati in modo naturale. Formule più complicate sono $\forall x\, \exists y\, x < y$ e $\exists y\, \forall x\, x < y$, che sono rispettivamente vera e falsa (sempre se tutti i simboli sono interpretati in modo naturale nell'ambito dell'insieme dei numeri naturali $\mathbb{N}$). Per stabilire se la formula $1 < x$ sia vera o falsa è necessario specificare, oltre all'interpretazione di 1 e di $<$, anche il significato che si attribuisce a x. Notate che x era presente anche in alcune formule precedenti, ma in quei casi non era necessario interpretarlo per capire se la formula fosse vera o falsa: questa distinzione viene resa precisa dalle nozioni di variabile libera e legata (definizione 5.34).

Come nel caso delle formule proposizionali, anche la costruzione dei termini e delle formule predicative avviene per ricorsione (definizioni 5.8 e 5.17). Prima di poter dare queste definizioni dobbiamo fissare gli ingredienti fondamentali che usiamo per costruire termini e formule.

1. Linguaggi predicativi

DEFINIZIONE 5.1. Un linguaggio predicativo contiene i seguenti elementi comuni:

- un insieme infinito numerabile di *variabili*, che indicheremo con $x, y, z, \ldots$;
- i *simboli logici*: i *connettivi* già visti nel caso proposizionale $\neg$, $\wedge$, $\vee$ e $\rightarrow$ e i *quantificatori* $\forall$ e $\exists$
- la virgola "," e le parentesi "(" e ")".

Questi elementi saranno sempre a nostra disposizione. Oltre a questi avremo a disposizione altri simboli, che scegliamo di volta in volta a seconda dell'argomento su cui intendiamo fare delle affermazioni. Questi elementi variabili (ma le variabili non sono tra essi!) identificano un linguaggio.

DEFINIZIONE 5.2. Un *linguaggio predicativo* consiste dei seguenti insiemi (che è sempre conveniente siano disgiunti, per evitare di incorrere in confusioni):

- un insieme di *simboli di costante*;
- un insieme di *simboli di funzione*, ciascuno fornito della propria *arietà*, che è un numero naturale $n \geq 1$;
- un insieme non vuoto di *simboli di relazione* (o *simboli predicativi*), ciascuno fornito della propria *arietà*, che è un numero naturale $n \geq 1$.

Intuitivamente la arietà di un simbolo di funzione indica a quanti elementi esso si può applicare. Simile è il significato della arietà di un simbolo di relazione. Se un simbolo di funzione o di relazione ha arietà n diciamo anche che è n-ario.

Notiamo che gli insiemi dei simboli di costante e di funzione possono essere vuoti, mentre ogni linguaggio contiene sempre almeno un simbolo di relazione. Questa è l'unica restrizione che poniamo sui tre insiemi: ognuno di essi può essere finito o infinito, quelli dei simboli di funzione e relazione possono contenere solo simboli di arietà 1 (detti usualmente unari), oppure qualche simbolo di arietà 2 (binario) e qualcuno di arietà 3 (ternario), oppure ancora simboli di infinite arietà diverse.

ESEMPIO 5.3. Il linguaggio $\mathcal{L}_{\text{arit}}$, adatto a fare affermazioni sull'aritmetica, consiste dei simboli di costante 0 e 1, dei simboli di funzione binari $+$ e $\cdot$, dei simboli di relazione binari $<$, $>$ e $=$.

ESEMPIO 5.4. Il linguaggio $\mathcal{L}_{\mathbb{R}}$, adatto a fare affermazioni sui numeri reali, consiste dei simboli di costante 0, 1, e e π, dei simboli di funzione unari di elevazione all'n-esima potenza ^{n} (uno per ogni $n \in \mathbb{N}$, così che abbiamo infiniti simboli di funzione), sin e cos, dei simboli di funzione binari $+$ e $\cdot$, dei simboli di relazione binari $<$, $>$ e $=$.

ESEMPIO 5.5. Il linguaggio $\mathcal{L}_{\text{seq}}$, adatto a fare affermazioni sulle successioni di numeri naturali, consiste dei simboli di costante $\langle\rangle$ e $\langle n\rangle$ (uno per ogni n, così che abbiamo infiniti simboli di costante), del simbolo di funzione binario $^\frown$, dei simboli di relazione binari $\subset$ e $=$. [$\langle\rangle$ rappresenta la successione vuota, $\langle n\rangle$ quella il cui unico elemento è n, $^\frown$ la concatenazione di due successioni, $\subset$ l'essere segmento iniziale.]

ESEMPIO 5.6. Il linguaggio $\mathcal{L}_{\text{set}}$, adatto a fare affermazioni sugli insiemi, consiste del simbolo di costante $\emptyset$, dei simboli di funzione unari $\{\}$, $\bigcup$ e $\mathcal{P}$, dei simboli di funzione binari $\cup$, $\cap$ e $\setminus$, dei simboli di relazione binari $\in$, $\subseteq$ e $=$.

ESEMPIO 5.7. Il linguaggio $\mathcal{L}_{\text{fam}}$, adatto a fare affermazioni sulle relazioni di parentela, consiste di un simbolo di costante per ogni membro della famiglia, dei simboli di funzione unari p ("il padre di"), e m ("la madre di"), dei simboli di relazione binari f_1 ("sono fratelli") e f_2 ("è figlio di").

2. Termini

I termini di un linguaggio fissato sono costruiti utilizzando solo alcuni dei suoi elementi: le variabili e i simboli di costante e di funzione, ed inoltre la virgola e le parentesi.

DEFINIZIONE 5.8. Sia $\mathcal{L}$ un linguaggio fissato. L'insieme dei *termini di* $\mathcal{L}$ è definito per ricorsione come segue:

- ogni variabile è un termine di $\mathcal{L}$;
- ogni simbolo di costante di $\mathcal{L}$ è un termine di $\mathcal{L}$;
- se $t_1, \dots t_n$ sono termini di $\mathcal{L}$ e f è un simbolo di funzione n-ario di $\mathcal{L}$, allora $f(t_1, \dots, t_n)$ è un termine di $\mathcal{L}$.

Un termine è *chiuso* se in esso non compare nessuna variabile (per definire i termini chiusi si può ripetere la definizione precedente omettendo la prima condizione).

Se un linguaggio non contiene né simboli di costante né simboli di funzione allora i suoi termini coincidono con le variabili. Se un linguaggio non contiene simboli di costante allora non esiste alcun termine chiuso.

ESEMPIO 5.9. Alcuni termini del linguaggio $\mathcal{L}_{\text{arit}}$ sono i seguenti: x, 1, $+(x,0)$ (il secondo è chiuso, gli altri no). Per quest'ultimo è più comune scrivere $x+0$, e convenzioni analoghe si usano anche per $\cdot$, così che $(x\cdot y)+(0\cdot(y+x))$ è anch'esso un termine (non chiuso), che, con l'usuale convenzione cha dà precedenza al prodotto rispetto alla somma, può venir scritto come $x\cdot y+0\cdot(y+x)$.

ESEMPIO 5.10. Alcuni termini del linguaggio $\mathcal{L}_{\mathbb{R}}$ sono e, $\cos(\pi)$, $\sin(\cos(x+e))$, $\sin(x\cdot y^7)+\cos(1)^3$ (anche in questo caso usiamo le usuali convenzioni di scrittura delle funzioni utilizzate in analisi): i primi due sono chiusi.

ESEMPIO 5.11. Alcuni termini del linguaggio $\mathcal{L}_{\text{seq}}$ sono $\langle 3\rangle^\frown\langle\rangle$ e $(x^\frown\langle 6\rangle)^\frown z$: il primo è chiuso.

ESEMPIO 5.12. Alcuni termini del linguaggio $\mathcal{L}_{\text{fam}}$ sono $p(x)$ e $m(p(a))$, dove a è un simbolo di costante che corrisponde a qualche membro della famiglia: il secondo è chiuso.

ESEMPIO 5.13. Sia $\mathcal{L}_0$ un linguaggio privo di simboli di costante e con un unico simbolo di funzione f, che è unario. I termini di $\mathcal{L}_0$ sono tutte le stringhe di simboli del tipo v, $f(v)$, $f(f(v))$, $f(f(f(v)))$, $\dots$, dove v è una qualsiasi variabile (per ogni v ci sono infiniti di questi termini!). Se quando f compare k volte scriviamo $f^{(k)}(v)$ (e quindi in particolare $f^{(0)}(v)$ è v), abbiamo che tutti i termini di $\mathcal{L}_0$ sono della forma $f^{(k)}(v)$ con $k\in\mathbb{N}$ e v variabile.

Nel linguaggio $\mathcal{L}_0$ non ci sono termini chiusi.

ESEMPIO 5.14. Sia $\mathcal{L}_1$ il linguaggio ottenuto aggiungendo a $\mathcal{L}_0$ dell'esempio precedente un unico simbolo di costante c. I termini chiusi di $\mathcal{L}_1$ sono quelli della forma $f^{(k)}(c)$ con $k\in\mathbb{N}$.

La definizione di termine è ricorsiva e questo fa sì che spesso ragioneremo induttivamente sui termini: per dimostrare che una proprietà vale per tutti i termini è sufficiente dimostrare che essa vale per le variabili e i simboli di costante e che se f è un simbolo di funzione n-ario e la proprietà vale per i termini $t_1,\dots,t_n$, allora vale anche per $f(t_1,\dots,t_n)$. Similmente le definizioni di operazioni sui termini possono venir date per ricorsione: basta definire il risultato dell'operazione su variabili e simboli di costante e, supponendo di aver già definito il risultato dell'operazione su $t_1,\dots,t_n$, definirlo anche su $f(t_1,\dots,t_n)$ (si veda ad esempio la definizione 5.15).

Un'operazione fondamentale sui termini è quella di sostituzione. L'idea è che se t è un termine, x una variabile e s un termine la sostituzione di x con s in t (indicata da $t\{x/s\}$), è il termine ottenuto rimpiazzando ogni occorrenza di x in t con s. La sostituzione è definita per ricorsione sulla complessità del termine t.

DEFINIZIONE 5.15. Se x è una variabile e s e t sono termini definiamo la *sostituzione di x con t in s*, $s\{x/t\}$, per ricorsione su s:

- se s è la variabile x allora $s\{x/t\}$ è t;
- se s è una variabile diversa da x oppure un simbolo di costante allora $s\{x/t\}$ è s;
- se s è $f(s_1,\dots,s_n)$ allora $s\{x/t\}$ è $f(s_1\{x/t\},\dots,s_n\{x/t\})$.

ESEMPIO 5.16. Consideriamo un linguaggio contenente i simboli di costante a e b e i simboli di funzione f e g, il primo binario e il secondo unario.

$f(g(x),a)\{x/b\}$ è $f(g(b),a)$	dove s è $f(g(x),a)$ e t è b;
$f(x,g(x))\{x/g(a)\}$ è $f(g(a),g(g(a)))$	dove s è $f(x,g(x))$ e t è $g(a)$;
$f(x,g(x))\{x/g(x)\}$ è $f(g(x),g(g(x)))$	dove s è $f(x,g(x))$ e t è $g(x)$;
$g(y)\{x/a\}$ è $g(y)$	dove s è $g(y)$ e t è a;
$x\{x/a\}$ è a	dove s è x e t è a.

3. Formule predicative

Le formule di un linguaggio predicativo fissato sono costruite a partire dai termini per mezzo dei simboli di relazione e dei simboli logici:

DEFINIZIONE 5.17. Sia $\mathcal{L}$ un linguaggio predicativo. Le *formule atomiche di* $\mathcal{L}$ sono le stringhe di simboli del tipo $p(t_1, \ldots, t_n)$ dove p è un simbolo di relazione n-ario e $t_1, \ldots, t_n$ sono termini.

L'insieme delle *formule di* $\mathcal{L}$ è definito per ricorsione come segue:

- ogni formula atomica di $\mathcal{L}$ è una formula di $\mathcal{L}$;
- se F è una formula di $\mathcal{L}$ allora $(\neg F)$ è una formula di $\mathcal{L}$;
- se F e G sono formule di $\mathcal{L}$ allora $(F \wedge G)$, $(F \vee G)$ e $(F \to G)$ sono formule di $\mathcal{L}$;
- se F è una formula di $\mathcal{L}$ e x è una variabile, allora $(\forall x\, F)$ e $(\exists x\, F)$ sono formule di $\mathcal{L}$ (che vengono dette rispettivamente la *quantificazione universale* e la *quantificazione esistenziale di* F *rispetto a* x).

Le formule $(\neg F)$, $(F \wedge G)$, $(F \vee G)$ e $(F \to G)$ vengono lette come nel caso proposizionale. La formula $(\forall x\, F)$ viene letta "per ogni x, F" e la formula $(\exists x\, F)$ viene letta "esiste x tale che F".

NOTA 5.18. Notate che se un linguaggio non avesse simboli di relazione non avrebbe formule atomiche e di conseguenza non avrebbe formule: questa è la ragione per cui nella definizione 5.2 abbiamo richiesto che l'insieme dei simboli di relazione sia non vuoto.

ESEMPIO 5.19. Sia $\mathcal{L}_0$ un linguaggio privo di simboli di costante, con un simbolo di funzione unario f (si veda l'esempio 5.13), con un simbolo di relazione unario p ed un simbolo di relazione binario r. Le formule atomiche di $\mathcal{L}_0$ sono della forma $p(f^{(k)}(v))$ e $r(f^{(k)}(v), f^{(h)}(u))$ con $k, h \in \mathbb{N}$ e v e u variabili. In particolare tra di esse ci sono $p(x)$, $r(f(y), x)$ e $p(f(f(x)))$. Notate che $p(p(x))$ **non** è una formula!

Alcune formule di $\mathcal{L}_0$ sono $(\neg p(x))$, $(p(x) \wedge (r(f(y), x) \vee p(f(f(x)))))$, $(\forall x\, p(x))$, $(\exists y\, r(x, z))$ e $(((\neg p(x)) \to (\forall x\, p(x))) \to (((\neg p(x)) \wedge (r(f(y), x) \vee p(z))) \wedge p(x)))$.

Per indicare formule e insiemi di formule continueremo ad utilizzare la convenzione 1.6.

Il seguente lemma è l'analogo del lemma 1.7.

LEMMA 5.20. *Ogni formula è di uno e uno solo dei seguenti sette tipi:*

- *una formula atomica;*
- *una* negazione*;*
- *una* congiunzione*;*
- *una* disgiunzione*;*
- *un'*implicazione*;*
- *una* quantificazione universale*, cioè una formula del tipo* $(\forall x\, F)$ *per una variabile* x *e una formula* F*;*
- *una* quantificazione esistenziale*, cioè una formula del tipo* $(\exists x\, F)$ *per una variabile* x *e una formula* F*.*

ESEMPIO 5.21. Le formule del secondo paragrafo dell'esempio 5.19 sono rispettivamente una negazione, una congiunzione, una quantificazione universale, una quantificazione esistenziale e un'implicazione.

L'ultima formula dell'esempio 5.19, per quanto formalmente corretta, è probabilmente già oltre il limite della leggibilità umana. Per migliorare la leggibilità delle formule adotteremo alcune convenzioni, che estendono quelle adottate nel caso proposizionale (convenzione 1.13).

CONVENZIONE 5.22. Nella scrittura delle formule adotteremo le seguenti *convenzioni*:

- si omettono le parentesi più esterne;
- $\neg$, $\forall$ e $\exists$ hanno la precedenza su $\wedge$, $\vee$ e $\to$, così che $\forall x\, F \to G$ abbrevia $((\forall x\, F) \to G)$;
- $\wedge$ e $\vee$ hanno la precedenza su $\to$;
- ulteriori parentesi eventualmente omesse in formule costruite con più di una $\wedge$ o $\vee$ si appoggiano a sinistra.

Con queste convenzioni l'ultima formula dell'esempio 5.19 diventa

$$(\neg p(x) \to \forall x\, p(x)) \to \neg p(x) \wedge (r(f(y), x) \vee p(z)) \wedge p(x).$$

ESERCIZIO 5.23. Stabilite qual è la differenza tra le formule

$$\forall x\, p(x) \to \exists y\, r(x, y) \wedge \neg\exists u\, r(f(u), z)$$
$$\forall x\, p(x) \to \exists y\, (r(x, y) \wedge \neg\exists u\, r(f(u), z))$$
$$\forall x\, (p(x) \to \exists y\, r(x, y) \wedge \neg\exists u\, r(f(u), z))$$

analizzando i passaggi attraverso cui sono state costruite a partire da formule atomiche.

La definizione di formula, come quella di termine, è induttiva e questo ci permette di ragionare induttivamente e definire operazioni ricorsivamente sulle formule. Come primo esempio di quest'ultimo procedimento consideriamo la seguente definizione, che estende quella analoga nel caso proposizionale (definizione 1.10):

DEFINIZIONE 5.24. Il *grado della formula* F, indicato con $g(F)$, è definito da:

- $g(F) = 0$ se F è atomica;
- $g(\neg F) = g(\forall x\, F) = g(\exists x\, F) = g(F) + 1$;
- $g(F \wedge G) = g(F \vee G) = g(F \to G) = g(F) + g(G) + 1$.

ESERCIZIO 5.25. Calcolare il grado delle formule dell'esempio 5.19.

ESERCIZIO 5.26. ($\star$) Dimostrare per induzione sulla complessità delle formule che il grado di F è il numero di connettivi e quantificatori che compaiono in F.

DEFINIZIONE 5.27. Se Γ è un insieme di formule il *linguaggio di* Γ è indicato con $\mathcal{L}(\Gamma)$ e consiste dei simboli di costante, funzione e relazione che compaiono in qualche elemento di Γ. Se $\Gamma = \{F\}$ scriveremo $\mathcal{L}(F)$, e similmente $\mathcal{L}(F, G)$ sta per $\mathcal{L}(\{F, G\})$.

ESEMPIO 5.28. Se F è

$$p(a) \wedge \forall x(p(f(x)) \to \neg r(x, b)) \to \exists y(q(a, y, g(y, b)) \vee q(y, g(y, a), f(b)))$$

$\mathcal{L}(F)$ consiste dei simboli di costante a e b, dei simboli di funzione f (unario) e g (binario), e dei simboli di relazione p (unario), r (binario) e q (ternario).

ESERCIZIO 5.29. Indicare quale tra le seguenti stringhe di simboli è una formula atomica del linguaggio dell'esempio 5.28:

$$q(a);\quad p(y);\quad p(g(b));\quad \neg r(x, a);\quad q(x, p(a), b);\quad p(g(f(a), g(x, f(x))));$$
$$q(f(a), f(f(x)), f(g(f(z), g(a, b))));\quad r(a, r(a, a));\quad r(a, g(a, a));\quad g(a, g(a, a)).$$

ESERCIZIO 5.30. Indicare quale tra le seguenti stringhe di simboli è una formula del linguaggio dell'esempio 5.28:

$$\forall x\, \neg p(x);\quad \forall x\, \neg p(y);\quad \neg r(p(a), x);\quad \exists a\, r(a, a);\quad \exists x\, q(x, f(x), b) \to \forall x\, r(a, x);$$
$$\exists x\, p(r(a, x));\quad \forall r(x, a);\quad \to p(b);\quad r(x, b)\neg\exists y\, q(y, y, y);\quad r(x, b) \vee \neg\exists y\, q(y, y, y);$$
$$\neg y\, p(y);\quad \neg\neg p(a);\quad \neg\neg\forall x\, \neg p(x);\quad \forall x\, \exists y(r(x, y) \to r(y, x)\neg r(g(y, x), f(x))).$$

ESERCIZIO 5.31. Indicare in quale tra le seguenti linguaggi la stringa $\exists x\, p(g(f(x)))$ è una formula:

(a) p è un simbolo di relazione unario e g e f sono simboli di funzione unari;
(b) p, g e f sono simboli di funzione unari;
(c) p e g sono simboli di relazione e f è un simbolo di funzione.

ESERCIZIO 5.32. Dire di che tipo devono essere i simboli p, q, f e g (se di funzione, di relazione e di quale arietà) affinché la stringa di simboli seguente sia una formula:

$$\forall x(p(g(a, f(x))) \vee q(g(x, y), f(g(a, b)))).$$

(Questo è equivalente a stabilire qual è il linguaggio della formula in questione.)

ESERCIZIO 5.33. Spiegare perché le stringhe di simboli

$$\forall x(p(x) \to p(x, a)), \quad \exists x(q(x) \wedge r(q(a), a)) \quad \text{e} \quad g(s(x) \to t(x))$$

non sono formule di nessun linguaggio.

4. Variabili libere e enunciati

Il ruolo della variabile x nelle formule $p(x)$ e $\forall x\, p(x)$ (ovviamente siamo in un linguaggio che contiene un simbolo di relazione unario p) è ben diverso. Infatti per decidere se la prima formula è vera o falsa è necessario dare un significato, oltre che a p, anche a x, mentre la verità o falsità della seconda formula dipende solo da come interpretiamo p. Per catturare questa differenza diamo la seguente definizione per ricorsione sulla complessità di una formula.

DEFINIZIONE 5.34. Sia F una formula e x una variabile. Definiamo le *occorrenze libere di x in F* come segue:

- se F è atomica allora ogni occorrenza di x in F è libera;
- se F è $\neg G$ allora le occorrenze libere di x in F sono le occorrenze libere di x in G;
- se F è $G \wedge H$, $G \vee H$ oppure $G \to H$, allora le occorrenze libere di x in F sono le occorrenze libere di x in G e le occorrenze libere di x in H;
- se F è $\forall x\, G$ oppure $\exists x\, G$, allora nessuna occorrenza di x in F è libera;
- se F è $\forall y\, G$ oppure $\exists y\, G$ dove y è una variabile diversa da x, allora le occorrenze libere di x in F sono le occorrenze libere di x in G.

Le occorrenze di x in F che non sono libere si dicono *occorrenze legate*. Le *variabili libere* di una formula F sono quelle che hanno almeno un'occorrenza libera in F. Una formula priva di variabili libere è chiamata *enunciato* o *formula chiusa*.

ESEMPIO 5.35. Nella formula

$$\forall x(r(f(x), \underline{y}) \to \exists y\, r(f(y), x)) \to \forall y(\neg r(y, \underline{x}) \vee r(\underline{z}, f(\underline{z})) \to \exists w\, \forall x\, r(x, w))$$

le occorrenze libere delle variabili sono sottolineate. Pertanto le variabili libere di questa formula sono y, x e z e la formula non è un enunciato. Notiamo che x e y hanno sia occorrenze libere che occorrenze legate in questa formula.

L'idea della definizione 5.34 è che le variabili libere di una formula vanno interpretate in un modo che è ancora da stabilire: la loro presenza impedisce quindi di stabilire se la formula in questione sia vera o falsa. Un enunciato invece esprime direttamente una proprietà di ciò di cui si sta parlando e pertanto, una volta stabilito come interpretare i simboli di costante, funzione e relazione, risulterà essere vero o falso.

ESEMPIO 5.36. Nel linguaggio dell'esempio 5.3 con i vari simboli interpretati in modo naturale, non ha senso chiedersi se la formula atomica con variabili libere $x+1=1$ sia vera o falsa. Invece gli enunciati $1+1=1$, $\forall x\, x+1=1$ e $\exists x\, x+1=1$ sono falsi i primi due e vero il terzo.

ESERCIZIO 5.37. Nel linguaggio dell'esempio 5.28 stabilire quali sono le variabili libere nelle seguenti formule e quali di esse sono enunciati:

$$p(a);\quad p(x)\land\neg r(y,a);\quad \exists x\, r(y,y);\quad \forall x\, p(x)\to\exists y\,\neg q(f(x),y,f(y));$$
$$\forall x\,\exists y\, r(x,f(y))\to r(x,y);\quad \forall x(\exists y\, r(x,f(y))\to r(x,y));$$
$$\neg r(f(a),a);\quad \forall z\,(p(z)\to\exists y(\exists x\, q(x,y,z)\lor q(z,y,x)));$$
$$\forall x(p(x)\to\exists y\,\neg q(f(x),y,f(y)));\quad \forall z\,\exists u\,\exists y(q(z,u,g(u,y))\lor r(u,g(z,u)));$$
$$\forall z\,\exists x\,\exists y(q(z,u,g(u,y))\lor r(u,g(z,u)));\quad \forall z(\exists y\, q(z,u,g(u,y))\lor\exists u\, r(u,g(z,u))).$$

Fate lo stesso con la formula dell'esempio 5.28 e quelle degli esercizi 5.29, 5.30, 5.31 e 5.32.

Le formule di certi tipi hanno una certa importanza e quindi hanno dei nomi specifici per designarle. Alcuni esempi li abbiamo già incontrati: le formule atomiche e gli enunciati.

DEFINIZIONE 5.38. Una *formula aperta* è una formula senza alcun quantificatore (cioè in cui non compaiono $\forall$ e $\exists$).

Se identifichiamo le formule atomiche della logica predicativa con le lettere proposizionali della logica proposizionale, le formule aperte diventano le formule proposizionali studiate nei primi capitoli di queste dispense.

5. Sottoformule

Questa sezione è quasi una ripetizione della sezione 3 del capitolo 1: le idee della definizione di sottoformula sono esattamente le stesse nel caso proposizionale e in quello predicativo.

DEFINIZIONE 5.39. Se F è una formula, diciamo che G è una *sottoformula* di F se G è una formula che è una sottostringa di F. G è una *sottoformula propria* di F se è diversa da F.

La definizione precedente va applicata tenendo a mente la definizione 5.17 di formula, anche quando si utilizza la convenzione 5.22.

ESEMPIO 5.40. Se F è

$$\forall x(p(x)\to\exists y\, p(y)\lor r(x,y)),$$

$\exists y\, p(y)\lor r(x,y)$ è una sottoformula di F, mentre $p(x)\to\exists y\, p(y)$ non lo è. Infatti inserendo alcune delle parentesi omesse in base alla convenzione 5.22 F è

$$\forall x(p(x)\to((\exists y\, p(y))\lor r(x,y))).$$

In effetti $\exists y\, p(y)\lor r(x,y)$ è una delle formule utilizzate nella costruzione di F, mentre $p(x)\to\exists y\, p(y)$ non lo è.

ESERCIZIO 5.41. Elencate tutte le sottoformule della F dell'esempio precedente (sono sette, di cui sei proprie).

Per dare una definizione precisa di sottoformula possiamo procedere per induzione sulla complessità delle formule.

DEFINIZIONE 5.42. Definiamo per ricorsione sulla complessità della formula F quali sono le *sottoformule* di F:

- se F è atomica, F è la sua unica sottoformula;
- se F è $\neg G$, $\forall x\, G$ oppure $\exists x\, G$ allora le sottoformule di F sono le sottoformule di G e F stessa;
- se F è $G \wedge H$, $G \vee H$ oppure $G \to H$ allora le sottoformule di F sono le sottoformule di G, le sottoformule di H e F stessa.

6. Sostituzioni in formule

Abbiamo visto come effettuare sostituzioni nei termini (definizione 5.15), e ora ci proponiamo di effettuare sostituzioni nelle formule. Nel caso delle formule atomiche il procedimento è semplice, e si basa proprio sulle sostituzioni in termini.

DEFINIZIONE 5.43. Se F è una formula atomica $p(s_1, \ldots, s_k)$, x è una variabile e t è un termine la *sostituzione di x con t in F* è $p(s_1\{x/t\}, \ldots, s_k\{x/t\})$ ed è denotata da $F\{x/t\}$.

ESEMPIO 5.44. $q(x, g(x, f(y)), f(g(z, y)))\{x/f(w)\}$ è

$$q(f(w), g(f(w), f(y)), f(g(z, y))).$$

Quando la formula in cui vogliamo effettuare la sostituzione non è atomica (ed in particolare quando non è aperta) in generale occorre un po' di cautela[1]. In questa sede il nostro interesse è ristretto al caso in cui vogliamo sostituire una variabile con un termine chiuso (ed in particolare con un simbolo di costante): in questo contesto la situazione è abbastanza semplice. Notiamo che se F è $p(x) \wedge \exists x\, q(x)$ e la sostituzione è $\{x/a\}$, dove a è un simbolo di costante, sostituendo sistematicamente x con a in tutta F otterremmo $p(a) \wedge \exists a\, q(a)$, che non è una formula. Il nostro obiettivo è in realtà asserire riguardo ad a ciò che la F asserisce riguardo a x, e quindi ottenere $p(a) \wedge \exists x\, q(x)$. A questo scopo basta stabilire che vanno sostituite solo le occorrenze libere delle variabili nel dominio della sostituzione.

DEFINIZIONE 5.45. Se F è una formula, x è una variabile e t è un termine chiuso la *sostituzione di x con t in F* è ottenuta sostituendo in F ogni formula atomica A in cui x occorre libero con $A\{x/t\}$ ed è denotata da $F\{x/t\}$.

ESEMPIO 5.46. Siano t un termine chiuso e F la formula

$$\exists y\, \forall x(p(x) \to r(x, y)) \to p(x).$$

Allora $F\{x/t\}$ è

$$\exists y\, \forall x(p(x) \to r(x, y)) \to p(t),$$

mentre $F\{y/t\}$ coincide con F.

ESEMPIO 5.47. A partire dalla formula $\forall z\, r(x, z) \wedge \exists y\, r(x, f(y)) \to \neg\exists x\, r(x, x)$ le sostituzioni $\{x/c\}$ e $\{x/f(a)\}$ conducono rispettivamente a

$$\forall z\, r(c, z) \wedge \exists y\, r(c, f(y)) \to \neg\exists x\, r(x, x),$$
$$\forall z\, r(f(a), z) \wedge \exists y\, r(f(a), f(y)) \to \neg\exists x\, r(x, x).$$

[1] Per comprendere il problema, si consideri ad esempio la sostituzione di x con y nella formula $\forall y\, r(x, y)$: in $\forall y\, r(y, y)$ la variabile y non è libera e quindi questa formula non asserisce di y ciò che la formula di partenza asseriva riguardo a x.

CAPITOLO 6

Semantica della logica predicativa

In questo capitolo svilupperemo la semantica della logica predicativa in analogia a quanto fatto nel capitolo 2 per la semantica della logica proposizionale. La nozione di interpretazione è piuttosto diversa nei due contesti, ma —una volta che essa sia stata stabilita— molte definizioni (conseguenza ed equivalenza logica, validità, ecc.) saranno identiche a quelle del capitolo 2.

1. Interpretazioni e soddisfazione

Per interpretare una formula predicativa dobbiamo prima di tutto interpretare i termini del suo linguaggio. Ad essi assegnamo un elemento di un dominio, che è l'insieme degli oggetti di cui vogliamo parlare. I simboli di costante, funzione e relazione, vanno interpretati con riferimento a questo dominio.

DEFINIZIONE 6.1. Dato un linguaggio $\mathcal{L}$ una *interpretazione* I per $\mathcal{L}$ è data da:

- un insieme non vuoto D^I, detto *dominio* dell'interpretazione;
- per ogni simbolo di costante c in $\mathcal{L}$, un elemento $c^I \in D^I$;
- per ogni simbolo di funzione n-ario f in $\mathcal{L}$, una funzione $f^I : (D^I)^n \to D^I$;
- per ogni simbolo di relazione n-ario p in $\mathcal{L}$, un insieme $p^I \subseteq (D^I)^n$.

DEFINIZIONE 6.2. Un'interpretazione I per il linguaggio $\mathcal{L}$ associa ad ogni termine chiuso t di $\mathcal{L}$ la sua *interpretazione in* I, che è un elemento $t^I \in D^I$ definito ricorsivamente da:

- se t è un simbolo di costante c allora $t^I = c^I$;
- se $t = f(t_1, \ldots, t_n)$ allora $t^I = f^I(t_1^I, \ldots, t_n^I)$.

Se un termine non è chiuso (e quindi contiene delle variabili) l'interpretazione non è sufficiente a stabilire quale elemento del dominio associare al termine. A questo scopo affianchiamo all'interpretazione un modo di interpretare le variabili.

DEFINIZIONE 6.3. Uno *stato* di un'interpretazione I è una funzione che ad ogni variabile associa un elemento del dominio di I.

Uno stato σ di I associa ad ogni termine t un *valore*, $\sigma(t) \in D^I$, definito ricorsivamente da:

- se t è una variabile x allora $\sigma(t) = \sigma(x)$;
- se t è una costante c allora $\sigma(t) = c^I$;
- se $t = f(t_1, \ldots, t_n)$ allora $\sigma(t) = f^I(\sigma(t_1), \ldots, \sigma(t_n))$.

ESERCIZIO 6.4. Dimostrate che se t è un termine chiuso e σ uno stato di un'interpretazione I, si ha $\sigma(t) = t^I$.

ESEMPIO 6.5. Sia $\mathcal{L}$ un linguaggio con un simbolo di costante c, un simbolo funzionale unario f, due simboli relazionali unari p e q e un simbolo relazionale binario r.

Definiamo un'interpretazione I per $\mathcal{L}$ ponendo

$$D^I = \{0, 1, 2\}, \quad c^I = 1, \quad f^I(0) = 1, \quad f^I(1) = 2, \quad f^I(2) = 1,$$
$$p^I = \{1, 2\}, \quad q^I = \{0, 2\}, \quad r^I = \{(0, 0), (0, 2), (1, 2)\}.$$

Definiamo uno stato di I ponendo $\sigma(x) = 0$, $\sigma(y) = 1$ e $\sigma(v) = 2$ per tutte le variabili v diverse da x e y.

L'interpretazione in I del termine chiuso $f(c)$ è $f(c)^I = f^I(c^I) = f^I(1) = 2$. Il valore secondo σ del termine $f(x)$ è $\sigma(f(x)) = f^I(\sigma(x)) = f^I(0) = 1$.

Il prossimo lemma asserisce che il valore assunto da un termine non dipende né dall'interpretazione dei simboli di costante e di funzione che non occorrono nel termine, né dall'interpretazione dei simboli di relazione, né dal valore dello stato su variabili che non occorrono nel termine.

LEMMA 6.6. *Sia t un termine di un linguaggio $\mathcal{L}$, e siano I e I' interpretazioni per $\mathcal{L}$ che hanno lo stesso dominio e coincidono sulle interpretazioni dei simboli di costante e di funzione che occorrono in t. Siano inoltre σ e σ' stati rispettivamente di I e I' che coincidono sulle variabili che occorrono in t. Allora $\sigma(t) = \sigma'(t)$.*

DIMOSTRAZIONE. La dimostrazione è per induzione sulla complessità di t. Se t è una variabile o una costante allora per ipotesi $\sigma(t) = \sigma'(t)$. Se $t = f(t_1, \ldots, t_n)$ allora per ipotesi si ha che f^I e $f^{I'}$ coincidono e, usando l'ipotesi induttiva, abbiamo $\sigma(t) = f^I(\sigma(t_1), \ldots, \sigma(t_n)) = f^{I'}(\sigma'(t_1), \ldots, \sigma'(t_n)) = \sigma'(t)$. □

Per arrivare alla definizione di interpretazione di una formula è opportuno introdurre la seguente notazione per gli stati.

NOTAZIONE 6.7. Se σ è uno stato dell'interpretazione I, x una variabile e $d \in D^I$ indichiamo con $\sigma[x/d]$ lo stato che coincide con σ su tutte le variabili diverse da x, e assegna a x il valore d.

DEFINIZIONE 6.8. Siano F una formula di un linguaggio $\mathcal{L}$, I un'interpretazione per $\mathcal{L}$ e σ uno stato di I. Definiamo la relazione $I, \sigma \models F$ (da leggersi *I allo stato σ soddisfa F*) per ricorsione su F ($I, \sigma \not\models F$ indica che $I, \sigma \models F$ non vale):

- $I, \sigma \models p(t_1, \ldots, t_n)$ se e solo se $(\sigma(t_1), \ldots, \sigma(t_n)) \in p^I$;
- $I, \sigma \models \neg G$ se e solo se $I, \sigma \not\models G$;
- $I, \sigma \models G \wedge H$ se e solo se $I, \sigma \models G$ e $I, \sigma \models H$;
- $I, \sigma \models G \vee H$ se e solo se $I, \sigma \models G$ oppure $I, \sigma \models H$;
- $I, \sigma \models G \to H$ se e solo se $I, \sigma \not\models G$ oppure $I, \sigma \models H$;
- $I, \sigma \models \forall x\, F$ se e solo se per ogni $d \in D$ si ha che $I, \sigma[x/d] \models F$;
- $I, \sigma \models \exists x\, F$ se e solo se per qualche $d \in D$ si ha che $I, \sigma[x/d] \models F$.

Diciamo che I *soddisfa* F, e scriviamo $I \models F$ se $I, \sigma \models F$ per ogni stato σ di I. In questo caso si dice anche che F *è vera in* I oppure che I *è un modello di* F.

Se Γ è un insieme di formule, diciamo che I *allo stato σ soddisfa* Γ, e scriviamo $I, \sigma \models \Gamma$, se I allo stato σ soddisfa ogni $F \in \Gamma$. Anche in questo caso diciamo che I *soddisfa* Γ, o che Γ *è vera in* I oppure che I *è un modello di* Γ, e scriviamo $I \models \Gamma$, se $I, \sigma \models \Gamma$ per ogni stato σ di I.

Questa definizione va vista come una generalizzazione al caso predicativo della definizione 2.3. Pur tenendo conto della differenza tra le interpretazioni proposizionali e quelle predicative, la seconda, terza, quarta e quinta clausola della definizione 6.8 ricalcano le clausole analoghe della definizione 2.3. La prima clausola usa le interpretazioni dei termini e l'interpretazione dei simboli predicativi per sostituire l'assegnazione che attribuiva valori di verità alle lettere proposizionali, mentre completamente nuove sono solo le ultime due clausole che riguardano i quantificatori.

Il simbolo $\models$ è stato usato nel caso proposizionale (e lo sarà anche nel caso predicativo) per indicare la conseguenza logica: per distinguere tra i due significati attribuiti a questo simbolo si veda la nota 6.26.

ESEMPIO 6.9. Siano $\mathcal{L}$, I e σ come nell'esempio 6.5. Sia F la formula

$$(q(f(y)) \to \neg p(c)) \vee \exists z(r(x,z) \wedge p(z) \wedge r(y,z)).$$

Per verificare se $I, \sigma \models F$ iniziamo con lo stabilire se $I, \sigma \models q(f(y)) \to \neg p(c)$. A questo scopo calcoliamo $\sigma(f(y)) = f^I(\sigma(y)) = f^I(1) = 2$; dato che $2 \in q^I$ si ha $I, \sigma \models q(f(y))$ e dobbiamo verificare se $I, \sigma \models \neg p(c)$: dato che $\sigma(c) = c^I = 1$ e $1 \in p^I$ si ha $I, \sigma \models p(c)$ e quindi $I, \sigma \not\models \neg p(c)$. Perciò $I, \sigma \not\models q(f(y)) \to \neg p(c)$ e l'unica possibilità che $I, \sigma \models F$ è che $I, \sigma \models \exists z(r(x,z) \wedge p(z) \wedge r(y,z))$. Per verificare se ciò avviene dobbiamo considerare tutti i possibili stati $\sigma[z/d]$ con $d \in D^I$ e per ognuno di essi controllare se $I, \sigma[z/d] \models r(x,z) \wedge p(z) \wedge r(y,z)$. Si ha che $I, \sigma[z/0] \models r(x,z)$, ma $I, \sigma[z/0] \not\models p(z)$ e $I, \sigma[z/0] \not\models r(y,z)$, così che 0 non è l'elemento adatto. Dato che $I, \sigma[z/1] \not\models r(x,z)$ neppure 1 va bene. Invece $I, \sigma[z/2] \models r(x,z)$, $I, \sigma[z/2] \models p(z)$ e $I, \sigma[z/2] \models r(y,z)$, così che $I, \sigma[z/2] \models r(x,z) \wedge p(z) \wedge r(y,z)$. Perciò $I, \sigma \models \exists z(r(x,z) \wedge p(z) \wedge r(y,z))$ e quindi $I, \sigma \models F$.

Nell'esempio precedente il valore assunto da σ su variabili (come w) che non occorrono in F è del tutto irrilevante. Anche il fatto che $\sigma(z) = 2$ non è mai stato utilizzato, malgrado z occorra in F. Ciò avviene perché z non è libera in F e pertanto quando abbiamo utilizzato il valore di uno stato su z si è sempre trattato di stati della forma $\sigma[z/d]$. Queste osservazioni conducono ad un lemma che corrisponde ai lemmi 6.6 (per quanto riguarda i termini) e 2.8 (nel caso proposizionale) e che asserisce che la soddisfazione di una formula non dipende né dall'interpretazione dei simboli di costante, di funzione e di relazione che non occorrono nella formula né dal valore dello stato su variabili che non occorrono libere nella formula.

LEMMA 6.10. *Sia F una formula di un linguaggio $\mathcal{L}$, e siano I e I' interpretazioni per $\mathcal{L}$ che hanno lo stesso dominio e coincidono sui simboli di costante, di funzione e di relazione che occorrono in F. Siano inoltre σ e σ' stati rispettivamente di I e I' che coincidono sulle variabili che occorrono libere in F. Allora $I, \sigma \models F$ se e solo se $I', \sigma' \models F$.*

DIMOSTRAZIONE. La dimostrazione è per induzione sulla complessità di F.

Se F è una formula atomica basta applicare la definizione 6.8 e il lemma 6.6.

Se F è del tipo $\neg G$, $G \wedge H$, $G \vee H$ oppure $G \to H$ basta applicare la definizione 6.8 e l'ipotesi induttiva.

Se F è del tipo $\forall x\, G$ e si ha $I, \sigma \models F$ allora per ogni $d \in D^I$ $I, \sigma[x/d] \models G$. Per dimostrare che $I', \sigma' \models F$ fissiamo $d \in D^{I'} = D^I$ e consideriamo lo stato $\sigma'[x/d]$: quest'ultimo coincide con $\sigma[x/d]$ sulle variabili libere in G (che, per la definizione 5.34, sono quelle libere in F più eventualmente x) e per ipotesi induttiva si ha $I', \sigma'[x/d] \models G$. Quindi $I', \sigma' \models F$. In maniera del tutto analoga si dimostra che da $I', \sigma' \models F$ segue $I, \sigma \models F$.

Se F è del tipo $\exists x\, G$ il ragionamento è del tutto analogo al caso del quantificatore universale. □

COROLLARIO 6.11. *Se F è un enunciato di $\mathcal{L}$, I un'interpretazione per $\mathcal{L}$ e σ e σ' due stati di I allora $I, \sigma \models F$ se e solo se $I, \sigma' \models F$. Quindi $I \models F$ se e solo se $I, \sigma \models F$ per qualche stato σ di I.*

Il corollario precedente asserisce che possiamo ignorare lo stato nello stabilire la soddisfazione di un enunciato in un'interpretazione. Per la verifica di questa soddisfazione uno stato ausiliario sarà però spesso utile: ad esempio per verificare se $I \models \forall x\, F$, dove F è una formula in cui x è l'unica variabile libera, dobbiamo considerare per ogni $d \in D^I$ se $I, \sigma[x/d] \models F$; in questo caso però possiamo scegliere σ arbitrariamente.

Esercizio 6.12. Siano $\mathcal{L}$, I e σ come nell'esempio 6.5. Sia J l'interpretazione per $\mathcal{L}$ definita da

$$D^J = \{0,1,2\}, \quad c^J = 2 \quad f^J(0) = 1, \quad f^J(1) = 1, \quad f^J(2) = 0,$$
$$p^J = \{0\}, \quad q^J = \{0,1\}, \quad r^J = \{(0,0),(1,2)\}.$$

Sia G l'enunciato

$$\forall x(\exists y\, r(x,y) \to \neg q(x) \vee \forall z(p(f(z)) \to r(z,x))).$$

Stabilite se $I \models G$, se $J \models G$ e se $J, \sigma \models F$, dove F è la formula dell'esempio 6.9 (notate che σ è anche uno stato di J).

Esercizio 6.13. Per ognuno degli enunciati seguenti nel linguaggio dell'esempio 6.5 definite un'interpretazione che lo renda vero ed una che lo renda falso:

$$\forall x\, \exists y\, r(x,y) \wedge \neg\forall x\, p(x); \quad \forall x\, p(x) \vee \forall x\, \neg p(x); \quad p(c) \to \neg p(c); \quad p(c) \to p(a);$$
$$(\exists x\, p(x) \to p(c)) \wedge \neg p(c); \quad (\exists x\, p(x) \to p(c)) \wedge p(f(c)); \quad \exists x\, \neg q(x) \wedge \forall x\, q(f(x))$$
$$\forall x(p(x) \vee q(x)) \to \forall x\, p(x) \vee \forall x\, q(x); \quad \forall x\, \exists y\, r(x,y) \to \exists y\, \forall x\, r(x,y).$$

Esercizio 6.14. Per ognuna delle formule seguenti F, trovate un'interpretazione I e due stati σ_1 e σ_2 di I tali che $I, \sigma_1 \models F$ e $I, \sigma_2 \not\models F$.

$$p(x); \quad \exists y\, r(x,y); \quad r(x,y) \to r(y,x); \quad \exists x\, \neg r(x, f(x)) \wedge r(x, f(x)).$$

Esercizio 6.15. Siano F e G gli enunciati $\forall x(r(x,a) \to r(x,b))$ e $\exists x(r(x,a) \wedge r(x,b))$. Definite interpretazioni I_1, I_2, I_3 e I_4 per il linguaggio $\mathcal{L}(F,G)$, tutte con dominio $\{0,1\}$, tali che:

$$I_1 \models F \quad \text{e} \quad I_1 \models G;$$
$$I_2 \models F \quad \text{e} \quad I_2 \not\models G;$$
$$I_3 \not\models F \quad \text{e} \quad I_3 \models G;$$
$$I_4 \not\models F \quad \text{e} \quad I_4 \not\models G.$$

Esercizio 6.16. Siano F e G gli enunciati

$$\forall x(p(x) \to \neg r(x,x)) \quad \text{e} \quad \neg\exists x(p(x) \wedge \forall y\, r(y,x)).$$

Definite interpretazioni I_1, I_2 e I_3 per $\mathcal{L}(F,G)$, tutte con dominio $\{0,1\}$, tali che:

$$I_1 \models F \quad \text{e} \quad I_1 \models G;$$
$$I_2 \not\models F \quad \text{e} \quad I_2 \models G;$$
$$I_3 \not\models F \quad \text{e} \quad I_3 \not\models G.$$

(Si veda l'esempio 6.27.)

Esercizio 6.17. Sia $\mathcal{L} = \{r\}$ dove r è un simbolo di relazione binario. Siano I e J le interpretazioni per $\mathcal{L}$ definite da: $D^I = \{0\}$, $r^I = \{(0,0)\}$, $D^J = \{1,2\}$, $r^J = \{(1,2),(2,1)\}$. Trovate un enunciato vero in I e falso in J. Trovate un enunciato vero in J e falso in I.

Esempio 6.18. Sia H l'enunciato

$$\forall x\, \forall y(r(x,y) \to \neg r(y,x)) \wedge \forall x\, \neg r(x,x) \wedge \forall x\, \exists y\, r(x,y).$$

Sia I un'interpretazione che soddisfa H. Se I avesse cardinalità 1 (cioè D^I avesse un solo elemento, chiamiamolo 0), dato che $I \models \forall x\, \exists y\, r(x,y)$, dovremmo avere $(0,0) \in r^I$, contro $I \models \forall x\, \neg r(x,x)$. Se I avesse cardinalità 2 (cioè D^I avesse esattamente due elementi, diciamo 0 e 1), ragionando in modo analogo avremmo $(0,1) \in r^I$ e $(1,0) \in r^I$, contro $I \models \forall x\, \forall y(r(x,y) \to \neg r(y,x))$. Abbiamo quindi dimostrato che H non ha modelli di cardinalità 1 o 2.

Esercizio 6.19. Sia H l'enunciato dell'esempio 6.18. Definite un modello di cardinalità 3 per H.

Esercizio 6.20. Siano F e G le formule

$$\forall x(p(x,x) \to p(f(x),x)) \quad \text{e} \quad \exists x\, \neg p(x,x) \wedge \forall x\, p(x, f(x)).$$

Definite interpretazioni I_1, I_2 e I_3 per $\mathcal{L}(F,G)$, tutte con dominio $D = \{0,1\}$, tali che:

$$\begin{array}{ccc} I_1 \models F & \text{e} & I_1 \models G; \\ I_2 \models F & \text{e} & I_2 \not\models G; \\ I_3 \not\models F & \text{e} & I_3 \not\models G. \end{array}$$

Definite un'interpretazione J per $\mathcal{L}(F,G)$ con dominio $D' = \{0,1,2\}$ tale che $J \not\models F$ e $J \models G$.

Esercizio 6.21. ($\star$) Siano I un'interpretazione, σ uno stato di I e F una formula. Una sola direzione dell'equivalenza "$I, \sigma \models F$ se e solo se $I, \sigma \models \forall x\, F$" è vera. Quale? Perché?

Esercizio 6.22. ($\star$) Dimostrate che l'enunciato

$$F = \neg p(a) \wedge \forall x(p(f(x)) \to p(x)) \wedge \exists x\, p(x)$$

non è vero in nessuna interpretazione I in cui ogni elemento del dominio è l'interpretazione di un termine chiuso di $\mathcal{L}(F)$ (cioè tale che per ogni $d \in D^I$ esiste un termine chiuso t con $d = t^I$).

Considerate l'interpretazione I che ha come dominio l'insieme dei termini chiusi di $\mathcal{L}(F)$ ($D^I = \{\, f^{(n)}(a) : n \in \mathbb{N} \,\}$) e che interpreta i simboli del linguaggio nel modo seguente: $a^I = a$, $f^I(a) = a$, $f^I(f^{(n)}(a)) = f^{(n+1)}(a)$, per ogni $n \geq 1$, $p^I = \{\, f^{(n)}(a) : n > 0 \,\}$. Dimostrate che $I \models F$. Perché questo risultato non è in contraddizione con quanto dimostrato prima?

2. Equivalenza e conseguenza logica

Come già preannunciato, le definizioni di equivalenza e conseguenza logica sono analoghe alle corrispondenti definizioni nel caso proposizionale (definizioni 2.12, 2.15 e 2.26)

Definizione 6.23. Siano F e G due formule dello stesso linguaggio $\mathcal{L}$. Diciamo che F e G sono *logicamente equivalenti* (in simboli $F \equiv G$) se per ogni interpretazione I per $\mathcal{L}$ e ogni stato σ di I si ha $I, \sigma \models F$ se e solo se $I, \sigma \models G$.

Definizione 6.24. Siano F e G due formule dello stesso linguaggio $\mathcal{L}$. Diciamo che G è *conseguenza logica* di F (in simboli $F \models G$) se per ogni interpretazione I per $\mathcal{L}$ e ogni stato σ di I tali che $I, \sigma \models F$ si ha $I, \sigma \models G$.

Definizione 6.25. Siano Γ e G un insieme di formule ed una formula dello stesso linguaggio $\mathcal{L}$. Diciamo che G è *conseguenza logica* di Γ (e scriviamo $\Gamma \models G$) se per ogni interpretazione I per $\mathcal{L}$ ed ogni stato σ di I tale che $I, \sigma \models \Gamma$ si ha $I, \sigma \models G$. Come nel caso proposizionale, $\models F$ sta ad indicare $\emptyset \models F$.

Nota 6.26. Lo stesso simbolo $\models$ viene usato per denotare sia la nozione di soddisfazione (definizione 6.8) che quella di conseguenza logica (definizioni 6.24 e 6.25). È sempre possibile capire con quale delle due nozioni si ha a che fare semplicemente guardando cosa compare a sinistra di $\models$ (a destra c'è sempre una formula): nel primo caso un'interpretazione eventualmente affiancata da uno stato, nel secondo caso una formula o un insieme di formule.

Molte proprietà dell'equivalenza e della conseguenza logica viste nel caso proposizionale valgono anche nel caso predicativo, spesso con le stesse dimostrazioni: ad esempio l'equivalenza logica è una relazione di equivalenza e la conseguenza logica è riflessiva e transitiva.

ESEMPIO 6.27. Siano F e G le formule dell'esercizio 6.16. Dimostriamo che $F \models G$ e che quindi non può esistere un'interpretazione in cui F è vera ma G è falsa.

Sia dunque I un'interpretazione tale che $I \models F$ (dato che F è un enunciato possiamo non utilizzare lo stato): vogliamo dimostrare che $I \models G$. Procediamo per assurdo e supponiamo $I \not\models G$, cioè $I \models \exists x(p(x) \wedge \forall y\, r(y,x))$. Esiste dunque $d_0 \in D^I$ tale che $d_0 \in p^I$ e per ogni $d \in D^I$, $(d, d_0) \in r^I$. In particolare $(d_0, d_0) \in r^I$ e quindi, per uno stato σ qualunque, $I, \sigma[x/d_0] \not\models p(x) \to \neg r(x,x)$. Dato che $I \models F$, questo non è possibile.

L'interpretazione I_2 costruita nell'esercizio 6.16 mostra che $G \not\models F$ (e quindi $F \not\equiv G$).

L'esempio precedente evidenzia nuovamente come per mostrare che $F \not\models G$ sia sufficiente trovare **una** interpretazione in cui F è vera, ma G è falsa. Invece per mostrare che $F \models G$ è necessario considerare **tutte** le interpretazioni del linguaggio in considerazione (e quindi bisogna ragionare in modo più astratto).

ESERCIZIO 6.28. Dimostrate:

$$\forall x\, \forall y\, F \equiv \forall y\, \forall x\, F;$$
$$\exists x\, \exists y\, F \equiv \exists y\, \exists x\, F;$$
$$\exists y\, \forall x\, F \models \forall x\, \exists y\, F;$$
$$\forall x\, \exists y\, r(x,y) \not\models \exists y\, \forall x\, r(x,y) \text{ e quindi } \forall x\, \exists y\, F \models \exists y\, \forall x\, F \text{ è falso.}$$

ESERCIZIO 6.29. Dimostrate:

$$p(c) \wedge \neg p(f(c)) \models \neg\forall x(p(x) \to p(f(x)));$$
$$\forall y\big(p(y) \to \neg q(f(y))\big), \exists x\big(q(x) \wedge p(f(x))\big) \models \exists x\big(q(x) \wedge \neg q(f(f(x)))\big);$$
$$\exists x(p(x) \wedge q(f(x))) \not\models \exists x(p(x) \wedge q(x));$$
$$\exists x(p(x) \wedge q(f(x))) \models \exists x\, p(x) \wedge \exists x\, q(x);$$
$$\forall x(\neg p(x) \vee \exists y\, q(x,y)), \exists z\, p(f(z)) \models \neg\forall y\, \forall z\, \neg q(f(z), y).$$

ESERCIZIO 6.30. Stabilite se:

$$\forall x\, (\exists y\, r(y,x) \to \forall y\, r(x,y)) \wedge \neg r(a,a) \models \neg r(f(a), a);$$
$$\forall x\, (\exists y\, r(y,x) \to \forall y\, r(x,y)) \wedge r(a,a) \models r(f(a), a).$$

I prossimi due esercizi chiedono di provare enunciati che sono identici ad alcune proprietà dimostrate nel caso proposizionale. Le dimostrazioni nel caso predicativo sono sostanzialmente identiche a quelle già viste, essenzialmente perché non è mai necessario considerare i quantificatori.

ESERCIZIO 6.31. Dimostrate che due formule F e G sono logicamente equivalenti se e solo se $F \models G$ e $G \models F$ (vedere il lemma 2.19).

ESERCIZIO 6.32. Verificate che i lemmi 2.20, 2.23 e 2.29 e l'esercizio 2.30 valgono anche se le formule coinvolte sono formule predicative.

Diverso è il caso del lemma 2.21 sulla sostituzione di sottoformule, la cui dimostrazione procedeva per induzione sulla complessità delle formule: ora è necessario considerare anche il caso dei quantificatori.

LEMMA 6.33. *Se F è una sottoformula di una formula H e $F \equiv G$ allora $H \equiv H'$ dove H' è la formula ottenuta da H rimpiazzando la sottoformula F con G.*

DIMOSTRAZIONE. La dimostrazione è per induzione sulla complessità delle formule H di cui F è sottoformula.

Quando H è F oppure H non è F ed è una negazione, una congiunzione, una disgiunzione o un'implicazione si può ripetere esattamente la dimostrazione del lemma 2.21.

Se H non è F ed è della forma $\forall x\, H_0$ oppure $\exists x\, H_0$, per ipotesi induttiva si ha $H_0 \equiv H_0'$. Per ogni interpretazione I, stato σ e $d \in D^I$ si ha che $I, \sigma[x/d] \models H_0$ se e solo se $I, \sigma[x/d] \models H_0'$. Questo significa che $H \equiv H'$. □

ESEMPIO 6.34. Combinando i lemmi 2.20, 2.23 (validi anche nel caso predicativo per l'esercizio 6.32) e 6.33 è immediato dimostrare che

$$F \wedge \exists x\, G \to \neg(\neg F \vee \forall y\, H) \equiv \neg(\exists x\, G \wedge F) \vee (\exists y\, \neg H \wedge F).$$

LEMMA 6.35. *Se x non è una variabile libera della formula F allora*

$$F \equiv \forall x\, F \equiv \exists x\, F.$$

DIMOSTRAZIONE. Immediata dalla definizione 6.8 e dal lemma 6.10. □

NOTA 6.36. Per F arbitraria non è vero che $F \models \forall x\, F$: vedere l'esercizio 6.21.

Il seguente lemma stabilisce come la relazione (semantica) di soddisfazione interagisce con l'operazione (sintattica) di sostituzione. La sua dimostrazione è piuttosto lunga e la omettiamo (non fa parte del programma d'esame): può essere trovata nell'appendice A.

LEMMA 6.37 (Lemma di Sostituzione). *Siano σ uno stato di un'interpretazione I, x una variabile, t un termine chiuso e F una formula. Allora*

$$I, \sigma \models F\{x/t\} \quad \textit{se e solo se} \quad I, \sigma[x/t^I] \models F.$$

Il prossimo risultato ci sarà utile nella discussione dei tableaux predicativi.

LEMMA 6.38. *Se t è un termine chiuso allora per qualsiasi formula F si ha:*

$$\forall x\, F \models F\{x/t\};$$
$$F\{x/t\} \models \exists x\, F.$$

DIMOSTRAZIONE. Se $I, \sigma \models \forall x\, F$ allora si ha $I, \sigma[x/d] \models F$ per ogni $d \in D^I$ e in particolare questo vale per $d = t^I$. Per il Lemma di Sostituzione 6.37 questo significa che $I, \sigma \models F\{x/t\}$.

Se $I, \sigma \models F\{x/t\}$ allora per il Lemma di Sostituzione 6.37 $I, \sigma[x/t^I] \models F$. Quindi si ha $I, \sigma \models \exists x\, F$. □

3. Alcune equivalenze logiche notevoli

In questa sezione dimostreremo alcune equivalenze logiche che coinvolgono i quantificatori e sono quindi caratteristiche della logica predicativa.

Iniziamo con l'esaminare l'interazione tra negazione e quantificatori.

LEMMA 6.39. *Per ogni formula F si ha*

$$\neg\forall x\, F \equiv \exists x\, \neg F,$$
$$\neg\exists x\, F \equiv \forall x\, \neg F.$$

DIMOSTRAZIONE. La dimostrazione consiste nello stabilire quattro conseguenze logiche, due per ogni equivalenza logica da dimostrare.

$\boxed{\neg\forall x\, F \models \exists x \neg F}$ Se $I, \sigma \models \neg\forall x\, F$ allora $I, \sigma \not\models \forall x\, F$, e quindi non è vero che per ogni $d \in D^I$ si ha $I, \sigma[x/d] \models F$. Esiste dunque $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \not\models F$, e quindi tale che $I, \sigma[x/d_0] \models \neg F$. Di conseguenza $I, \sigma \models \exists x \neg F$.

$\boxed{\exists x \neg F \models \neg\forall x\, F}$ Se $I, \sigma \models \exists x \neg F$ esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models \neg F$, cioè $I, \sigma[x/d_0] \not\models F$. Perciò $I, \sigma \not\models \forall x\, F$, e quindi si ha che $I, \sigma \models \neg\forall x F$.

$\boxed{\neg\exists x\, F \models \forall x \neg F}$ Se $I, \sigma \models \neg\exists x\, F$ allora $I, \sigma \not\models \exists x\, F$. Qualunque sia $d \in D^I$, si ha allora che $I, \sigma[x/d] \not\models F$, cioè $I, \sigma[x/d] \models \neg F$. Di conseguenza $I, \sigma \models \forall x \neg F$.

$\boxed{\forall x \neg F \models \neg\exists x\, F}$ Se $I, \sigma \models \forall x \neg F$ allora per ogni $d \in D^I$ si ha $I, \sigma[x/d] \not\models F$. Non esiste dunque alcun $d \in D^I$ tale che $I, \sigma[x/d] \models F$ e quindi $I, \sigma \not\models \exists x\, F$. Di conseguenza $I, \sigma \models \neg\exists x\, F$. □

ESERCIZIO 6.40. Date una dimostrazione alternativa della seconda equivalenza logica del lemma 6.39, utilizzando la prima equivalenza logica dello stesso lemma, e i lemmi 2.20.1 e 6.33. Iniziate da $\neg\exists x\, F \equiv \neg\exists x \neg\neg F$.

Passiamo ora a considerare il caso della congiunzione e della disgiunzione.

LEMMA 6.41. *Siano $\circ$ uno qualunque di $\wedge$ e $\vee$ e Q uno qualunque di $\forall$ e $\exists$. Per ogni formula F, ogni variabile x e ogni formula G in cui x non è libera, si ha*

$$Qx\, F \circ G \equiv Qx(F \circ G),$$
$$G \circ Qx\, F \equiv Qx(G \circ F).$$

DIMOSTRAZIONE. Ognuna delle equivalenze logiche contenute nella seconda riga segue da quella corrispondente contenuta nella prima riga usando il lemma 2.20.2 o 2.20.3 e il lemma 6.33. Restano quindi da dimostrare le quattro equivalenze logiche (e quindi le otto conseguenze logiche) contenute nella prima riga. Un utile esercizio consiste nel dimostrare autonomamente queste conseguenze logiche.

$\boxed{\forall x\, F \wedge G \models \forall x(F \wedge G)}$ Se $I, \sigma \models \forall x\, F \wedge G$, allora $I, \sigma \models \forall x\, F$ e $I, \sigma \models G$. Da $I, \sigma \models \forall x\, F$ si ha che per ogni $d \in D^I$, $I, \sigma[x/d] \models F$. Inoltre, da $I, \sigma \models G$, dato che x non è libera in G, si ha anche che $I, \sigma[x/d] \models G$. Di conseguenza $I, \sigma[x/d] \models F \wedge G$. Poiché questo è vero per ogni $d \in D^I$, segue che $I, \sigma \models \forall x(F \wedge G)$.

$\boxed{\forall x(F \wedge G) \models \forall x\, F \wedge G}$ Se $I, \sigma \models \forall x(F \wedge G)$, allora per ogni $d \in D^I$ si ha che $I, \sigma[x/d] \models F$ e $I, \sigma[x/d] \models G$. Dal fatto che per ogni $d \in D^I$ si abbia $I, \sigma[x/d] \models F$, segue che $I, \sigma \models \forall x\, F$. D'altra parte quando $d = \sigma(x)$, $\sigma[x/d]$ è σ, e perciò da $I, \sigma[x/d] \models G$, segue che $I, \sigma \models G$. Di conseguenza $I, \sigma \models \forall x\, F \wedge G$.

$\boxed{\forall x\, F \vee G \models \forall x(F \vee G)}$ Se $I, \sigma \models \forall x\, F \vee G$, allora $I, \sigma \models \forall x\, F$ oppure $I, \sigma \models G$. Se $I, \sigma \models \forall x\, F$, allora per ogni $d \in D^I$, $I, \sigma[x/d] \models F$ da cui segue che per ogni $d \in D^I$, $I, \sigma[x/d] \models F \vee G$ e quindi che $I, \sigma \models \forall x(F \vee G)$. D'altro canto se $I, \sigma \models G$, allora per ogni $d \in D^I$, $I, \sigma[x/d] \models G$, in quanto x non occorre libera in G. Ne segue anche in questo caso, che per ogni $d \in D^I$, $I, \sigma[x/d] \models F \vee G$ e quindi che $I, \sigma \models \forall x(F \vee G)$.

$\boxed{\forall x(F \vee G) \models \forall x\, F \vee G}$ Se $I, \sigma \models \forall x(F \vee G)$ allora dimostreremo che $I, \sigma \models \forall x\, F \vee G$, sia se $I, \sigma \models G$, che se $I, \sigma \not\models G$. Se $I, \sigma \models G$, ovviamente $I, \sigma \models \forall x\, F \vee G$. Se invece $I, \sigma \not\models G$, per ogni $d \in D^I$, $I, \sigma[x/d] \not\models G$, in quanto x non occorre libera in G. Poiché per ogni $d \in D^I$, $I, \sigma[x/d] \models F \vee G$, ne segue che per ogni $d \in D^I$, $I, \sigma[x/d] \models F$. Quindi $I, \sigma \models \forall x\, F$ da cui segue che $I, \sigma \models \forall x\, F \vee G$.

$\boxed{\exists x\, F \wedge G \models \exists x(F \wedge G)}$ Se $I, \sigma \models \exists x\, F \wedge G$, allora $I, \sigma \models \exists x\, F$ e $I, \sigma \models G$. Da $I, \sigma \models \exists x\, F$, segue che esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models F$. D'altra parte dato

che x non occorre libera in G, si ha $I, \sigma[x/d_0] \models G$. Di conseguenza $I, \sigma[x/d_0] \models F \wedge G$, e quindi $I, \sigma \models \exists x(F \wedge G)$.

$\boxed{\exists x(F \wedge G) \models \exists x\, F \wedge G}$ Se $I, \sigma \models \exists x(F \wedge G)$ allora esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models F \wedge G$. Perciò $I, \sigma[x/d_0] \models F$ e $I, \sigma[x/d_0] \models G$. Da $I, \sigma[x/d_0] \models F$ segue che $I, \sigma \models \exists x\, F$. Dato che x non è libera in G da $I, \sigma[x/d_0] \models G$, segue che $I, \sigma \models G$. Di conseguenza $I, \sigma \models \exists x\, F \wedge G$.

$\boxed{\exists x\, F \vee G \models \exists x(F \vee G)}$ Se $I, \sigma \models \exists x\, F \vee G$, allora $I, \sigma \models \exists x\, F$ oppure $I, \sigma \models G$. Nel primo caso esiste $d_0 \in D^I$, tale che $I, \sigma[x/d_0] \models F$, e quindi $I, \sigma[x/d_0] \models F \vee G$, da cui $I, \sigma \models \exists x(F \vee G)$. Nel secondo caso si ha $I, \sigma \models F \vee G$; posto $d_0 = \sigma(x)$, σ è $\sigma[x/d_0]$ e quindi $I, \sigma[x/d_0] \models F \vee G$, da cui segue che $I, \sigma \models \exists x(F \vee G)$. Poiché la conclusione segue in entrambi i casi possibili, possiamo concludere che $I, \sigma \models \exists x(F \vee G)$.

$\boxed{\exists x(F \vee G) \models \exists x\, F \vee G}$ Se $I, \sigma \models \exists x(F \vee G)$ sia $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models F \vee G$, cioè tale che $I, \sigma[x/d_0] \models F$ oppure $I, \sigma[x/d_0] \models G$. Nel primo caso $I, \sigma \models \exists x\, F$ e quindi $I, \sigma \models \exists x\, F \vee G$. Nel secondo caso, dato che x non è libera in G, si ha $I, \sigma \models G$, da cui segue $I, \sigma \models \exists x\, F \vee G$. Poiché la conclusione segue in entrambi i casi possibili, possiamo concludere che $I, \sigma \models \exists x\, F \vee G$. □

Esercizio 6.42. Nelle ipotesi del lemma 6.41 giustificate la seguente catena di equivalenze logiche:

$$\exists x\, F \vee G \equiv \neg\neg(\exists x\, F \vee G) \equiv \neg(\neg\exists x\, F \wedge \neg G) \equiv \neg(\forall x\, \neg F \wedge \neg G) \equiv$$
$$\equiv \neg\forall x(\neg F \wedge \neg G) \equiv \neg\forall x\, \neg(F \vee G) \equiv \neg\neg\exists x(F \vee G) \equiv \exists x(F \vee G).$$

In questo modo abbiamo dimostrato una delle equivalenze logiche riguardanti $\vee$ del lemma 6.41 sulla base (tra l'altro) di una delle equivalenze logiche riguardanti $\wedge$ dello stesso lemma. Analogamente dimostrate $\forall x\, F \vee G \equiv \forall x(F \vee G)$ attraverso una catena di equivalenze logiche.

Passiamo ora a considerare il caso dell'implicazione.

Lemma 6.43. *Per ogni formula F, ogni variabile x e ogni formula G in cui x non è libera, si ha*

$$\forall x\, F \to G \equiv \exists x(F \to G), \qquad G \to \forall x\, F \equiv \forall x(G \to F),$$
$$\exists x\, F \to G \equiv \forall x(F \to G), \qquad G \to \exists x\, F \equiv \exists x(G \to F).$$

Dimostrazione. Questo lemma può essere dimostrato direttamente considerando interpretazioni e stati come fatto per i lemmi 6.39 e 6.41. Ne diamo invece una dimostrazione attraverso la costruzione di catene di equivalenze logiche:

$$\forall x\, F \to G \equiv \neg\forall x\, F \vee G \equiv \exists x\, \neg F \vee G \equiv \exists x(\neg F \vee G) \equiv \exists x(F \to G);$$
$$G \to \forall x\, F \equiv \neg G \vee \forall x\, F \equiv \forall x(\neg G \vee F) \equiv \forall x(G \to F);$$
$$\exists x\, F \to G \equiv \neg\exists x\, F \vee G \equiv \forall x\, \neg F \vee G \equiv \forall x(\neg F \vee G) \equiv \forall x(F \to G);$$
$$G \to \exists x\, F \equiv \neg G \vee \exists x\, F \equiv \exists x(\neg G \vee F) \equiv \exists x(G \to F).$$

Abbiamo utilizzato i lemmi 6.33, 2.23.3, 6.39 e 6.41. □

Esercizio 6.44. Dimostrate il lemma 6.43 utilizzando interpretazioni e stati (dovete dimostrare otto conseguenze logiche).

Esempio 6.45. Se F è $r(x, y)$ e G è $p(y)$ allora $\forall x\, F \wedge \forall y\, G \models \forall x\, \forall y(F \wedge G)$ è falso (un'interpretazione e uno stato che lo mostrano sono ad esempio $D^I = \{0, 1\}$, $p^I = \{0, 1\}$, $r^I = \{(0, 1), (1, 1)\}$, $\sigma(y) = 1$).

Se però y non è libera in F e x non è libera in G allora il lemma 6.41 implica $\forall x\, F \wedge \forall y\, G \equiv \forall x\, \forall y(F \wedge G)$ e quindi la conseguenza logica deve valere.

In qualche caso è possibile trovare una formula logicamente equivalente che contenga un quantificatore in meno di quella di partenza.

LEMMA 6.46. *Per ogni formula F e G si ha*

$$\begin{aligned}\forall x\, F \wedge \forall x\, G &\equiv \forall x(F \wedge G),\\ \exists x\, F \vee \exists x\, G &\equiv \exists x(F \vee G),\\ \forall x\, F \to \exists x\, G &\equiv \exists x(F \to G).\end{aligned}$$

DIMOSTRAZIONE. La prima equivalenza logica può essere dimostrata direttamente, mentre per la seconda stabiliamo le due conseguenze logiche.

$\boxed{\forall x\, F \wedge \forall x\, G \equiv \forall x(F \wedge G)}$ $I, \sigma \models \forall x(F \wedge G)$ se e solo se per ogni $d \in D^I$ si ha $I, \sigma[x/d] \models F$ e $I, \sigma[x/d] \models G$, se e solo se $I, \sigma \models \forall x\, F$ e $I, \sigma \models \forall x\, G$, se e solo se $I, \sigma \models \forall x\, F \wedge \forall x\, G$.

$\boxed{\exists x\, F \vee \exists x\, G \models \exists x(F \vee G)}$ Se $I, \sigma \models \exists x\, F \vee \exists x\, G$ allora $I, \sigma \models \exists x\, F$ oppure $I, \sigma \models \exists x\, G$. Nel primo caso esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models F$, mentre nel secondo esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models G$. In ogni caso $I, \sigma[x/d_0] \models F \vee G$ e quindi $I, \sigma \models \exists x(F \vee G)$.

$\boxed{\exists x(F \vee G) \models \exists x\, F \vee \exists x\, G}$ Se $I, \sigma \models \exists x(F \vee G)$ allora esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models F \vee G$ e quindi $I, \sigma[x/d_0] \models F$ oppure $I, \sigma[x/d_0] \models G$. Nel primo caso $I, \sigma \models \exists x\, F$, nel secondo caso $I, \sigma \models \exists x\, G$; in ogni caso $I, \sigma \models \exists x\, F \vee \exists x\, G$.

Per dimostrare la terza equivalenza logica osserviamo che

$$\begin{aligned}\forall x\, F \to \exists x\, G &\equiv \neg\forall x\, F \vee \exists x\, G\\ &\equiv \exists x\, \neg F \vee \exists x\, G\\ &\equiv \exists x(\neg F \vee G)\\ &\equiv \exists x(F \to G),\end{aligned}$$

dove i passaggi sono giustificati dalla seconda equivalenza logica, dai lemmi 6.33, 6.39 e 2.23.3. □

ESERCIZIO 6.47. Dimostrate l'equivalenza logica nella terza riga del lemma 6.46 utilizzando interpretazioni e stati.

ESERCIZIO 6.48. Dimostrate le seguenti conseguenze logiche ma che le formule non sono logicamente equivalenti (cioè che la conseguenza logica inversa è falsa):

$$\begin{aligned}\exists x(p(x) \wedge q(x)) &\models \exists x\, p(x) \wedge \exists x\, q(x);\\ \forall x\, p(x) \vee \forall x\, q(x) &\models \forall x(p(x) \vee q(x));\\ \exists x\, p(x) \to \forall x\, q(x) &\models \forall x(p(x) \to q(x)).\end{aligned}$$

NOTA 6.49. L'esercizio 6.48 mostra che

$$\begin{aligned}\exists x\, F \wedge \exists x\, G &\not\equiv \exists x(F \wedge G),\\ \forall x\, F \vee \forall x\, G &\not\equiv \forall x(F \vee G),\\ \exists x\, F \to \forall x\, G &\not\equiv \forall x(F \to G).\end{aligned}$$

Per memorizzare le equivalenze logiche del lemma 6.46 è utile notare che $\forall$ e $\exists$ possono essere considerati come estensioni rispettivamente di $\wedge$ e $\vee$ a tutti gli elementi del dominio ($\forall x\, p(x)$ asserisce che p è vera per questo elemento del dominio **e** quest'altro, **e** quest'altro…; $\exists x\, p(x)$ asserisce che p è vera per questo elemento del dominio **oppure** per quest'altro, **oppure** per quest'altro…). Inoltre è bene ricordare che $\to$ è equivalente a una disgiunzione (lemma 2.23.3) e quindi si comporta come $\vee$.

4. Validità e soddisfacibilità

Le definizioni di validità e soddisfacibilità sono analoghe alle corrispondenti definizioni nel caso proposizionale (definizioni 2.31 e 2.40).

DEFINIZIONE 6.50. Se F è una formula diciamo che

- F è *valida* se F è soddisfatta da ogni interpretazione per $\mathcal{L}(F)$;
- F è *soddisfacibile* se F è soddisfatta da qualche interpretazione per $\mathcal{L}(F)$;
- F è *insoddisfacibile* se non esiste un'interpretazione per $\mathcal{L}(F)$ che soddisfa F.

Se Γ è un insieme di formule diciamo che

- Γ è *valido* se ogni interpretazione per $\mathcal{L}(\Gamma)$ soddisfa Γ, cioè soddisfa ogni $F \in \Gamma$;
- Γ è *soddisfacibile* se qualche interpretazione per $\mathcal{L}(\Gamma)$ soddisfa Γ, cioè soddisfa ogni $F \in \Gamma$;
- Γ è *insoddisfacibile* se ogni interpretazione per $\mathcal{L}(\Gamma)$ non soddisfa Γ, cioè non soddisfa qualche $F \in \Gamma$ (F può dipendere dall'interpretazione).

Come nel caso proposizionale sia per singole formule che per insiemi di formule essere insoddisfacibile è equivalente a non essere soddisfacibile.

ESEMPIO 6.51. La formula $p(a) \vee \neg p(a)$ è valida. La formula $p(a) \wedge \neg p(a)$ è insoddisfacibile. La formula $p(a)$ è soddisfacibile ma non valida. $\forall x\, p(x) \to p(a)$ e $p(a) \to \exists x\, p(x)$ sono valide, $\exists x\, p(x) \wedge \forall x\, \neg p(x)$ è insoddisfacibile. $p(a) \wedge \exists x\, \neg p(x)$ è soddisfacibile ma non valida.

Più in generale, per ogni F le formule $\forall x\, F \to F\{x/a\}$ e $F\{x/a\} \to \exists x\, F$ sono valide, mentre $\exists x\, F \wedge \forall x\, \neg F$ è insoddisfacibile.

NOTA 6.52. Come già nel caso proposizionale un insieme di formule è valido se e solo se tutti i suoi elementi sono validi. La proprietà analoga non è vera però per soddisfacibilità e insoddisfacibilità.

ESERCIZIO 6.53. Dimostrate che ogni formula atomica è soddisfacibile e non valida.

ESEMPIO 6.54. Verifichiamo che l'enunciato

$$\exists x\, p(x) \wedge \forall x(p(x) \to \forall y\, q(x, y)) \to \exists x\, q(x, x).$$

è valido.

A questo scopo consideriamo una qualunque interpretazione I per il linguaggio $\{p, q\}$ con p simbolo di relazione unario e q simbolo di relazione binario. Se $I \models \exists x\, p(x) \wedge \forall x(p(x) \to \forall y\, q(x, y))$ esiste $d_0 \in D^I$ tale che $d_0 \in p^I$ e per qualunque stato σ di I si ha $I, \sigma[x/d_0] \models p(x) \to \forall y\, q(x, y)$. Questi due fatti implicano che per qualunque $d \in D^I$ si ha $(d_0, d) \in q^I$ e quindi in particolare che $(d_0, d_0) \in q^I$. Perciò $I \models \exists x\, q(x, x)$.

Abbiamo quindi verificato che se $I \models \exists x\, p(x) \wedge \forall x(p(x) \to \forall y\, q(x, y))$ allora $I \models \exists x\, q(x, x)$, cioè che per qualunque I si ha

$$I \models \exists x\, p(x) \wedge \forall x(p(x) \to \forall y\, q(x, y)) \to \exists x\, q(x, x).$$

Molte proprietà viste nel caso proposizionale si trasferiscono a quello predicativo senza alcuna difficoltà. Ad esempio il seguente teorema è uguale al teorema 2.35.

TEOREMA 6.55. *Sia F una formula:*

(a) *F è valida se e solo se $\neg F$ è insoddisfacibile;*
(b) *F è insoddisfacibile se e solo se $\neg F$ è valida.*

ESERCIZIO 6.56. Verificate che i lemmi 2.38, 2.43 e 2.45 valgono anche se le formule coinvolte sono formule predicative.

Il prossimo lemma sarà utile nella discussione dei tableaux predicativi del capitolo 8.

LEMMA 6.57. *Siano* Γ *un insieme di formule,* F *una formula e* a *una costante che non compare né in* Γ *né in* F*. Se* $\Gamma, \exists x\, F$ *è soddisfacibile allora* $\Gamma, F\{x/a\}$ *è soddisfacibile.*

DIMOSTRAZIONE. Siano I e σ un'interpretazione e uno stato che soddisfano $\Gamma, \exists x\, F$. Dato che $I, \sigma \models \exists x\, F$ esiste $d_0 \in D^I$ tale che $I, \sigma[x/d_0] \models F$. Definiamo un'interpretazione I' con $D^{I'} = D^I$, interpretando in I' tutti simboli di costante, funzione e relazione diversi da a come sono interpretati in I e ponendo $a^{I'} = d_0$.

Dato che a non compare né in Γ né in F per il lemma 6.10 si ha $I', \sigma \models \Gamma$ e $I', \sigma[x/a^{I'}] \models F$. Per il Lemma di Sostituzione 6.37 abbiamo anche $I', \sigma \models F\{x/a\}$. Quindi I' e σ mostrano la soddisfacibilità di $\Gamma, F\{x/a\}$. □

NOTA 6.58. L'ipotesi su a del lemma 6.57 è necessaria, come testimoniato dal caso in cui $\Gamma = \{\neg p(a)\}$ e F è $p(x)$: $\{\neg p(a), \exists x\, p(x)\}$ è soddisfacibile, ma $\{\neg p(a), p(a)\}$ è insoddisfacibile.

ESERCIZIO 6.59. Dimostrate la validità dei seguenti enunciati:

$$\forall x(p(x) \to q(f(x))) \land \exists x\, p(x) \to \exists x\, q(x)$$

$$\exists x\, \forall y\, r(x, y) \to \forall y\, \exists x\, r(x, y);$$

$$(\star)\ \exists x(p(f(x)) \to p(x));$$

$$(\star)\ \neg\exists x\, \forall y((r(x, y) \to \neg r(y, y)) \land (r(y, y) \to \neg r(x, y))).$$

ESERCIZIO 6.60. Dimostrate che l'enunciato

$$\forall x\big(p(x) \to (\neg q(f(x), x) \land \forall y\, q(x, y))\big) \land \exists x\big(p(x) \land p(f(x))\big)$$

è insoddisfacibile.

ESERCIZIO 6.61. ($\star$) Dimostrate che l'insieme

$$\Gamma = \{\exists x\, \neg p(x)\} \cup \{\, p(z) \,:\, z \in \mathrm{Var}\,\}$$

(Var è l'insieme di tutte le variabili) è soddisfacibile.

ESERCIZIO 6.62. Dimostrate che l'enunciato

$$\forall x\, (\forall y(r(x, y) \to p(y)) \land \exists z \neg p(z) \to \exists z \neg r(x, z))$$

è valido.

ESERCIZIO 6.63. Siano F e G gli enunciati $\forall x(p(x) \to \neg p(f(x)))$ e $\exists x\, p(f(x))$. Per ognuno dei quattro insiemi $\{F, G\}$, $\{F, \neg G\}$, $\{\neg F, G\}$ e $\{\neg F, \neg G\}$ costruite se possibile un'interpretazione con dominio $\mathbb{N}$. Nell'unico caso in cui ciò non è possibile dimostrate che l'insieme è insoddisfacibile.

ESERCIZIO 6.64. L'enunciato $\forall y\, \exists x(q(x, y) \to q(y, y))$ è valido? Se la risposta è positiva, dimostratelo, se la risposta è negativa, definite un'interpretazione in cui esso non è soddisfatto.

ESERCIZIO 6.65. Considerate il linguaggio $\mathcal{L} = \{p, r, a\}$, dove p è un simbolo di relazione unario, r un simbolo di relazione binario e a un simbolo di costante. Sia F l'enunciato

$$\big(\exists x\, \exists y\, r(x, y) \land \forall x(r(x, x) \to p(x))\big) \to p(a).$$

(a) Dimostrate che F è vero in ogni interpretazione il cui dominio ha un solo elemento.

(b) F è valido?

ESERCIZIO 6.66. ($\star$) Sia F una formula e c un simbolo di costante che non compare in F.

(a) Dimostrate che se $F\{x/c\}$ è valida allora $\forall x\, F$ è valida.
(b) È vero che per ogni interpretazione I e stato σ vale $I, \sigma \models F\{x/c\}$ se e solo se $I, \sigma \models \forall x\, F$?
(c) Dimostrate con un controesempio che in (a) l'ipotesi su c e F è necessaria.
(d) Dal fatto che c occorra in F segue la falsità di (a)?

CAPITOLO 7

Traduzioni dal linguaggio naturale

In questo capitolo ci occuperemo di tradurre frasi del linguaggio naturale (nel nostro caso, l'italiano) in formule logiche e viceversa. Il problema analogo per la logica proposizionale è stato trattato nella sezione 5 del capitolo 2: la ricchezza espressiva della logica predicativa ci permette di tradurre in modo più preciso molte frasi del linguaggio naturale.

Il metodo migliore per familiarizzarsi con queste traduzioni è naturalmente la pratica. Per questa ragione questo capitolo (così come la sezione 2.5) consiste prevalentemente di esempi ed esercizi.

Anche nel caso predicativo la traduzione dal linguaggio formale al linguaggio naturale non presenta in genere difficoltà (anche se alcune traduzioni, come quella di $\exists x(v(x) \to d(x))$ nel linguaggio dell'esempio 7.1, appaiono poco naturali), mentre la direzione inversa è spesso più delicata.

1. Traduzioni di frasi

ESEMPIO 7.1. Sia $\{a, b, v, d\}$ un linguaggio in cui a e b sono simboli di costante (da interpretarsi rispettivamente come "Andrea" e "Bruna") e v e d sono simboli di relazione unari ($v(x)$ da interpretarsi come "x va alla festa", $d(x)$ come "x si diverte").

La formula $v(b) \wedge d(b) \to \neg v(a)$ viene interpretata come "se Bruna va alla festa e si diverte allora Andrea non va alla festa".

La frase "Andrea va alla festa e Bruna no, oppure Andrea va alla festa e Bruna si diverte" viene tradotta nella formula $(v(a) \wedge \neg v(b)) \vee (v(a) \wedge d(b))$.

La formula $\exists x(v(x) \wedge \neg d(x))$ viene interpretata come "qualcuno va alla festa e non si diverte" oppure "c'è qualcuno che non si diverte, pur andando alla festa".

La frase "tutti quelli che vanno alla festa si divertono" viene tradotta nella formula $\forall x(v(x) \to d(x))$, che traduce anche "chi va alla festa si diverte".

NOTA 7.2. Per tradurre in un linguaggio proposizionale la prima frase dell'esempio 7.1 avremmo scelto alcune lettere proposizionali (ad esempio p per "Bruna va alla festa", q per "Bruna si diverte" e r per "Andrea va alla festa") e la traduzione sarebbe stata $p \wedge q \to \neg r$. In questa formula proposizionale non c'è traccia né del fatto che Bruna è l'oggetto delle prime due affermazioni atomiche né del fatto che Andrea non fà ciò che fà Bruna (divertirsi).

L'espressività della logica predicativa emerge quindi anche se non si usano quantificatori, ma è ancora più evidente nel caso di espressioni che riguardano la totalità degli oggetti considerati: l'unico modo per tradurre "tutti quelli che vanno alla festa si divertono" nella logica proposizionale è quello di introdurre una lettera proposizionale che corrisponda a questa affermazione!

ESEMPIO 7.3. Utilizziamo il linguaggio $\{m, d, c, s\}$ dove m e d sono simboli di relazione unari ($m(x)$ sta per "x è un malato", $d(x)$ sta per "x è un dottore") e c e s sono simboli di relazione binari ($c(x, y)$ sta per "x cura y" e $s(x, y)$ sta per "x stima y").

"Ogni malato non stima se stesso" è tradotto da

$$\forall x(m(x) \to \neg s(x,x)).$$

"Ci sono dottori che curano se stessi" è tradotto da

$$\exists x(d(x) \wedge c(x,x)).$$

"Qualche malato stima tutti i dottori che lo curano" è tradotto da

$$\exists x(m(x) \wedge \forall y(d(y) \wedge c(y,x) \to s(x,y))).$$

"Tutti i malati stimano almeno un dottore che li cura" è tradotto da

$$\forall x(m(x) \to \exists y(d(y) \wedge c(y,x) \wedge s(x,y))).$$

Notate come l'ultima formula sia logicamente equivalente (per il lemma 6.43) a

$$\forall x\, \exists y(m(x) \to d(y) \wedge c(y,x) \wedge s(x,y)),$$

che può quindi essere considerata un'altra traduzione corretta della frase in esame.

NOTA 7.4. Questi esempi evidenziano come $\forall$ è spesso abbinato ad una implicazione (l'antecedente dell'implicazione restringe l'ambito degli elementi a cui si applica il $\forall$, nel caso della prima frase dell'esempio 7.3 ai malati), mentre $\exists$ spesso precede una congiunzione (l'elemento di cui si asserisce l'esistenza ha spesso diverse proprietà, nel caso della seconda frase oltre ad essere un dottore ha la caratteristica di curare se stesso).

Se in una traduzione ci si trova ad avere una quantificazione universale di una congiunzione, o una quantificazione esistenziale di un'implicazione, è bene controllare accuratamente il significato della frase in esame.

ESERCIZIO 7.5. Consideriamo il linguaggio $\{s, c, f, u\}$, dove s, c e f sono simboli di relazione unari (da interpretarsi come "x è uno studente", "x è un computer" e "x è funzionante") mentre u è un simbolo di relazione binario (da interpretarsi come "x utilizza y"). Considerate le seguenti frasi:

(i) Un computer non è utilizzato da nessuno studente.
(ii) Ogni computer funzionante è utilizzato da almeno uno studente.
(iii) Non tutti i computer sono funzionanti.

Quale dei seguenti enunciati è una traduzione di (i)?

$$\exists x(c(x) \wedge \forall y(\neg s(y) \wedge \neg u(y,x)));$$
$$\exists x(c(x) \to \forall y(s(y) \to \neg u(y,x)));$$
$$\exists x(c(x) \wedge \forall y(s(y) \to \neg u(y,x))).$$

Qual è il significato degli altri enunciati? Traducete (ii) e (iii).

Per ora abbiamo considerato solo linguaggi privi di simboli di funzione. Ecco un linguaggio con simboli di funzione.

ESEMPIO 7.6. Sia $\{c, p, r, s\}$ un linguaggio in cui c è un simbolo di costante, p è un simbolo di funzione unario, r e s sono simboli di relazione binari. Interpretiamo c come "Claudio", $p(x)$ come "il padre di x", $r(x,y)$ come "x è parente di y" e $s(x,y)$ come "x stima y".

"Tutti i parenti di Claudio stimati da Claudio, sono stimati anche dal padre di Claudio" è tradotta da

$$\forall x(r(x,c) \wedge s(c,x) \to s(p(c),x)).$$

"Claudio stima se stesso e tutti quelli che stimano suo padre, ma non stima suo padre" è tradotta da

$$s(c,c) \wedge \forall x(s(x,p(c)) \to s(c,x)) \wedge \neg s(c,p(c)).$$

"Claudio stima solo quelli che stimano il loro nonno paterno" è tradotta da

$$\forall x(s(c,x) \to s(x,p(p(x)))).$$

ESEMPIO 7.7. Sia $\{p,c,g,a\}$ un linguaggio dove p è un simbolo di funzione unario, c e g sono simboli di relazione unari, e a è un simbolo di relazione binario. Interpretiamo $p(x)$ come "il padrone di x", $c(x)$ come "x è un cane", $g(x)$ come "x è un gatto", $a(x,y)$ come "x ama y".

"Tutti i cani e i gatti amano i loro padroni" è tradotta da

$$\forall x(c(x) \vee g(x) \to a(x,p(x))).$$

Notiamo come "e" sia stato tradotto da $\vee$: l'enunciato

$$\forall x(c(x) \wedge g(x) \to a(x,p(x)))$$

corrisponde all'affermazione "tutti coloro che sono sia cane che gatto amano i loro padroni", che ha un significato ben diverso.

"Tutti i cani non amano i padroni di un gatto" è tradotta da

$$\forall x(c(x) \to \forall y(g(y) \to \neg a(x,p(y)))),$$

oppure da

$$\forall x\, \forall y(c(x) \wedge g(y) \to \neg a(x,p(y)))$$

(i due enunciati sono logicamente equivalenti).

ESERCIZIO 7.8. Formalizzate "se ogni sorella di Gianni litiga con almeno una sorella di Fabio, il miglior amico di Gianni litiga con il miglior amico di Fabio" utilizzando il linguaggio $\{g,f,s,m,l\}$ dove g,f sono costanti (che denotano rispettivamente "Gianni" e "Fabio"), m è un simbolo di funzione unario ($m(x)$ sta per "il miglior amico di x"), s e l sono simboli di relazione binari ($s(x,y)$ sta per "x è sorella di y" e $l(x,y)$ sta per "x litiga con y").

ESEMPIO 7.9. Utilizziamo il linguaggio dell'esempio 5.4. Ecco alcune traduzioni:

(a) "esistono numeri reali il cui seno e coseno coincidono"

$$\exists x\ \sin(x) = \cos(x);$$

(b) "il seno di un numero reale è minore del coseno di un altro numero reale"

$$\exists x\, \exists y(x \neq y \wedge \sin(x) < \cos(y)).$$

[La sottoformula $x \neq y$, che abbrevia $\neg\, x = y$, è stata inserita per tradurre "altro".]

ESERCIZIO 7.10. Traducete le frasi seguenti utilizzando il linguaggio dell'esempio 5.5:

(a) "la successione vuota è segmento iniziale di ogni successione";
(b) "ogni successione è segmento iniziale della sua concatenazione con un'altra successione";
(c) "ogni successione è segmento iniziale della sua concatenazione con un'altra successione o con se stessa";
(d) "esistono due successioni che non sono l'una segmento iniziale dell'altra";
(e) "qualche successione ha tutte le successioni come segmento iniziale".

ESERCIZIO 7.11. Traducete le frasi seguenti utilizzando il linguaggio dell'esempio 5.6:

(a) "un sottoinsieme di un insieme appartiene all'insieme delle parti di quell'insieme";
(b) "esiste un insieme i cui elementi sono anche suoi sottoinsiemi".

ESERCIZIO 7.12. Introducendo opportuni linguaggi, traducete le frasi seguenti:

(a) "Se tutti gli uomini sono mortali e Socrate è un uomo, allora Socrate è mortale".
(b) "Se tutti i gatti sono animali e Fifì è un gatto, allora Fifì è un animale" (confrontate questa traduzione con quella di (a)).
(c) "Se ogni amico di Mario è amico di Luca e Pietro non è amico di Mario, allora Pietro non è amico di Luca".
(d) "Nessun ladro è onesto".
(e) "Se tutti i filosofi intelligenti sono curiosi e solo i tedeschi sono filosofi intelligenti, allora, se ci sono filosofi intelligenti, qualche tedesco è curioso".
(f) "Il cervello di un delfino è più grande di quello di un topo".
[Suggerimento: utilizzate il linguaggio contenente il simbolo di funzione unario c ("il cervello di x"), il simbolo di relazione binario g ("x è più grande di y") e i simboli di relazione unari d e t ("x è un delfino" e "x è un topo").]
(g) "Se Carlo è più basso di Luca, allora almeno un amico di Carlo è più basso di tutti gli amici di Luca".
(h) "Se tutti gli studenti sono persone serie, tutti gli studenti sono studiosi e tutte le persone serie e studiose non fanno tardi la sera, allora se esiste qualcuno che fa tardi la sera, non tutti sono studenti".

ESERCIZIO 7.13. Utilizzando il linguaggio $\{u, c, d, a\}$ dove u è un simbolo di funzione unario ($u(x)$ sta per "l'ultimo cd di x"), c è un simbolo di relazione unario ($c(x)$ sta per "x è un cantante") e d, a sono simboli di relazione binari ($d(x, y)$ sta per "x è un cd di y" e $a(x, y)$ sta per "x acquista y"), formalizzate le frasi seguenti:

(a) "qualcuno acquista tutti i cd di qualche cantante";
(b) "ogni cd di un cantante è sempre acquistato da qualcuno";
(c) "l'ultimo cd di un cantante è sempre acquistato da qualcuno".

ESERCIZIO 7.14. Introducendo un linguaggio opportuno, traducete le frasi seguenti (che conducono a formule prive di quantificatori e con variabili libere): "x^2 è pari, se x è pari", "una condizione sufficiente affinché x sia dispari é che x sia primo", "una condizione necessaria affinché x^2 sia pari è che x non sia primo".
[Suggerimento: utilizzate i simboli di relazione $p(x)$ ("x è pari"), $pr(x)$ ("x è primo"), il simbolo funzionale $f(x)$ ("il quadrato di x") e considerate "dispari" come la negazione di "pari"].

ESERCIZIO 7.15. Considerate il linguaggio $\{b, d, c, t, a\}$, dove b e d sono simboli di costante che rappresentano Barbara e Donatella, c e t sono simboli di relazione unari ($c(x)$ sta per "x ama il cinema", $t(x)$ sta per "x ama il teatro"), mentre a è un simbolo di relazione binario ($a(x, y)$ sta per "x è amico di y"). Formalizzate nel linguaggio le seguenti frasi:

(a) Chi è amico di qualcuno che ama il cinema, ama il cinema.
(b) Chi ama il teatro, è amico di qualcuno che ama il teatro.
(c) Barbara è amica di Donatella e ama il teatro, ma non il cinema.

ESERCIZIO 7.16. ($\star$) Sia $\mathcal{L} = \{r\}$ dove r è un simbolo di relazione binario. Formalizzate in $\mathcal{L}$ le seguenti proprietà:

(a) r è transitiva;
(b) r non è riflessiva su alcun punto.

Siano F_a e F_b gli enunciati corrispondenti. Dimostrate che l'enunciato

$$\forall x \, \exists y \, r(x, y) \wedge F_a \wedge F_b$$

non è vero in nessuna interpretazione con dominio finito, ma è soddisfacibile.

2. Traduzioni di argomenti

La nozione di conseguenza logica ci permette di analizzare la correttezza dei ragionamenti, come discusso nell'introduzione.

ESEMPIO 7.17. Consideriamo le frasi seguenti:

(i) Tutti gli attori ed i giornalisti invitati alla festa sono in ritardo.
(ii) Qualcuno è puntuale.
(iii) Qualche invitato non è né attore né giornalista.

Utilizziamo il linguaggio $\{a, g, i, p\}$, dove $a(x)$ sta per "x è un attore", $g(x)$ sta per "x è un giornalista", $i(x)$ sta per "x è invitato alla festa" e $p(x)$ sta per "x è puntuale". Le frasi precedenti vengono tradotte rispettivamente da

$$\forall x\,((a(x) \vee g(x)) \wedge i(x) \to \neg p(x))\,; \quad \exists x\, p(x); \quad \exists x(i(x) \wedge \neg a(x) \wedge \neg g(x)).$$

(Notate che in (i) "e" è stato tradotto da $\vee$: vedere l'esempio 7.7.)

Indicando con F, G e H queste tre traduzioni abbiamo che $F, G \not\models H$ (trovate un'interpretazione che lo mostri!) e quindi possiamo dire che la frase in (iii) **non** segue logicamente dalle frasi in (i) e (ii), e quindi il ragionamento che deduce (iii) a partire da (i) e (ii) non è corretto.

ESERCIZIO 7.18. Se nell'esempio precedente (ii) venisse sostituita da "qualche invitato è puntuale" il ragionamento diventerebbe corretto?

ESERCIZIO 7.19. Esaminate alla luce delle conoscenze attuali gli esempi di pagina 1, stabilendo rigorosamente quali dei tre argomenti sono corretti.

ESERCIZIO 7.20. Utilizzando il linguaggio $\{s, p, v, g\}$ dove i quattro simboli sono simboli di relazione predicati unari, da interpretarsi come come "x è stupido", "x è presuntuoso", "x è vanitoso", e "x è simpatico", ed inoltre considerando "antipatico" come la negazione di "simpatico", formalizzate le frasi seguenti:

(i) tutti gli stupidi sono presuntuosi o vanitosi;
(ii) i presuntuosi sono antipatici;
(iii) le persone simpatiche non sono vanitose;
(iv) tutti gli stupidi sono antipatici.

Possiamo dire che la frase in (iv) segue logicamente dalle precedenti?

ESERCIZIO 7.21. Formalizzate in un linguaggio opportuno il seguente argomento: "alcuni studenti apprezzano tutti i professori, ogni studente non apprezza i ciarlatani, dunque nessun professore è un ciarlatano". Verificate la correttezza dell'argomento.

ESERCIZIO 7.22. ($\star$) Sia r un simbolo di relazione binario e sia F l'enunciato

$$\forall x(\exists y\, r(x, y) \to r(x, x)).$$

(a) Dimostrate che F è vero in ogni interpretazione I in cui r^I è una relazione simmetrica e transitiva.
(b) Scrivete un enunciato G che esprima il fatto che r è simmetrica e transitiva. Stabilite se $G \models F$.

ESERCIZIO 7.23. Formalizzate in un opportuno linguaggio le frasi seguenti:

(a) Tutti coloro che scendono dall'aereo tranne i membri dell'equipaggio sono perquisiti da almeno un poliziotto.
(b) Alcuni ladri scendono dall'aereo e sono perquisiti solo da ladri.
(c) Nessun ladro è membro dell'equipaggio.
(d) Alcuni poliziotti sono ladri.

Siano F_a, F_b, F_c e F_d gli enunciati corrispondenti: stabilite se $F_a \wedge F_b \wedge F_c \models F_d$.

CAPITOLO 8

Il metodo dei tableaux: caso predicativo

In questo capitolo il metodo dei tableaux studiato nel capitolo 4 verrà esteso al caso della logica predicativa. L'idea che guida il metodo dei tableaux è la stessa: cerchiamo sistematicamente un'interpretazione che soddisfi l'enunciato di partenza. La maggior complessità della nozione di interpretazione nel caso predicativo rende necessari diversi cambiamenti all'algoritmo. Introdurremo nuove regole per la costruzione di nodi nel caso in cui si agisca su formule con quantificatori e il metodo avrà proprietà piuttosto diverse da quelle dimostrate nel caso della logica proposizionale: l'algoritmo predicativo non ha la proprietà di terminazione ed è solo una procedura di semidecisione per la validità (si veda la nota 8.43).

CONVENZIONE 8.1. Per semplificare la nostra discussione dei tableaux predicativi in questo capitolo ci occupiamo solamente di linguaggi che non contengono nessun simbolo di funzione. Questo significa che gli unici termini chiusi sono i simboli di costante.

1. Letterali, γ e δ-formule

La seguente definizione è la ovvia generalizzazione al caso predicativo delle definizioni date nel caso proposizionale (3.1 e 4.1).

DEFINIZIONE 8.2. Un *letterale* è una formula atomica oppure la negazione di una formula atomica.

Se A è una formula atomica $\{A, \neg A\}$ è una *coppia complementare di letterali*. Più in generale se F è una formula $\{F, \neg F\}$ è una *coppia complementare*. Diciamo che F e $\neg F$ sono ciascuno il *complemento* dell'altro.

La proprietà fondamentale delle coppie complementari è contenuta nel seguente lemma di immediata dimostrazione.

LEMMA 8.3. *Se un insieme di formule contiene una coppia complementare allora è insoddisfacibile.*

La nostra discussione dei tableaux proposizionali si basava sulla distinzione tra doppie negazioni, α-formule e β-formule. Nel caso predicativo le definizioni di questi tipi di formule sono le stesse che nel caso proposizionale (definizione 3.6), ma per poter classificare tutte le formule predicative è necessario introdurre dei nuovi tipi di formule.

DEFINIZIONE 8.4. Una formula è *una γ-formula* se esiste F tale che la formula è di uno dei tipi che compaiono nella colonna sinistra della prima delle seguenti tabelle. Una formula è *una δ-formula* se esiste F tale che la formula è di uno dei tipi che compaiono nella colonna sinistra della seconda delle seguenti tabelle. In entrambi i casi un'*istanza* di una γ- o δ-formula è una formula del tipo che compare nella colonna più a destra, dove a è un simbolo di costante.

γ-formula	
$\forall x\, F$	$F\{x/a\}$
$\neg\exists x\, F$	$\neg F\{x/a\}$

δ-formula	
$\exists x\, F$	$F\{x/a\}$
$\neg\forall x\, F$	$\neg F\{x/a\}$

Diciamo che $F\{x/a\}$ o $\neg F\{x/a\}$ è l'*istanza della γ- o δ-formula relativa ad a.*

NOTA 8.5. Notiamo che se G è una γ- o δ-formula che è un enunciato allora tutte le sue istanze sono enunciati.

I prossimi due lemmi corrispondono al lemma 3.7 nel caso proposizionale, sebbene in questo caso non si ottengano delle equivalenze logiche, ma solo delle conseguenze logiche o dei risultati di soddisfacibilità.

LEMMA 8.6. *Se G è una γ-formula e G_1 una sua istanza allora $G \models G_1$.*

DIMOSTRAZIONE. Per le formule del tipo $\forall x\, F$ questo è un caso particolare del lemma 6.38. Per le formule del tipo $\neg\exists x\, F$ basta usare la seconda equivalenza logica del lemma 6.39. □

LEMMA 8.7. *Siano Γ un insieme di formule, G una δ-formula e G_1 un istanza di G relativa ad una costante che non compare né in Γ né in G. Se Γ, G è soddisfacibile allora Γ, G_1 è soddisfacibile.*

DIMOSTRAZIONE. Per le formule del tipo $\exists x\, F$ questo è il lemma 6.57. Per le formule del tipo $\neg\forall x\, F$ basta usare la prima equivalenza logica del lemma 6.39. □

Il prossimo lemma corrisponde al lemma 3.8 nel caso proposizionale.

LEMMA 8.8. *Una formula è di uno e uno solo dei tipi seguenti:*

- *un letterale,*
- *una doppia negazione,*
- *una α-formula,*
- *una β-formula,*
- *una γ-formula,*
- *una δ-formula.*

DIMOSTRAZIONE. La dimostrazione è analoga a quella del lemma 3.8, basandosi questa volta sul lemma 5.20. La lasciamo come esercizio per il lettore. □

2. Esempi preliminari

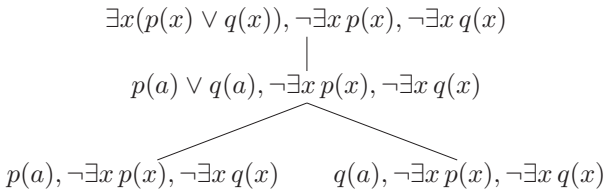
Come nel caso proposizionale, prima di descrivere l'algoritmo dei tableaux esaminiamo in dettaglio alcuni esempi in cui partiamo dalle regole per i tableaux proposizionali già introdotte e le estendiamo alle γ- e δ-formule, evidenziando alcune importanti novità che è necessario introdurre.

ESEMPIO 8.9. Consideriamo l'enunciato $\exists x(p(x) \vee q(x)) \to \exists x\, p(x) \vee \exists x\, q(x)$, che indichiamo con F: è valido per il lemma 6.46 (e l'esercizio 6.56). Per dimostrare la validità di F con il metodo dei tableaux partiamo da $\neg F$, che è una α-formula, e al passo successivo abbiamo un'altra α-formula su cui agire. Il tableau comincia quindi in questo modo:

$$\begin{array}{c} \neg F \\ | \\ \exists x(p(x) \vee q(x)), \neg(\exists x\, p(x) \vee \exists x\, q(x)) \\ | \\ \exists x(p(x) \vee q(x)), \neg\exists x\, p(x), \neg\exists x\, q(x) \end{array}$$

A questo punto il caso proposizionale non ci dice più cosa fare: abbiamo due γ-formule e una δ-formula. La δ-formula $\exists x(p(x) \vee q(x))$ asserisce l'esistenza di un elemento del dominio con certe caratteristiche. Non sappiamo qual è il dominio dell'interpretazione che cerchiamo di costruire: la cosa più conveniente è introdurre una costante che rappresenti l'elemento che sappiamo esistere. Sia a questa

costante: sostituiamo la δ-formula con $p(a) \vee q(a)$, che è l'istanza relativa ad a di $\exists x(p(x) \vee q(x))$. Otteniamo una β-formula e sappiamo come fare il passo successivo:



Siamo nuovamente giunti ad un punto in cui non possiamo applicare le regole proposizionali. La γ-formula $\neg\exists x\, p(x)$ è logicamente equivalente a $\forall x\, \neg p(x)$ (lemma 6.39) e quindi asserisce che per ogni elemento del dominio, ed in particolare per quello rappresentato da a, non vale p. Dunque possiamo sostituire nel ramo di sinistra $\neg\exists x\, p(x)$ con la sua istanza $\neg p(a)$ e ottenere un nodo la cui etichetta

$$p(a), \neg p(a), \neg\exists x\, q(x)$$

contiene una coppia complementare di letterali.

Similmente nel ramo di destra possiamo agire sulla γ-formula $\neg\exists x\, q(x)$ ed ottenere un nodo con etichetta

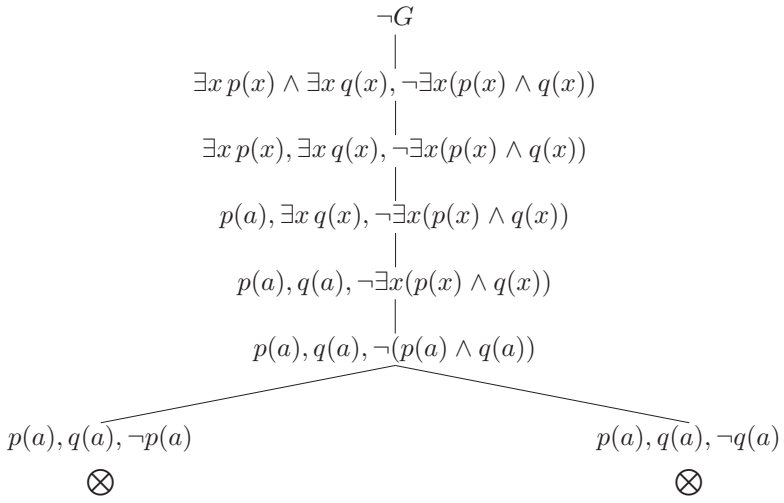
$$q(a), \neg\exists x\, p(x), \neg q(a)$$

Tutte le foglie del tableau contengono dunque una coppia complementare di letterali e quindi il tableau è chiuso, come volevamo.

Esercizio 8.10. Usate le idee dell'esempio 8.9 per mostrare che l'enunciato $\forall x(p(x) \to q(x)) \to (\forall x\, p(x) \to \forall x\, q(x))$ è valido.

Esempio 8.11. Consideriamo l'enunciato $\exists x\, p(x) \wedge \exists x\, q(x) \to \exists x(p(x) \wedge q(x))$, che indichiamo con G. G è soddisfacibile ma non valido (vedere esercizio 6.48), e quindi ci aspettiamo che il tableau per $\neg G$ sia aperto.

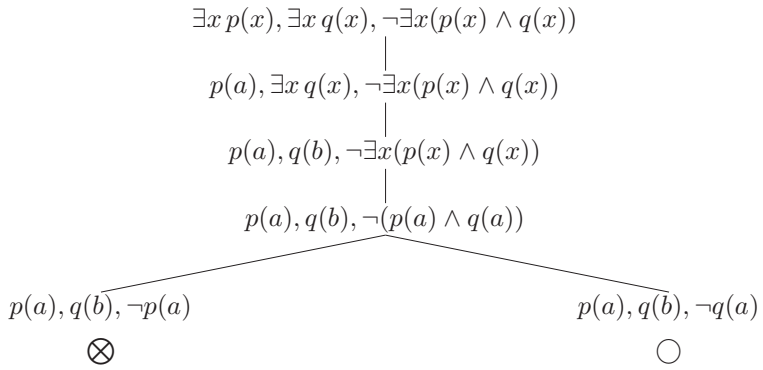
Ecco un tentativo di tableau per $\neg G$, in cui abbiamo utilizzato le idee dell'esempio 8.9:



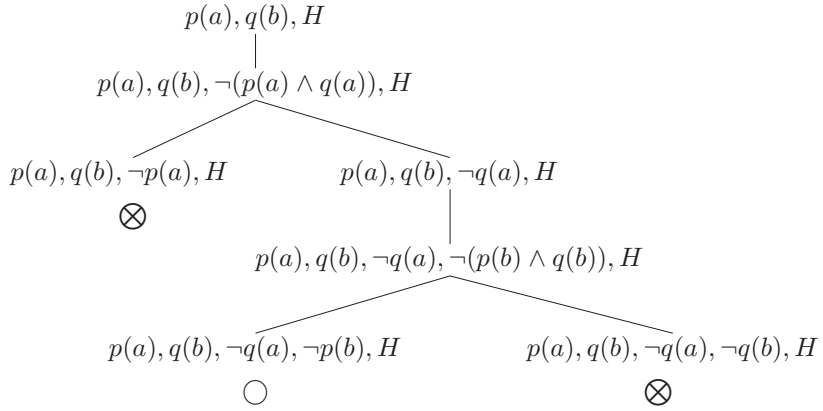
Abbiamo ottenuto un tableau chiuso e quindi qualcosa non ha funzionato. Il problema è che abbiamo usato la stessa costante a per istanziare sia $\exists x\, p(x)$ che $\exists x\, q(x)$: non c'è nessuna ragione di pensare che l'elemento che soddisfa $p(x)$ sia lo stesso che soddisfa $q(x)$!

La soluzione è imporre che quando si utilizza una costante per istanziare una δ-formula esistenziale essa debba essere **nuova**, cioè non comparire nel tableau

sviluppato fino a quel momento. Seguendo questa regola le ultime righe del nostro tentativo di tableau diventano:



C'è però ancora qualcosa di insoddisfacente in questo tentativo di tableau. La γ-formula $\neg\exists x(p(x) \wedge q(x))$ asserisce che $p(x) \wedge q(x)$ non vale per nessun elemento del dominio. Questa informazione è però stata utilizzata solo per a, mentre sarebbe utile poterla sfruttare anche per b. La soluzione è che le γ-formule non vanno mai cancellate dalle etichette dei nodi, per poter essere usate —se necessario— più volte. Otteniamo così la seguente parte finale del tableau, in cui indichiamo con H l'enunciato $\neg\exists x(p(x) \wedge q(x))$:



Il nodo aperto ci suggerisce un'interpretazione che soddisfa $\neg G$: $D^I = \{a, b\}$, $p^I = \{a\}$, $q^I = \{b\}$ (notiamo che non è necessario interpretare i simboli di costante a e b, che non facevano parte del linguaggio di G).

ESEMPIO 8.12. Consideriamo ora l'enunciato F definito da $F_1 \wedge F_2 \wedge F_3$ dove F_1, F_2 e F_3 sono rispettivamente

$$\forall x\, \exists y\, r(x, y);$$
$$\forall x\, \forall y\, \forall z(r(x, y) \wedge r(y, z) \to r(x, z));$$
$$\forall x\, \neg r(x, x).$$

Supponiamo di voler mostrare che F è soddisfacibile, e a questo scopo costruiamo un tableau per F.

Dopo alcuni passaggi che riguardano solo α-formule otteniamo un nodo etichettato da F_1, F_2, F_3, che sono tutte γ-formule. Non abbiamo nessuna costante su cui istanziare le γ-formule, e quindi introduciamo una nuova costante, come fatto nel caso delle δ-formule. Se a_1 è questa costante e consideriamo l'istanza di F_1 relativa ad a_1 otteniamo un nodo etichettato da

$$F_1, F_2, F_3, \exists y\, r(a_1, y)$$

(notare che non abbiamo cancellato la γ-formula utilizzata).

Se ora agiamo sulla δ-formula $\exists y\, r(a_1, y)$ (che possiamo cancellare) introducendo la nuova costante a_2 il nodo successivo sarà etichettato da

$$F_1, F_2, F_3, r(a_1, a_2).$$

Ora F_1 può essere istanziata su a_2 e otteniamo un nodo etichettato da

$$F_1, F_2, F_3, r(a_1, a_2), \exists y\, r(a_2, y)$$

che conduce al nodo etichettato da

$$F_1, F_2, F_3, r(a_1, a_2), r(a_2, a_3).$$

Proseguendo in questo tableau è chiaro che la costruzione non terminerà mai, ma otterremo un ramo infinito le cui etichette conterranno i letterali del tipo $r(a_i, a_{i+1})$. Ciò suggerisce un interpretazione che soddisfa F ed ha dominio infinito: $D^I = \mathbb{N}$, $r^I = \{\, (i, j) \,:\, i < j \,\}$.

In effetti F non ha interpretazioni con dominio finito, come dimostrato nell'esercizio 7.16, e quindi il ramo infinito di questo tableau è inevitabile.

Nota 8.13. L'esempio 8.12 mostra che il metodo dei tableaux non è una procedura di decisione per la soddisfacibilità (o la validità) degli enunciati predicativi: è possibile che la sua esecuzione si prolunghi all'infinito.

Esempio 8.14. Sia F l'enunciato dell'esempio 8.12 e indichiamo con F' l'enunciato $F \wedge p(a_1) \wedge \neg p(a_1)$ (in questo caso a_1 è un simbolo di costante del linguaggio). È evidente che F' è insoddisfacibile. È altrettanto evidente che potremmo ripetere la costruzione del tableau dell'esempio 8.12 aggiungendo semplicemente in ogni etichetta l'enunciato $p(a_1) \wedge \neg p(a_1)$, senza mai agire su di essa. Otterremo dunque un tableau con un ramo infinito, che suggerisce soddisfacibilità dell'enunciato di partenza.

Nota 8.15. Il comportamento dell'esempio 8.14 è dovuto al fatto che le γ-formule non vengono cancellate dalle etichette e quindi si può continuare ad agire su di esse e "dimenticarsi" di qualche altra formula (nel nostro esempio la α-formula $p(a) \wedge \neg p(a)$) mancando così delle opportunità di chiudere il tableau. Notiamo che questo non avveniva nel caso proposizionale, perché ogni formula utilizzata veniva cancellata dall'etichetta e si aveva la proprietà della terminazione forte.

La soluzione a questo problema verrà ottenuta procedendo ad una costruzione *sistematica* del tableau, in cui non ci si potrà "dimenticare" di qualche formula.

Nella prossima sezione descriviamo l'algoritmo dei tableaux predicativi **senza** tener conto del problema evidenziato dall'esempio 8.14. Il tableau quindi non verrà costruito in maniera sistematica, ma questo è sufficiente a dimostrare la correttezza del nostro metodo nella sezione 4. Nella sezione 5 descriveremo come ottenere la costruzione sistematica del tableau, che ci porterà ad un risultato di completezza di cui accenneremo la dimostrazione nella sezione 6.

3. L'algoritmo

Diverse parti della descrizione dell'algoritmo riproducono fedelmente le parti corrispondenti nell'algoritmo corrispondente nel caso proposizionale (algoritmo 4.8): bisogna dunque prestare attenzione soprattutto alle parti nuove, cioè quelle che riguardano γ- e δ-formule. Un'altra differenza è che il fatto che le γ-formule non vengano mai eliminate dalle etichette ha come conseguenza che non possiamo aspettarci di arrivare a foglie la cui etichetta è un insieme di letterali: una foglia verrà chiusa quando contiene una coppia complementare di letterali, anche se a fianco di essa ci sono formule che non sono letterali.

ALGORITMO 8.16. Un tableau per un enunciato F è un albero in cui ogni nodo è etichettato con un insieme finito di enunciati. Il tableau è costruito per stadi $T_0, \ldots, T_i, \ldots$: per ogni i, T_{i+1} è un albero che estende T_i aggiungendo uno o due nodi con le rispettive etichette e lasciando invariate le etichette dei nodi già appartenenti a T_i. L'unione degli alberi T_i è il tableau per F. Se n è un nodo di qualche T_i indichiamo con $E(n)$ l'etichetta di n (che, per quanto detto prima, è la stessa per tutti i T_i cui appartiene n).

All'inizio della costruzione T_0 consiste di un solo nodo (la *radice* dell'albero) etichettato con $\{F\}$. Allo stadio i cerchiamo una foglia n dell'albero T_i tale che $E(n)$ non contiene una coppia complementare di letterali e contiene qualche enunciato che non è un letterale. Se una tale foglia non esiste siamo al termine della costruzione e il tableau è completo. Altrimenti fissiamo n e scegliamo un enunciato $G \in E(n)$ che non è un letterale. Per il lemma 8.8 ci sono cinque possibilità:

(1) se G è una doppia negazione con ridotto G_1 aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$;
(2) se G è una α-formula con ridotti G_1 e G_2 aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1, G_2\}$;
(3) se G è una β-formula con ridotti G_1 e G_2 aggiungiamo due nodi tra loro inconfrontabili n_1 e n_2 sotto n e poniamo $E(n_1) = (E(n) \setminus \{G\}) \cup \{G_1\}$ e $E(n_2) = (E(n) \setminus \{G\}) \cup \{G_2\}$;
(4) se G è una γ-formula sia a un simbolo di costante che compare in $E(n)$ (se nessun simbolo di costante compare in $E(n)$ sia a una nuova costante) e sia G_1 l'istanza di G relativa ad a: aggiungiamo un nodo n' sotto n e poniamo $E(n') = E(n) \cup \{G_1\}$;
(5) se G è una δ-formula sia a una costante che **non** compare in $E(n)$ e sia G_1 l'istanza di G relativa ad a: aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$.

Notiamo che in ogni caso n non è una foglia di T_{i+1} e le etichette dei nuovi nodi contengono solo enunciati.

L'algoritmo che abbiamo appena descritto è non deterministico perché ad ogni passo scegliamo una foglia n con certe caratteristiche e, all'interno di $E(n)$, scegliamo un enunciato che non sia un letterale. Inoltre in alcuni casi è necessario scegliere anche un'istanza della γ-formula considerata. Questo implica che qualche enunciato che non è un letterale può non essere mai scelto (come accade negli esempi 8.12 e 8.14), oppure che qualche istanza di una γ-formula non venga mai utilizzata. Nel caso proposizionale ciò non poteva avvenire, come dimostrato dal teorema 4.14.

Come nel caso proposizionale, gli alberi $T_0, \ldots, T_i, \ldots$ non sono rappresentati separatamente: si deve piuttosto pensare che il tableau "cresce" verso la sua forma finale, che usualmente è l'unica che vediamo.

DEFINIZIONE 8.17. Sia n un nodo del tableau che ha successori nel tableau: la *formula su cui si agisce in* n è la G della descrizione dell'algoritmo.

NOTA 8.18. I nodi su cui non possiamo agire nella costruzione del tableau sono quelli la cui etichetta contiene una coppia complementare di letterali oppure contiene solamente letterali. La costruzione del tableau si arresta se e soltanto se tutte le foglie dell'albero sono di uno di questi tipi.

CONVENZIONE 8.19. Come nel caso proposizionale, per comodità di lettura aggiungeremo sotto i nodi del tableau su cui non possiamo agire uno dei simboli $\bigotimes$ e $\bigcirc$: se l'etichetta del nodo contiene una coppia complementare di letterali useremo $\bigotimes$, altrimenti $\bigcirc$.

Inoltre, per alleggerire la notazione, evitiamo di indicare le parentesi { e } intorno agli elementi di $E(n)$.

NOTA 8.20. È importante notare che la costruzione dei tableaux predicativi **non** gode della proprietà della terminazione forte, come già evidenziato nell'esempio 8.12 e nella nota successiva. È quindi possibile che un tableau non abbia foglie, o solo alcuni dei suoi rami terminino con una foglia mentre altri siano infiniti.

Per semplificare la costruzione dei tableau adottiamo da subito la convenzione 4.33, che riportiamo qui:

CONVENZIONE 8.21. Da questo punto in poi ogniqualvolta nelle etichette di un nodo di un tableau compare una doppia negazione F con ridotto G scriveremo direttamente G, utilizzando la regola della doppia negazione immediatamente e contraendo due nodi in uno.

ESEMPIO 8.22. Costruiamo un tableau per $\neg(\forall x\, \neg p(x) \to \neg\exists x\, p(x))$. In ogni nodo sottolineiamo la formula su cui agiamo in quel nodo e utilizziamo la convenzione 8.21. Notiamo che la γ-formula $\forall x\, \neg p(x)$ non viene mai cancellata.

$$\underline{\neg(\forall x\, \neg p(x) \to \neg\exists x\, p(x))}$$
$$\forall x\, \neg p(x), \underline{\exists x\, p(x)}$$
$$\underline{\forall x\, \neg p(x)}, p(a)$$
$$\forall x\, \neg p(x), \neg p(a), p(a)$$
$$\otimes$$

Ecco un altro tableau per lo stesso enunciato: qui abbiamo agito sulla γ-formula prima di aver eliminato la δ-formula, e questo porta alla creazione di un simbolo di costante "inutile" (in questo caso a) e al dover agire due volte sulla γ-formula (una volta per creare l'istanza relativa a a, che non ci porterà a chiudere alcuna foglia, ed una volta per creare l'istanza relativa a b).

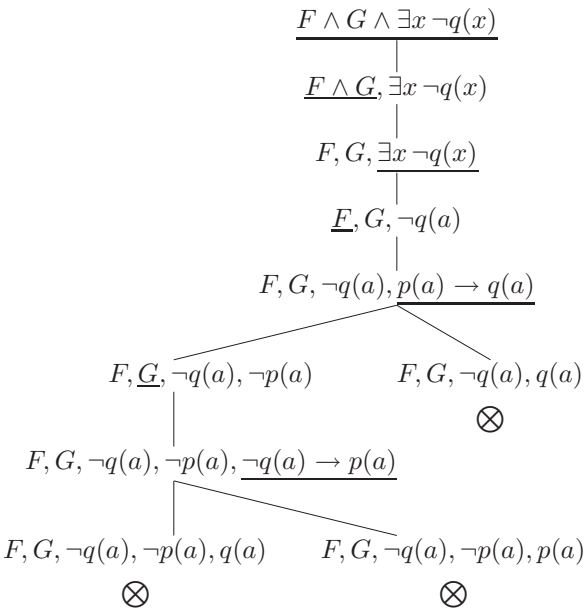
$$\underline{\neg(\forall x\, \neg p(x) \to \neg\exists x\, p(x))}$$
$$\underline{\forall x\, \neg p(x)}, \exists x\, p(x)$$
$$\forall x\, \neg p(x), \neg p(a), \underline{\exists x\, p(x)}$$
$$\underline{\forall x\, \neg p(x)}, \neg p(a), p(b)$$
$$\forall x\, \neg p(x), \neg p(b), \neg p(a), p(b)$$
$$\otimes$$

NOTA 8.23. Per semplificare il più possibile il tableau costruito dall'algoritmo 8.16 è opportuno agire sulle formule disponibili nel seguente ordine: doppie negazioni o α-formule, β-formule, δ-formule, γ-formule. In questo modo si dilaziona il più possibile l'introduzione di nuove costanti e si usano prima le formule che possono essere eliminate.

ESEMPIO 8.24. Costruiamo un tableau per l'enunciato

$$\forall x(p(x) \to q(x)) \land \forall x(\neg q(x) \to p(x)) \land \exists x\, \neg q(x).$$

Indichiamo con F e G gli enunciati $\forall x(p(x) \to q(x))$ e $\forall x(\neg q(x) \to p(x))$. In ogni passaggio sottolineiamo la formula su cui agiamo.



Esercizio 8.25. Costruite un tableau per

$$\exists x\, p(x) \wedge \forall x\, (\exists y\, r(x, y) \to \neg p(x)) \wedge \forall x\, r(x, a).$$

4. Il teorema di correttezza

In questa sezione dimostreremo l'analogo per i tableaux predicativi del teorema 4.23 per i tableaux proposizionali. Anche in questo caso iniziamo con le definizioni di tableau chiuso e aperto.

Definizione 8.26. Un tableau è *chiuso* se non ha rami infiniti e tutte le sue foglie sono etichettate con insiemi di enunciati che contengono una coppia complementare di letterali. Un tableau è *aperto* se non è chiuso, cioè se contiene un ramo infinito oppure una foglia etichettata con un insieme di letterali che non contiene coppie complementari.

Un *ramo aperto* di un tableau è un ramo infinito oppure un ramo che collega la radice dell'albero con una foglia etichettata con un insieme di letterali che non contiene coppie complementari.

Teorema 8.27 (Teorema di correttezza). *Se T è un tableau chiuso per l'enunciato F allora F è insoddisfacibile.*

Dimostrazione. La dimostrazione ricalca da vicino quella del caso proposizionale. In particolare dimostreremo un fatto più generale, che indichiamo nuovamente con ✠:

> sia T un tableau e n un nodo di T tale che il sottoalbero di T che ha radice in n è chiuso; allora l'insieme di enunciati $E(n)$ è insoddisfacibile.

Il teorema è il caso particolare di ✠ ottenuto considerando T il tableau chiuso per F e n la radice di T (ricordate che in questo caso $E(n) = \{F\}$).

Come nel caso proposizionale, la dimostrazione di ✠ è per induzione sull'altezza del nodo n in T.

Il caso in cui l'altezza di n in T è 0 è immediato per il lemma 8.3.

Se l'altezza di n in T è maggiore di 0, allora n ha uno o due successori in T, che sono stati ottenuti agendo su qualche $G \in E(n)$ e ci sono cinque possibilità.

Nei primi tre casi, cioè quando G è una doppia negazione, una α-formula o una β-formula, la dimostrazione del caso proposizionale può essere ripetuta tale e quale, cambiando solamente la notazione utilizzata per le interpretazioni. Restano dunque solo i casi delle γ- e δ-formule.

(4) se G è una γ-formula, n ha un solo successore n' e si ha $E(n') = E(n) \cup \{G_1\}$ dove G_1 è un'istanza di G. Ragionando come negli altri casi e sfruttando l'ipotesi induttiva si ottiene che $E(n')$ è insoddisfacibile. Supponiamo per assurdo che $I, \sigma \models E(n)$. Dato che $G \models G_1$ (lemma 8.6) e $I, \sigma \models G$ (perché $G \in E(n)$) si ha anche $I, \sigma \models G_1$. Quindi $I, \sigma \models E(n')$, contro l'insoddisfacibilità di $E(n')$.

(5) se G è una δ-formula, n ha un solo successore n' e si ha $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$ dove G_1 è un'istanza di G relativa ad un simbolo di costante a che non compare in $E(n)$. Sfruttando l'ipotesi induttiva si ottiene che $E(n')$ è insoddisfacibile. Sia $\Gamma = E(n) \setminus \{G\}$. Il lemma 8.7 implica che se $E(n) = \Gamma, G$ fosse soddisfacibile lo sarebbe anche $E(n') = \Gamma, G_1$. Quindi $E(n)$ è insoddisfacibile.

Abbiamo dunque dimostrato ✠ e quindi il teorema. □

COROLLARIO 8.28. *Se un tableau per l'enunciato $\neg F$ è chiuso allora F è valido.*

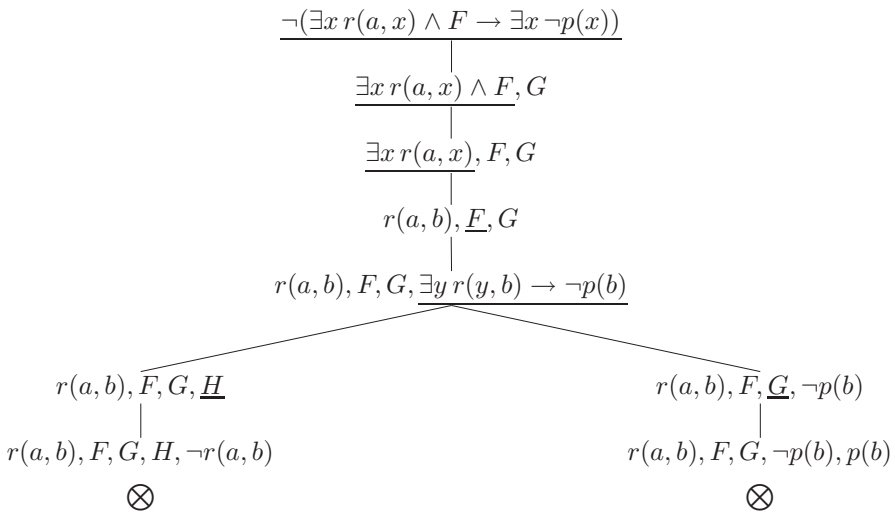
DIMOSTRAZIONE. Immediata dai teoremi 8.27 e 6.55. □

Il corollario 8.28 è molto importante: infatti l'algoritmo dei tableaux predicativi serve principalmente a mostrare che un enunciato è valido.

ESEMPIO 8.29. Costruiamo un tableau per mostrare la validità dell'enunciato

$$\exists x\, r(a,x) \land \forall x(\exists y\, r(y,x) \to \neg p(x)) \to \exists x\, \neg p(x).$$

Indichiamo con F l'enunciato $\forall x(\exists y\, r(y,x) \to \neg p(x))$, con G l'enunciato $\neg\exists x\, \neg p(x)$ e con H l'enunciato $\neg\exists y\, r(y,b)$. In ogni passaggio sottolineiamo la formula su cui agiamo.

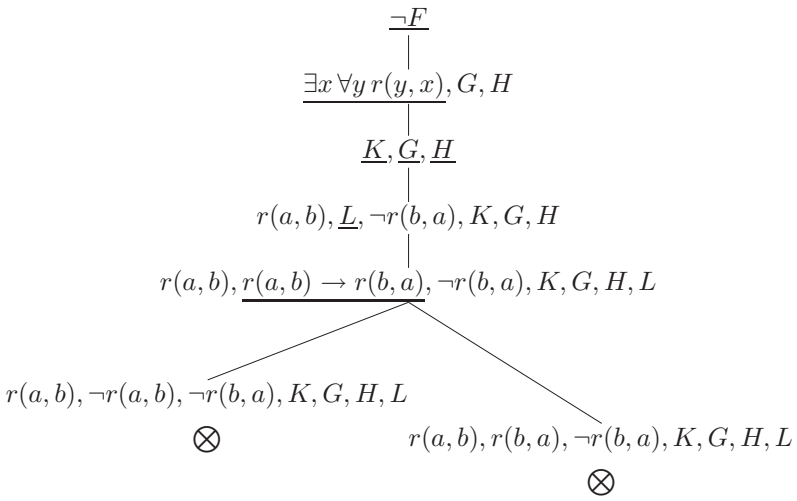


Notate come nel passaggio in cui abbiamo agito su F avremmo potuto aggiungere l'istanza di F relativa a a (anziché, come abbiamo fatto, quella relativa a b). La scelta di b è però apparsa migliore perché il nostro obiettivo è chiudere il tableau: in particolare la presenza di $r(a,b)$ nell'etichetta del nodo ha suggerito di creare un'istanza in cui b apparisse come secondo argomento di r. Considerazioni analoghe possono essere fatte per i nodi in cui abbiamo agito su H e su G.

ESEMPIO 8.30. Sia F l'enunciato

$$\exists x\,\forall y\,r(y,x) \wedge \forall x\,\forall y(r(y,x) \to r(x,y)) \to \exists x\,r(x,a).$$

Usiamo il metodo dei tableaux per mostrare che F è valida. Indichiamo con G l'enunciato $\forall x\,\forall y(r(y,x) \to r(x,y))$, con H l'enunciato $\neg\exists x\,r(x,a)$, con K l'enunciato $\forall y\,r(y,b)$ e con L l'enunciato $\forall y(r(y,b) \to r(b,y))$. Come al solito sottolineiamo le formule su cui agiamo: visto che qualche passaggio è condensato, in qualche caso si agisce su più di una formula.



Anche in questo caso siamo riusciti ad ottenere una chiusura piuttosto rapida del tableau scegliendo opportunamente i simboli di costante su cui istanziare le γ-formule.

ESERCIZIO 8.31. Studiate con il metodo del tableaux gli enunciati degli esempi 6.51 e 6.54 e degli esercizi 6.13, 6.18, 6.59, 6.60 e 6.62 che non contengono simboli di funzione. Verificate che il risultato ottenuto con i tableaux coincida con quello ottenuto in precedenza.

5. La costruzione sistematica dei tableaux

Come evidenziato dall'esempio 8.14, l'algoritmo 8.16 non è sufficiente per dimostrare il teorema di completezza per i tableaux predicativi: se F è insoddisfacibile è possibile che un tableau per F non sia chiuso. Come già suggerito nella nota 8.15, la soluzione a questo problema è rendere l'algoritmo 8.16 "sistematico", eliminando alcuni aspetti di non determinismo che esso contiene.

In pratica si tratta di fare ogni possibile tentativo per chiudere il tableau, non "dimenticando" nessuna formula e considerando ogni possibile istanza delle γ-formule che si incontrano (può essere utile riesaminare l'esempio 8.11, ed in particolare le considerazioni che hanno portato alla versione finale del tableau).

In sostanza vogliamo fare in modo che se r è un ramo aperto di un tableau e $\Sigma = \bigcup_{n\in r} E(n)$:

(a) se $G \in \Sigma$ non è un letterale allora in qualche nodo di r si agisce su G;
(b) se $G \in \Sigma$ è una γ-formula e c è un simbolo di costante che compare in qualche formula di Σ allora l'istanza di G relativa a c appartiene a Σ.

Per raggiungere questi obiettivi:

(a) agiremo sempre prima sulle formule che vengono eliminate (doppie negazioni, α-, β- e δ-formule) che sulle γ-formule: questo ci assicura che nessuna doppia

negazione, α-, β- o δ-formula venga "dimenticata" e che prima o poi si arriverà anche ad agire sulle γ-formule;

(b) se $E(n)$ contiene qualche γ-formula affiancheremo a $E(n)$ una funzione C_n che assegna ad ogni γ-formula $G \in E(n)$ un insieme di simboli di costante $C_n(G)$: l'idea è che se $c \in C_n(G)$ l'istanza di G relativa a c deve essere ancora considerata. Quando agiamo su G (e in realtà agiremo su tutte le γ-formule di $E(n)$ simultaneamente) aggiungeremo all'etichetta del nodo che stiamo creando tutte le istanze di G relative a elementi di $C_n(G)$.

ALGORITMO 8.32. Un tableau sistematico per un enunciato F è un albero in cui ogni nodo è etichettato con un insieme finito di enunciati $E(n)$ e con una funzione C_n che associa ad ogni γ-formula di $E(n)$ (se ce ne sono) un insieme finito di simboli di costante. Il tableau è costruito per stadi $T_0, \ldots, T_i, \ldots$: per ogni i, T_{i+1} è un albero che estende T_i aggiungendo uno o due nodi con le rispettive etichette e lasciando invariate le etichette dei nodi già appartenenti a T_i. L'unione degli alberi T_i è il tableau per F.

All'inizio della costruzione T_0 consiste di un solo nodo (la *radice* dell'albero) etichettato con $\{F\}$ e, se F è una γ-formula, con la funzione che associa a F l'insieme dei simboli di costante che compaiono in F o, se non ce ne sono, l'insieme $\{c\}$ dove c è un simbolo di costante qualsiasi.

Allo stadio i diciamo che una foglia n dell'albero T_i è

una foglia finale: se $E(n)$ contiene una coppia complementare di letterali, oppure se $E(n)$ contiene solo letterali e γ-formule G tali che $C_n(G) = \emptyset$;

un γ-nodo: se non è una foglia finale ma $E(n)$ contiene solo letterali e γ-formule (questo implica che esiste qualche γ-formula $G \in E(n)$ per cui $C_n(G) \neq \emptyset$);

un nodo ordinario: se non è né una foglia finale né un γ-nodo (questo implica che $E(n)$ contiene qualche doppia negazione, α-, β- o δ-formula).

Cerchiamo una foglia n dell'albero T_i che sia un nodo ordinario. Se esiste fissiamo n e scegliamo un enunciato $G \in E(n)$ che sia una doppia negazione, oppure una α-, β- o δ-formula:

(1) se G è una doppia negazione con ridotto G_1 aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$; per ogni γ-formula $H \in E(n)$ poniamo $C_{n'}(H) = C_n(H)$; se G_1 è una γ-formula definiamo $C_{n'}(G_1)$ come l'insieme dei simboli di costante che compaiono in $E(n')$ o, se non ce ne sono, l'insieme $\{c\}$ dove c è un simbolo di costante qualsiasi;

(2) se G è una α-formula con ridotti G_1 e G_2 aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1, G_2\}$; per ogni γ-formula $H \in E(n)$ poniamo $C_{n'}(H) = C_n(H)$; se G_i ($i = 1$ o $i = 2$) è una γ-formula definiamo $C_{n'}(G_i)$ come l'insieme dei simboli di costante che compaiono in $E(n')$ o, se non ce ne sono, l'insieme $\{c\}$ dove c è un simbolo di costante qualsiasi;

(3) se G è una β-formula con ridotti G_1 e G_2 aggiungiamo due nodi tra loro inconfrontabili n_1 e n_2 sotto n e poniamo per $i = 1$ e $i = 2$, $E(n_i) = (E(n) \setminus \{G\}) \cup \{G_i\}$; per ogni γ-formula $H \in E(n)$ poniamo $C_{n_i}(H) = C_n(H)$; se G_i è una γ-formula definiamo $C_{n_i}(G_i)$ come l'insieme dei simboli di costante che compaiono in $E(n_i)$ o, se non ce ne sono, l'insieme $\{c\}$ dove c è un simbolo di costante qualsiasi;

(4) se G è una δ-formula sia a una costante che non compare in $E(n)$ e sia G_1 l'istanza di G relativa ad a: aggiungiamo un nodo n' sotto n e poniamo $E(n') = (E(n) \setminus \{G\}) \cup \{G_1\}$; per ogni γ-formula $H \in E(n)$ poniamo $C_{n'}(H) = C_n(H) \cup \{a\}$; se G_1 è una γ-formula definiamo $C_{n'}(G_1)$ come l'insieme dei simboli di costante che compaiono in $E(n')$ (che comprende almeno a ed è quindi non vuoto).

Se non esistono nodi ordinari cerchiamo un γ-nodo n. Sia Γ l'insieme di tutte le istanze di una γ-formula $G \in E(n)$ relative ad un simbolo di costante di $C_n(G)$ (notiamo che la definizione di γ-nodo implica che Γ non è vuoto). Aggiungiamo un nodo n' sotto n e poniamo $E(n') = E(n) \cup \Gamma$; per ogni γ-formula $H \in E(n)$ poniamo $C_{n'}(H) = \emptyset$; se $H \in \Gamma \setminus E(n)$ è una γ-formula definiamo $C_{n'}(H)$ come l'insieme dei simboli di costante che compaiono in $E(n')$ (esistono certamente quelli su cui sono state istanziate le γ-formule di $E(n)$).

Se infine non esistono né nodi ordinari né γ-nodi siamo al termine della costruzione e il tableau è completo.

ESEMPIO 8.33. Descriviamo un tableau sistematico per l'enunciato $\forall x \exists y\, r(x, y)$ (che indichiamo con F) che è all'origine del fatto che il tableau dell'esempio 8.12 è infinito.

F è una γ-formula e quindi alla radice n_1 del tableau possiamo porre $C_{n_1}(F) = \{a_1\}$. Dato che n_1 è un γ-nodo il nodo successivo m_1 è etichettato da F, $\exists y\, r(a_1, y)$ (in questo caso Γ ha un solo elemento) e abbiamo $C_{m_1}(F) = \emptyset$.

Ora abbiamo un nodo ordinario e agiamo sulla δ-formula $\exists y\, r(a_1, y)$: il nodo successivo n_2 è etichettato da F, $r(a_1, a_2)$ e abbiamo $C_{n_2}(F) = \{a_2\}$.

n_2 è nuovamente un γ-nodo: il nodo successivo m_2 è etichettato da F, $r(a_1, a_2)$, $\exists y\, r(a_2, y)$ (Γ ha nuovamente un solo elemento) e abbiamo $C_{m_2}(F) = \emptyset$.

m_2 è un nodo ordinario in cui si agisce su una δ-formula ed è seguito da un γ-nodo n_3, poi da un altro nodo ordinario m_4 e così via. Quando si agisce su m_i si introduce il simbolo di costante a_{i+1} e l'istanza di F relativa a a_{i+1} verrà introdotta agendo su n_{i+1} (e quindi nell'etichetta di m_{i+1}). Inoltre $C_{n_i}(F) = \{a_i\}$ e $C_{m_i}(F) = \emptyset$. Nelle etichette dei nodi si aggiungono uno dopo l'altro gli enunciati atomici della forma $r(a_i, a_{i+1})$.

Il tableau suggerisce un'interpretazione con dominio infinito che soddisfa F: $D^I = \mathbb{N}$, $r^I = \{\, (i, i+1) : i \in \mathbb{N} \,\}$. Notiamo però che F è soddisfatta anche da interpretazioni con dominio finito (addirittura da un'interpretazione con dominio di un solo elemento!) che il tableau non è riuscito a individuare.

La dimostrazione del teorema di correttezza 8.27 può essere adattata facilmente ai tableaux sistematici (l'unico caso in cui c'è una reale differenza è quello delle γ-formule, in cui si può comunque sfruttare il lemma 8.6) ottenendo che anche il metodo dei tableaux sistematici è corretto.

TEOREMA 8.34 (Teorema di correttezza per i tableaux sistematici). *Se T è un tableau sistematico chiuso per l'enunciato F allora F è insoddisfacibile.*

6. Cenni sul teorema di completezza

Il metodo dei tableaux sistematici è stato introdotto per ottenere la completezza, cioè per dimostrare che se un enunciato è insoddisfacibile allora il metodo dei tableaux riesce a scoprirlo.

TEOREMA 8.35 (Teorema di completezza). *Se l'enunciato F è insoddisfacibile allora ogni tableau sistematico per F è chiuso.*

SCHEMA DELLA DIMOSTRAZIONE. Seguiamo l'approccio utilizzato nel caso dei tableaux proposizionali (teorema 4.24), cioè proviamo che se qualche tableau sistematico per F è aperto allora F è soddisfacibile. Sia dunque T un tableau sistematico aperto per F e fissiamo un ramo aperto (finito o infinito) r di T. La dimostrazione si sviluppa in tre passi:

a. definizione di insieme di Hintikka (definizione 8.36);
b. dimostrazione che ogni insieme di Hintikka è soddisfacibile (lemma 8.40);
c. dimostrazione che $\bigcup_{n \in r} E(n)$ è un insieme di Hintikka (lemma 8.41).

Dato che $F \in \bigcup_{n \in r} E(n)$ (perché la radice del tableau appartiene a r) questi passi sono sufficienti a completare la dimostrazione.

Nel seguito diamo la definizione di insieme di Hintikka, ed enunciamo precisamente i lemmi che corrispondono al secondo e al terzo passo, ma ci limitiamo a dare solamente alcuni cenni sulle loro dimostrazioni. □

La definizione di insieme di Hintikka riprende quella del caso proposizionale (definizione 4.25): aggiungiamo le condizioni riguardanti γ- e δ-formule.

Definizione 8.36. Un insieme di enunciati Γ è un *insieme di Hintikka* se sono soddisfatte alle seguenti condizioni:

(1) Γ non contiene coppie complementari di letterali;
(2) se una doppia negazione appartiene a Γ allora il suo ridotto appartiene a Γ;
(3) se una α-formula appartiene a Γ allora entrambi i suoi ridotti appartengono a Γ;
(4) se una β-formula appartiene a Γ allora almeno uno dei suoi ridotti appartiene a Γ;
(5) se una γ-formula F appartiene a Γ allora almeno un simbolo di costante compare in Γ e tutte le istanze di F relative a simboli di costante che compaiono in Γ appartengono a Γ;
(6) se una δ-formula appartiene a Γ allora almeno una delle sue istanze appartiene a Γ.

Esempio 8.37. L'insieme

$$\{\forall x(\neg r(x,a) \to \exists y\, r(y,x)), \neg r(a,a) \to \exists y\, r(y,a), \exists y\, r(y,a), \\ r(b,a), \neg r(b,a) \to \exists y\, r(y,b), \neg\neg r(b,a)\}$$

è un insieme di Hintikka. Infatti non contiene coppie complementari di letterali, contiene il ridotto della doppia negazione $\neg\neg r(b,a)$, uno dei ridotti di ciascuna delle β-formule $\neg r(a,a) \to \exists y\, r(y,a)$ e $\neg r(b,a) \to \exists y\, r(y,b)$, le istanze relative a a e b della γ-formula $\forall x(\neg r(x,a) \to \exists y\, r(y,x))$ ed una istanza (quella relativa a b) della δ-formula $\exists y\, r(y,a)$.

Esercizio 8.38. Siano F_1 e F_2 gli enunciati $\forall x(\neg p(x) \wedge r(x,x))$ e $\exists x(p(x) \vee \neg r(x,x))$.

(i) Definite insiemi di Hintikka Γ_1 e Γ_2 con $F_i \in \Gamma_i$.
(ii) $\Gamma_1 \cup \Gamma_2$ è un insieme di Hintikka?
(iii) ($\star$) Esiste un insieme di Hintikka che contiene sia F_1 che F_2?

Esercizio 8.39. Verificare che l'insieme di enunciati che compaiono nelle etichette del ramo aperto del tableau ottenuto al termine dell'esempio 8.11 è un insieme di Hintikka.

Lemma 8.40 (Lemma di Hintikka). *Ogni insieme di Hintikka è soddisfacibile.*

Cenno di dimostrazione. Sia Γ un insieme di Hintikka e sia $\mathcal{C}$ l'insieme dei simboli di costante che compaiono in Γ. Non è difficile dimostrare che $\mathcal{C} \neq \emptyset$.

Definiamo un'interpretazione I per il linguaggio di Γ ponendo: $D^I = \mathcal{C}$, $c^I = c$ per ogni $c \in \mathcal{C}$, $p^I = \{\, (c_1, \ldots, c_n) \in \mathcal{C}^n : p(c_1, \ldots, c_n) \in \Gamma \,\}$ per ogni simbolo di relazione n-ario p che compare in Γ (ricordiamo che in questo capitolo stiamo supponendo che i linguaggi non contengano simboli di funzione).

Si dimostra poi, estendendo i ragionamenti svolti per dimostrare il lemma 4.29 (che è la versione proposizionale del presente lemma), che per ogni $F \in \Gamma$ si ha $I \models F$ e questo implica che Γ è soddisfacibile. □

LEMMA 8.41. *Se r è un ramo aperto (finito o infinito) di un tableau sistematico allora $\Gamma = \bigcup_{n \in r} E(n)$ è un insieme di Hintikka.*

CENNO DI DIMOSTRAZIONE. Bisogna verificare che Γ soddisfa le sei proprietà della definizione di insieme di Hintikka. La dimostrazione è anche in questo caso una generalizzazione di quella del caso proposizionale (lemma 4.30). Ci sono però diverse difficolta in più, principalmente dovute al fatto che r e Γ possono essere infiniti.

Preliminarmente è necessario dimostrare che si agisce su ogni formula di Γ che non è un letterale. Inoltre il caso delle γ-formule richiede particolare attenzione: bisogna infatti verificare che **tutte** le loro istanze relative a simboli di costante che compaiono in Γ appartengono a Γ. □

La dimostrazione del teorema di completezza accennata sopra fornisce informazioni su come trovare un'interpretazione che soddisfi un enunciato il cui tableau è aperto.

LEMMA 8.42. *Sia r un ramo di un tableau sistematico per l'enunciato F che è aperto. Siano $\Gamma = \bigcup_{n \in r} E(n)$ e $\mathcal{C}$ l'insieme dei simboli di costante che compaiono nelle etichette dei nodi di r.*

Sia I l'interpretazione definita da: $D^I = \mathcal{C}$, $c^I = c$ per ogni $c \in \mathcal{C}$ che compare in F, $p^I = \{ (c_1, \ldots, c_n) \in \mathcal{C}^n : p(c_1, \ldots, c_n) \in \Gamma \}$ per ogni simbolo di relazione n-ario p che compare in F. Allora $I \models F$.

Un esempio di applicazione di questo lemma è contenuto nell'esempio 8.33. Anche l'esempio 8.11 contiene un esempio di questo tipo: in questo caso il tableau ottenuto al termine di quell'esempio non soddisfa esattamente alla nostra definizione di tableau sistematico (perché le istanze di H relative ad a e b non vengono introdotte simultaneamente), ma è stato costruito senza dimenticare nessuna formula e soddisfa al teorema di completezza.

NOTA 8.43. I teoremi di correttezza e completezza mostrano che il metodo dei tableaux sistematici è una procedura di semidecisione per l'insoddisfacibilità (e quindi per il lemma 6.55 anche per la validità) degli enunciati della logica predicativa. Questo significa che se l'enunciato F è insoddisfacibile l'algoritmo 8.32 si fermerà fornendoci la risposta corretta; se invece F è soddisfacibile l'algoritmo può arrestarsi (e dare la risposta corretta) oppure proseguire la sua esecuzione all'infinito.

NOTA 8.44. Nel 1936 Alonzo Church, basandosi sul lavoro precedente di Alan Turing, ha dimostrato la non esistenza di una procedura di decisione per l'insoddisfacibilità degli enunciati predicativi: una procedura con le caratteristiche di quella dei tableaux è quanto di meglio si possa ottenere. Vediamo dunque che il caso predicativo è molto diverso dal caso proposizionale, in cui abbiamo introdotto due procedure di decisione per l'insoddisfacibilità (le tavole di verità ed il metodo dei tableaux).

NOTA 8.45. Come già osservato nel caso proposizionale (nota 4.17), anche nel caso predicativo l'algoritmo di costruzione dei tableaux **non** prevede la sostituzione di una formula con un'altra logicamente equivalente ad essa. Se introducessimo questa possibilità, il teorema di completezza non sarebbe più vero, perché un enunciato insoddisfacibile potrebbe avere tableaux con rami infiniti, e quindi aperti.

7. I tableaux e la conseguenza logica

Questa sezione ripete quasi parola per parola l'analoga sezione del capitolo 4.

Abbiamo presentato il metodo dei tableaux come un metodo per studiare la validità o la soddisfacibilità di un singolo enunciato. Usando l'esercizio 6.56 (in particolare la parte riguardante il lemma 2.43) possiamo usare i tableaux per studiare la validità o la soddisfacibilità di un insieme finito di enunciati $\{F_1, \dots, F_n\}$. Come nel caso proposizionale si ottiene il seguente algoritmo.

ALGORITMO 8.46. Per stabilire se un insieme finito $\Gamma = \{F_1, \dots, F_n\}$ di enunciati è soddisfacibile costruiamo un tableau la cui radice è etichettata con Γ. Se il tableau è aperto Γ è soddisfacibile (e un ramo aperto ci permette di definire un'interpretazione che soddisfa Γ), se il tableau è chiuso Γ è insoddisfacibile.

Similmente la parte dell'esercizio 6.56 riguardante il lemma 2.38 ci permette di usare i tableaux per stabilire la sussistenza della relazione di conseguenza logica. Come nel caso proposizionale si ottiene il seguente algoritmo.

ALGORITMO 8.47. Per stabilire se $F_1, \dots, F_n \models G$ costruiamo un tableau la cui radice è etichettata con $\{F_1, \dots, F_n, \neg G\}$. Se il tableau è chiuso $F_1, \dots, F_n \models G$, se il tableau è aperto $F_1, \dots, F_n \not\models G$ (e un ramo aperto ci permette di definire un'interpretazione che soddisfa $F_1, \dots, F_n$ ma non G).

ESERCIZIO 8.48. Studiate con il metodo del tableaux le conseguenze logiche dei lemmi 6.39, 6.41, 6.43 e 6.46, degli esempi 6.27 e 6.45 e degli esercizi 6.28 e 6.48 (verificate che il risultato ottenuto con i tableaux coincida con quello ottenuto in precedenza). Per i problemi che riguardano generiche F e G utilizzate enunciati atomici specifici. Quando avete a che fare con un'equivalenza logica usate l'esercizio 6.31.

APPENDICE A

La dimostrazione del Lemma di Sostituzione

Questa appendice è dedicata alla dimostrazione del Lemma di Sostituzione (lemma A.2 qui sotto, e lemma 6.37 nel testo). Questa dimostrazione non fa parte del programma d'esame ma è inclusa qui per completezza delle dispense, sperando nell'interesse di qualche studente particolarmente curioso.

Iniziamo, come al solito, a considerare il caso dei termini.

LEMMA A.1. *Siano σ uno stato di un'interpretazione I, x una variabile e t e s due termini, con t chiuso. Allora*

$$\sigma(s\{x/t\}) = \sigma[x/t^I](s).$$

DIMOSTRAZIONE. Per induzione sulla complessità di s (il lettore tenga presente la definizione 5.15).

Se s è un simbolo di costante c, si ha $\sigma(c\{x/t\}) = \sigma(c) = c^I = \sigma[x/t^I](c)$.

Se s è una variabile y diversa da x si ha $\sigma(y\{x/t\}) = \sigma(y) = \sigma[x/t^I](y)$.

Se s è x si ha $\sigma(x\{x/t\}) = t^I = \sigma[x/t^I](x)$.

Se s è $f(s_1, \ldots, s_n)$ allora

$$\begin{aligned}
\sigma(s\{x/t\}) &= \sigma(f(s_1, \ldots, s_n)\{x/t\}) \\
&= \sigma(f(s_1\{x/t\}, \ldots, s_n\{x/t\})) \\
&= f^I(\sigma(s_1\{x/t\}), \ldots, (\sigma(s_n\{x/t\}))) \\
&= f^I(\sigma[x/t^I](s_1), \ldots, \sigma[x/t^I](s_n)) \\
&= \sigma[x/t^I](f(s_1, \ldots, s_n)) \\
&= \sigma[x/t^I](s),
\end{aligned}$$

dove nel passaggio dalla terza alla quarta riga si è usata l'ipotesi induttiva. □

Passiamo ora al caso delle formule.

LEMMA A.2. *Siano σ uno stato di un'interpretazione I, x una variabile, t un termine chiuso e F una formula. Allora*

$$I, \sigma \models F\{x/t\} \quad \textit{se e solo se} \quad I, \sigma[x/t^I] \models F.$$

DIMOSTRAZIONE. Per induzione sulla complessità di F (qui bisogna ricordare le definizioni 5.43 e 5.45).

Se F è la formula atomica $p(t_1, \ldots, t_n)$ allora

$$\begin{aligned}
I, \sigma \models F\{x/t\} &\text{ se e solo se } I, \sigma \models p(t_1, \ldots, t_n)\{x/t\} \\
&\text{se e solo se } I, \sigma \models p(t_1\{x/t\}, \ldots, t_n\{x/t\}) \\
&\text{se e solo se } (\sigma(t_1\{x/t\}), \ldots, \sigma(t_n\{x/t\})) \in p^I \\
&\text{se e solo se } (\sigma[x/t^I](t_1), \ldots, \sigma[x/t^I](t_n)) \in p^I \\
&\text{se e solo se } I, \sigma[x/t^I] \models p(t_1, \ldots, t_n) \\
&\text{se e solo se } I, \sigma[x/t^I] \models F,
\end{aligned}$$

dove nel passaggio dalla terza alla quarta riga abbiamo utilizzato il lemma A.1.

Se F è $G \vee H$ si ha

$$\begin{aligned} I, \sigma \models F\{x/t\} &\text{ se e solo se } I, \sigma \models (G \vee H)\{x/t\} \\ &\text{se e solo se } I, \sigma \models (G\{x/t\} \vee H\{x/t\}) \\ &\text{se e solo se } I, \sigma \models G\{x/t\} \text{ oppure } I, \sigma \models H\{x/t\} \\ &\text{se e solo se } I, \sigma[x/t^I] \models G \text{ oppure } I, \sigma[x/t^I] \models H \\ &\text{se e solo se } I, \sigma[x/t^I] \models G \vee H \\ &\text{se e solo se } I, \sigma[x/t^I] \models F, \end{aligned}$$

dove nel passaggio dalla terza alla quarta riga abbiamo utilizzato l'ipotesi induttiva.

In modo del tutto analogo si tratta il caso di negazioni, congiunzioni e implicazioni.

Se F è $\forall y\, G$ dobbiamo distinguere due casi.

(i) Se x non è libero in F (in particolare se y è x) si ha che $F\{x/t\}$ è F e che $I, \sigma \models F$ è equivalente a $I, \sigma[x/t^I] \models F$ per il lemma 6.10.

(ii) Se x è libero in F si ha che y è una variabile diversa da x. Allora

$$\begin{aligned} I, \sigma \models \forall y\, G\{x/t\} &\text{ se e solo se per ogni } d \in D^I,\ I, \sigma[y/d] \models G\{x/t\} \\ &\text{se e solo se per ogni } d \in D^I,\ I, \sigma[y/d][x/t^I] \models G \\ &\text{se e solo se per ogni } d \in D^I,\ I, \sigma[x/t^I][y/d] \models G \\ &\text{se e solo se } I, \sigma[x/t^I] \models \forall y\, G, \end{aligned}$$

dove nel passaggio dalla prima alla seconda riga abbiamo utilizzato l'ipotesi induttiva, in quello dalla seconda alla terza riga il fatto che x e y sono variabili diverse (e quindi $\sigma[y/d][x/t^I]$ e $\sigma[x/t^I][y/d]$ sono lo stesso stato).

In modo del tutto analogo si tratta il caso della quantificazione esistenziale. □

Esercizio A.3. Fate i passi induttivi che sono stati omessi nella dimostrazione del lemma A.2.

APPENDICE B

Ulteriori esercizi

In questo capitolo sono raccolti, senza un ordine particolare, alcuni esercizi che possono essere utili nella preparazione dell'esame. Ulteriori esercizi si possono ricavare dai testi degli esami disponibili sulla pagina web del corso, dove è anche possibile trovare le loro soluzioni.

Esercizio B.1. Siano F e G enunciati e H e K formule con una sola variabile libera x di un linguaggio $\mathcal{L}$. Sia I un'interpretazione per $\mathcal{L}$. Alcune delle seguenti affermazioni sono corrette, altre no. Dimostrate le prime e trovate un controesempio alle seconde.

(a) Se $I \models H\{x/c\}$ per un simbolo di costante c di $\mathcal{L}$ allora $I \models \exists x\, H$.
(b) Se $I \models \exists x\, H$ allora esiste un simbolo di costante c di $\mathcal{L}$ tale che $I \models H\{x/c\}$.
(c) Se $\exists x\, H$ è valida allora esiste un termine chiuso t di $\mathcal{L}$ tale che $H\{x/t\}$ è valida.
(d) Se $I \models F \vee G$, allora $I \models F$ oppure $I \models G$.
(e) ($\star$) Se $I \models H \vee K$, allora $I \models H$ oppure $I \models K$.

Esercizio B.2. Sia Γ l'insieme delle formule ottenute nell'esercizio 7.15. Stabilite quali tra $c(d)$, $t(d)$ e $\neg c(d)$ è conseguenza logica di Γ. In ogni caso giustificate le vostre risposte, fornendo se necessario interpretazioni con le proprietà opportune.

Nel linguaggio dell'esercizio 7.15 traducete anche "le amiche di Carla che non amano né il teatro né il cinema non sono amiche di Barbara".

Esercizio B.3. Considerate il linguaggio $\{b, c, d, s, m, a\}$, dove b, c, d sono simboli di costante che rappresentano Bruna, Carla e Davide, s, m sono simboli di relazione unari ($s(x)$ sta per "a x piace la spiaggia", $m(x)$ sta per "a x piace la montagna"), mentre a è un simbolo di relazione binario ($a(x, y)$ sta per "x è amico di y"). Formalizzare le seguenti frasi:

(a) Ad ogni amico di Davide piace la spiaggia o la montagna.
(b) A nessun amico di Bruna piace la spiaggia.
(c) Carla è amica sia di Bruna che di Davide.
(d) A qualche amico di Davide piace la montagna.

Se F, G, H e K sono le rispettive formalizzazioni, dimostrate usando il metodo dei tableaux che $F, G, H \models K$.

Esercizio B.4. Considerate le frasi seguenti:

(a) se un insegnante è soddisfatto, almeno un suo alunno ha passato l'esame;
(b) ogni insegnante è soddisfatto se almeno un suo alunno ha passato l'esame.

(i) Trovate un traduzione F di (a) ed una traduzione G di (b) nel linguaggio $\{i, a, p, s\}$, dove $i(x)$ sta per x *è un insegnante*, $a(x, y)$ sta per y *è alunno di* x, $p(x)$ sta per x *ha passato l'esame*, $s(x)$ sta per x *è soddisfatto*;
(ii) stabilite se $F \models G$ e se $G \models F$.

Esercizio B.5. Dimostrate che l'enunciato

$$\forall x(q(x) \to \exists y\, r(x, y)) \wedge \forall x(\neg q(x) \to \exists y\, r(x, y)) \wedge \exists x\, \forall y\, \neg r(x, y)$$

è insoddisfacibile. Utilizzate sia la nozione di soddisfazione che il metodo dei tableaux.

Esercizio B.6. Considerate il linguaggio $\{r, s, i, u, \ell, c\}$, dove r, s, i, u, ℓ sono simboli di relazione unari ($r(x)$ sta per "x è un ragazzo", $s(x)$ sta per "x è una ragazza", $i(x)$ sta per "x è iscritto all'università", $u(x)$ sta per "x sa usare un computer", $\ell(x)$ sta per "x troverà lavoro"), mentre c è un simbolo di relazione binario ($c(x, y)$ sta per "x conosce y"). Formalizzate le seguenti frasi:

(a) Tutti i ragazzi e le ragazze iscritti all'università sanno usare il computer o conoscono qualcuno che lo sa usare.
(b) Fra quelli che sanno usare un computer c'è qualcuno che troverà lavoro.
(c) Tutti conoscono qualcuno che troverà lavoro.

Esercizio B.7. Dimostrate con le tavole di verità e con il metodo dei tableaux la validità delle seguenti formule proposizionali

$$(p \to q) \land (p \to (q \to r)) \to (p \to r);$$
$$(p \lor q) \land (q \to r) \to (\neg r \to p).$$

Esercizio B.8. Stabilite se $p \to q \land r \models (p \to q) \to r$.

Esercizio B.9. Utilizzando il metodo dei tableaux, dimostrate che l'enunciato seguente è valida:

$$\exists x(\neg q(x) \to r(x)) \land \forall x(q(x) \to r(a)) \to \exists x\, r(x).$$

Esercizio B.10. Stabilite con le tavole di verità e con il metodo dei tableaux se le seguenti equivalenze logiche sono corrette:

$$(p \to q) \to r \equiv p \land q \to r;$$
$$p \to (q \to r) \equiv p \land q \to r.$$

Esercizio B.11. Dimostrate sia direttamente che con i tableaux che

$$\forall x(\exists y\, r(x, y) \to \forall y\, \neg r(y, x)) \models \neg\exists x\, r(x, x).$$

Esercizio B.12. Utilizzando il metodo dei tableaux, dimostrate che l'enunciato

$$\forall x\, \exists y\, r(x, y) \land \exists x\, \forall y\, \neg r(x, y)$$

è insoddisfacibile.

Esercizio B.13. Se p, q, r, s, t e u sono lettere proposizionali, mettete in forma normale congiuntiva e in forma normale disgiuntiva usando gli algoritmi di Fitting la formula

$$(p \land \neg q) \lor r \to \neg(\neg s \to t \land \neg u).$$

Esercizio B.14. Formalizzate in un opportuno linguaggio le seguenti frasi:

(a) Tutti i filosofi ed i matematici sono distratti.
(b) C'è qualcuno che non è distratto.
(c) Qualcuno non è né filosofo né matematico.

Se F, G e H sono le rispettive formalizzazioni, dimostrate sia direttamente che con i tableaux che $F, G \models H$.

Esercizio B.15. Dimostrate, sia direttamente che attraverso i tableaux, che

$$\exists x\, r(x) \lor \exists x\, \neg s(x) \models \exists x(s(x) \to r(x)).$$

Esercizio B.16. Siano p, q e r tre lettere proposizionali. Usate il metodo dei tableaux per mostrare che

$$\neg p \to q, q \to r, \neg r \lor p \models p.$$

Esercizio B.17. L'enunciato $\forall y\, \exists x(p(x, y) \to p(y, y))$ è valido? Se sì, dimostratelo, se no definite un'interpretazione in cui esso non è soddisfatto.

Esercizio B.18. Dimostrate sia direttamente che con i tableaux la validità dei seguenti enunciati:

$$\exists x(\neg p(x) \wedge \exists y(q(x) \to p(y))) \wedge \forall x(\neg q(x) \to p(x)) \to \exists x\, p(x);$$
$$\forall x\, \forall y(p(x,y) \to q(x)) \wedge \neg q(a) \to \forall y\, \neg p(a,y);$$
$$\forall x(p(x) \vee q(x) \to r(x)) \wedge \exists x\, \neg r(x) \to \exists x(\neg p(x) \wedge \neg q(x)).$$

Esercizio B.19. Utilizzate gli algoritmi di Fitting per trasformare in forma normale congiuntiva e in forma normale disgiuntiva la formula

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

Esercizio B.20. Verificate la validità di

$$\exists x\, p(x) \wedge \forall x(p(x) \to \forall y\, q(x,y)) \to \exists x\, q(x,x).$$

Esercizio B.21. Dimostrate con il metodo dei tableaux la validità di

$$\forall x\, \exists y(p(y) \to p(x)).$$

Esercizio B.22. Usando il linguaggio $\{m, c, l, b, a\}$ dove m è un simbolo di funzione unario ($m(x)$ rappresenta "il miglior amico di x") c e l sono costanti (che denotano rispettivamente "Carlo" e "Luca"), b e a sono simboli predicativi binari ($b(x,y)$ sta per "x è più basso di y" e $a(x,y)$ sta per "x è amico di y"), formalizzare le frasi seguenti:

(a) gli amici di Carlo sono amici di Luca;
(b) ogni amico di Carlo è più basso di qualche amico Luca;
(c) ogni amico di Carlo è più basso del miglior amico di Luca;
(d) qualche amico di Luca è più basso di Carlo, ma non di tutti gli amici di Carlo;
(e) un amico di Carlo è più basso di Luca soltanto se è più basso del miglior amico di Carlo;
(f) se Carlo è più basso del miglior amico di Luca, allora almeno un amico di Carlo è più basso di tutti i migliori amici degli amici di Luca.

Esercizio B.23. Dimostrate sia direttamente che con i tableaux la validità degli enunciati:

$$p(a) \to \exists x\, p(x); \quad \exists x\, \forall y\, r(x,y) \to \forall y\, \exists x\, r(x,y);$$
$$\neg\exists x\, \forall y\, ((r(x,y) \to \neg r(y,y)) \wedge (\neg r(y,y) \to r(x,y)));$$
$$\forall x\, (\forall y(r(x,y) \to p(y)) \wedge \exists z\, \neg p(z) \to \exists z\, \neg r(x,z))\,.$$

Esercizio B.24. Usando il metodo dei tableaux dimostrate che

$$\forall x\, \exists y\, r(x,y), \exists x\, \forall y\, q(y,x) \models \exists x\, \exists y(r(x,y) \wedge q(y,x)).$$

Esercizio B.25. Tra le seguenti affermazioni riguardanti le formule proposizionali F e G alcune sono corrette, altre no. Dimostrate le prime e trovate un controesempio alle seconde.

(a) Se F e G sono soddisfacibili allora esiste un'interpretazione che soddisfa sia F che G.
(b) $G \not\models \neg F$ se e solo $F \wedge G$ è soddisfacibile.
(c) $F \models \neg G$ se e solo se $F \not\models G$.
(d) Se F non è valido allora $\neg F$ è valido.
(e) Se $v(F \to G) = \mathbf{V}$ e $v(F) = \mathbf{F}$, allora $v(G) = \mathbf{F}$.
(f) Se $F \vee G$ è valida allora F è valida oppure G è valida.

ESERCIZIO B.26. Trovate un'interpretazione che soddisfa la congiunzione di:

$$\exists u\, \forall x (p(u,x) \to q(x));$$
$$\forall x\, \exists v (p(x,v) \vee p(v,x));$$
$$\forall x\, \forall y (q(x) \wedge q(y) \wedge \forall z (p(x,z) \to p(y,z)) \to \exists z (p(y,z) \wedge \neg p(x,z))).$$

ESERCIZIO B.27. $F \vee (\neg G \wedge H)$ è logicamente equivalente a $G \vee \neg H \to F$?

ESERCIZIO B.28. Dimostrate sia direttamente che con il metodo dei tableaux la validità di

$$\exists x p(x) \to \neg \forall x (p(x) \to \neg p(x)).$$

ESERCIZIO B.29. Sia $\{s, r, i, c, p\}$ un linguaggio dove s, r e i sono simboli di relazione unari, c è un simbolo di relazione binario e p è un simbolo di funzione unario. Interpretando $s(x)$ come "x è uno scrittore", $r(x)$ come "x è ricco", $i(x)$ come "x è interessante", $c(x,y)$ come "x conosce y", $p(x)$ come "il primo libro di x", traducete le seguenti frasi:

(i) il primo libro di qualche scrittore non è interessante;
(ii) ogni scrittore è ricco oppure conosce uno scrittore ricco;
(iii) se uno scrittore non è ricco allora il suo primo libro è interessante.

ESERCIZIO B.30. Usando l'algoritmo di Fitting, mettere in forma normale disgiuntiva la formula

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

ESERCIZIO B.31. Supponiamo che F, G, H, K siano formule (proposizionali o predicative, il risultato non cambia) tali che $F, H \models G$ e $H \models G \to K$. Quali delle seguenti conclusioni sono corrette?

(a) $F \to H \models F \to G$;
(b) $F \to H \models K$;
(c) $\neg F \vee H \models \neg G \vee K$.

ESERCIZIO B.32. Sia $\{d, m, p, c, g\}$ un linguaggio dove d ed m sono simboli di funzione unari, p e c sono simboli di relazione binari, e g è un simbolo di costante. Interpretando $d(x)$ come "il dentista di x", $m(x)$ come "il miglior amico di x", $p(x,y)$ come "x è parente di y", $c(x,y)$ come "x cura y", e g come "Gianni" traducete le seguenti frasi:

(i) il dentista di Gianni cura qualcuno che non è parente di Gianni;
(ii) i parenti di Gianni non curati dal dentista di Gianni sono curati dal dentista del miglior amico di Gianni;
(iii) i dentisti che curano il proprio miglior amico non sono suoi parenti;
(iv) nessun dentista cura il suo miglior amico.

ESERCIZIO B.33. Dimostrate con il metodo dei tableaux che

$$\forall x\, \exists y\, r(x,y), \exists x\, \forall y\, q(y,x) \models \exists x\, \exists y (r(x,y) \wedge q(y,x)).$$

ESERCIZIO B.34. Sia $\{a, m, i, p, c\}$ un linguaggio dove a è un simbolo di costante, m un simbolo di funzione unario, i e p simboli di relazione unari e c un simbolo di relazione binario. Interpretando a come "Andrea", $m(x)$ come "il miglior amico di x", $i(x)$ come "x è invitato alla festa", $p(x)$ come "x è parente di Andrea", $c(x,y)$ come "x conosce y", traducete le seguenti frasi:

(i) tutti gli invitati alla festa conoscono almeno un parente di Andrea;
(ii) il miglior amico di Andrea conosce tutti i parenti di Andrea ma non è parente di Andrea;
(iii) se il miglior amico di un parente di Andrea è invitato alla festa, lo è anche il miglior amico di Andrea.