Ingegneria della conoscenza

Introduzione alla logica predicativa del primo ordine

Marco Colombetti
Dipartimento di elettronica e informazione
Politecnico di Milano

2 aprile 2007

1. Che cos'è la logica

Queste note costituiscono un'introduzione alla *logica moderna*[1]; in questo senso del termine, la logica è una disciplina relativamente recente, le cui origini si possono far risalire ai lavori di George Boole (1854) e di Gottlob Frege (1879). Nella storia del pensiero occidentale, tuttavia, la tradizione della logica si può far risalire alla Grecia classica: l'obiettivo di questa disciplina, per i filosofi greci, era stabilire sotto quali condizioni si possa dire che un'argomentazione è valida[2]. In questo paragrafo prenderemo quindi le mosse dal concetto di argomentazione per introdurre gli elementi fondamentali della logica.

1.1 Argomentazioni e forma canonica

Un'*argomentazione*[3] è un discorso che ha la funzione di sostenere la verità di un enunciato, sulla base di un insieme di ragioni espresse da altri enunciati; l'enunciato di cui si vuole sostenere la verità è detto *conclusione*, mentre gli enunciati che esprimono le ragioni a favore della verità della conclusione sono detti *premesse*. Ovvi esempi di argomentazioni sono le dimostrazioni di teoremi matematici (dove le premesse sono le ipotesi del teorema e la conclusione è la tesi), le arringhe degli avvocati in tribunale (dove le premesse sono i fatti accertati nel dibattimento e la conclusione è il giudizio d'innocenza o di colpevolezza dell'imputato), i discorsi politici e così via[4].

---

[1] Per un'introduzione alla logica moderna semplice ma più ampia di quella fornita in queste note si veda ad es. Bencivenga (1984). Per un'introduzione alla storia della logica si vedano invece Mangione e Bozzi (1993).

[2] La logica è spesso definita come la disciplina che studia le condizioni di correttezza del *ragionamento*. In realtà la logica non studia il ragionamento in quanto processo mentale, bensì le rappresentazioni linguistiche del ragionamento, come ad esempio le argomentazioni. È anche frequente definire la logica come la disciplina che studia l'*inferenza deduttiva* o, più semplicemente, la *deduzione*. Anche in questo caso va sottolineato che la logica non studia la deduzione in quanto processo mentale, ma si occupa piuttosto di testi linguistici che rappresentano deduzioni (come ad esempio le dimostrazioni di teoremi matematici). Il punto fondamentale è che l'oggetto della logica è comunque costituito da entità linguistiche (frasi e testi).

[3] Per essere più precisi bisognerebbe distinguere fra *argomentazione* (intesa come attività umana), *intervento argomentativo* (inteso come caso specifico di attività argomentativa) e *argomento* (inteso come testo linguistico la cui enunciazione costituisce un intervento argomentativo). Nel seguito, per semplicità, useremo il termine 'argomentazione' per denotare anche gli interventi argomentativi e gli argomenti.

[4] Le argomentazioni non sono l'unico tipo di testo articolato in premesse e conclusioni. Le *spiegazioni*, ad esempio, presentano la stessa struttura: quando diciamo che

Alberto è ricco perché ha vinto il primo premio alla lotteria

l'enunciato 1.2 continua a fungere da conclusione, e l'enunciato

Alberto ha vinto il primo premio alla lotteria

funge da premessa. Una differenza fondamentale fra un'argomentazione e una spiegazione sta nella *funzione comunicativa*: mentre la funzione di un'argomentazione è di portare ragioni a favore della verità di un enunciato controverso, la funzione di una spiegazione è di descrivere le origini di uno stato di cose il cui sussistere non è controverso.

Le argomentazioni permeano anche il discorso quotidiano. Ad esempio, supponiamo che qualcuno ci dica:

(1.1) *Alberto è ricco: va in giro in Ferrari!*

Enunciando questa frase il parlante intende sostenere la verità dell'enunciato

(1.2) *Alberto è ricco*

portando come ragione la verità dell'enunciato

(1.3) *Alberto va in giro in Ferrari*.

Quindi siamo in presenza di un'argomentazione, con l'enunciato 1.2 come conclusione e l'enunciato 1.3 come premessa.

È facile rendersi conto che la verità dell'enunciato 1.3 non basta ad assicurare la verità dell'enunciato 1.2; c'è infatti, nel discorso 1.1, un chiaro elemento sottinteso, che potremmo rappresentare con l'ulteriore enunciato

(1.4) *Tutti coloro che vanno in giro in Ferrari sono ricchi*.

Nell'argomentazione che stiamo analizzando l'enunciato 1.4 compare come *premessa nascosta*. La presenza di premesse nascoste è tipica delle argomentazioni nella loro più comune forma quotidiana, che chiameremo *forma naturale*. In certi casi può essere nascosta persino la conclusione. Ad esempio, alla domanda *Che cos'ha oggi Alberto?* si può rispondere *Tutti hanno paura di andare dal dentista*, che è ovviamente una premessa; in questo caso l'argomentazione completa comprende presumibilmente un'ulteriore premessa nascosta, *Alberto va dal dentista*, e la conclusione, pure nascosta, *Alberto ha paura*.

Per analizzare un'argomentazione in forma naturale è necessario innanzi tutto trasformarla in un'opportuna *forma canonica*, in cui

- vengono esplicitati gli enunciati nascosti, siano essi premesse o la conclusione;
- le premesse vengono scritte per prime e la conclusione per ultima (in genere la conclusione viene marcata da un simbolo convenzionale, '$\therefore$', che si legge *quindi*).

L'argomentazione 1.1 sarà quindi riscritta nel modo seguente:

(1.5) *Alberto va in giro in Ferrari*

Tutti coloro che vanno in giro in Ferrari sono ricchi

$\therefore$ *Alberto è ricco*

L'inserimento degli enunciati nascosti è un punto critico. In genere, un parlante lascia sottintesi gli enunciati ovvi, che fanno parte del cosiddetto *sfondo comune* (ovvero le cose che tutti sanno, per lo meno in un determinato ambiente culturale) o che comunque appaiono evidentemente veri nella situazione in cui ci si trova. Tuttavia, lasciare qualcosa sottinteso è sempre un rischio, perché gli enunciati nascosti possono essere ricostruiti dall'ascoltatore in modo non conforme alle intenzioni del parlante; inoltre, la presenza di enunciati sottintesi apre la porta alle manipolazioni, in quanto è più facile che una premessa di dubbia verità venga accettata se non è presentata esplicitamente. Di questo, tuttavia, la logica non si occupa: l'analisi logica vera e propria inizia dall'argomentazione posta in forma canonica.

1.2 Validità di un'argomentazione

Come dicevamo, la logica si pone l'obiettivo di stabilire se un'argomentazione è *valida*. Intuitivamente, ciò significa che la verità della conclusione discende dalla verità delle premesse o, in altre parole, che *la conclusione è certamente vera nell'ipotesi che siano vere le premesse*. In questo senso possiamo dire che l'argomentazione 1.5 è valida: infatti se gli enunciati 1.3 e 1.4 sono veri non è possibile che l'enunciato 1.2 sia falso.

È importante rendersi conto che stabilire la verità di un enunciato particolare non è compito della logica: ad esempio, la logica non ci può dire se l'enunciato *Alberto va in giro in Ferrari* sia vero o

falso. Per stabilire se un enunciato particolare è vero o falso dobbiamo ricorrere ai metodi dell'indagine empirica: ad esempio, potremmo aver visto molte volte Alberto alla guida di una Ferrari. Se l'enunciato appartiene all'ambito di una disciplina scientifica potremo utilizzare le procedure d'indagine previste dalla disciplina; se invece l'enunciato concerne la vita quotidiana utilizzaremo i mezzi più disparati, come l'osservazione personale o il "sentito dire". Qualunque mezzo si usi, comunque, valutare la verità di un singolo enunciato non spetta alla logica; ciò che invece spetta alla logica, come vedremo, è stabilire *sotto quali condizioni la verità di un enunciato dipenda dalla verità di altri enunciati*.

1.3 Forma logica

Ciò che caratterizza la logica moderna è l'utilizzo di metodi formali; per questa ragione la logica moderna è anche detta *logica formale* o *logica simbolica*; è anche detta *logica matematica*, e questo per due motivi: perché utilizza un approccio di tipo matematico, ma anche perché uno dei suoi obiettivi è chiarire alcuni problemi concernenti i fondamenti della matematica. Insomma, la logica moderna è per così dire doppiamente matematica: nei metodi e negli obiettivi.

Ritorniamo ora all'argomentazione 1.5. Possiamo distinguere fra il *contenuto* dell'argomentazione (che concerne una specifica persona, Alberto, e certi specifici fatti, come l'andare in giro in Ferrari e l'essere ricco) e la sua *forma*; come vedremo è proprio la forma, o per essere più precisi la *forma logica*, ad essere critica per la validità dell'argomentazione. Per capire che cosa sia la forma logica di un'argomentazione cominciamo ad analizzare l'enunciato 1.2. La frase *Alberto è ricco* può essere analizzata utilizzando i concetti di *soggetto* e di *predicato*. Il soggetto (*Alberto*) è un nome proprio, la cui funzione è *fare riferimento a uno specifico individuo*; a questo scopo non si utilizzano soltanto nomi propri, ma anche *descrizioni definite*, che in generale prendono la forma di sintagmi nominali di complessità arbitraria, come ad esempio

il Papa,

il Rettore del Politecnico di Milano,

l'uomo che ieri ha scavato una buca nel giardino dietro alla casa di tua cognata,

e così via. Il predicato (*è ricco*) è un sintagma verbale, la cui funzione è *rappresentare una proprietà* di un individuo. Si esprime poi una *proposizione* applicando un predicato a un soggetto: questa è appunto la funzione dell'enunciato completo *Alberto è ricco*; la proposizione espressa da questo enunciato dice che: l'individuo cui si fa riferimento con il nome *Alberto* gode della proprietà rappresentata dal sintagma verbale *è ricco*.

Il primo passo verso l'analisi formale dell'argomentazione consiste nel sostituire dei simboli al soggetto e al predicato: per il soggetto utilizzeremo un simbolo, *a*, detto *costante individuale* o più semplicemente *costante*; per il predicato utilizzeremo invece un simbolo, *R*, detto *costante predicativa* o più semplicemente *predicato*; infine per rappresentare il fatto che il predicato *R* è applicato alla costante *a*, scriveremo *R*(*a*) o più semplicemente, tralasciando le parentesi,

Ra.

Questa *formula* rappresenta quindi la forma logica della frase *Alberto è ricco*. Analogamente, la formula *Fa* rappresenterà la forma logica della frase *Alberto va in giro in Ferrari*, con la convenzione che la costante predicativa *F* rappresenti il predicato *va in giro in Ferrari*.

L'analisi dell'enunciato 1.4 è un po' più complessa. Innanzi tutto parafrasiamo l'enunciato come

(1.4') *Per ogni individuo x: se x va in giro in Ferrari, allora x è ricco*.

La forma logica dell'enunciato 1.4' è rappresentata allora dalla formula

$\forall x\,(Fx \rightarrow Rx)$,

dove il simbolo '$\forall x$' significa *per ogni individuo x*, e il simbolo '$\rightarrow$' significa *se ..., allora ...*.

Ora che abbiamo stabilito la forma logica di ciascun enunciato dell'argomentazione 1.5 possiamo rappresentare la forma logica dell'intera argomentazione, ovvero

(1.6) $Fa,\ \forall x\,(Fx \rightarrow Rx)\ \therefore\ Ra.$

1.4 Condizioni di verità delle formule

Come abbiamo già detto, un'argomentazione è valida quando la conclusione è certamente vera nell'ipotesi che siano vere le premesse. Per stabilire se l'argomentazione formale 1.6 è valida dobbiamo quindi definire le *condizioni di verità* delle formule che la compongono. A questo scopo introduciamo una struttura algebrica, che seguendo la tradizione della logica moderna chiameremo *modello*, costruita nel modo seguente.

Il primo passo consiste nel definire un insieme non vuoto D di individui, che chiameremo *dominio*. Intuitivamente, il dominio è un insieme che contiene tutti gli individui di cui ci interessa parlare, e per il resto arbitrario.

Il secondo passo nella definizione del modello consiste nell'assegnare un'*interpretazione* nel dominio alle costanti e ai predicati che abbiamo utilizzato. Per definire l'interpretazione si procede nel modo seguente:

- Dato che ogni costante fa riferimento a un individuo, e che gli individui sono gli elementi del dominio, a ogni costante associamo esattamente un elemento di D, che chiamiamo il *referente* della costante; nel caso di una generica costante c avremo quindi che $ref(c) \in D$.
- Dato che ogni predicato rappresenta una proprietà degli individui, a ogni predicato associamo esattamente un sottoinsieme di D, chiamato l'*estensione* del predicato, intesa come l'insieme di tutti gli individui del dominio che godono della proprietà rappresentata dal predicato; nel caso di un generico predicato P avremo quindi che $ext(P) \subseteq D$.

Ogni dominio D, assieme alle interpretazioni dei predicati e delle costanti, costituisce un particolare modello M. Dunque un modello M può essere definito come una terna ordinata

$M = \langle D,ext,ref\rangle,$

dove D è un insieme non vuoto,

$D \neq \varnothing,$

ext è una funzione dall'insieme **P** di tutti i predicati ai sottoinsiemi di D,

ext: $\mathbf{P} \longrightarrow 2^D$,

e ref è una funzione dall'insieme **C** di tutte le costanti a D,

ref: $\mathbf{C} \longrightarrow D$.

Il terzo passo consiste nello stabilire le condizioni di verità dei diversi tipi di formule. Il caso più semplice è costituito dalle formule del tipo Pc, dove P è un predicato qualsiasi e c è una costante qualsiasi. Per stabilire le condizioni di verità di una formula di questo genere ricordiamoci che:

- la formula Pc rappresenta il fatto che il predicato P è applicato all'argomento c;
- la costante c fa riferimento all'individuo $ref(c)$;
- il predicato P rappresenta una proprietà goduta, all'interno del dominio D, da tutti e soli gli individui appartenenti a $ext(P)$.

Pertanto in un particolare modello $M = \langle D,ext,ref\rangle$ le condizioni di verità di una formula del tipo Pc sono definite come segue:

Pc è vera in M se, e solo se, $ref(c) \in ext(P)$.

Utilizzando una notazione simbolica per rappresentare la verità di una formula in un modello, questa definizione si riscrive come

$M \models Pc$ se, e solo se, $ref(c) \in ext(P)$.

È importante notare che una formula è vera o falsa non in sé, ma soltanto relativamente a uno specifico modello *M*. Si noti che anche la frase italiana *Alberto è ricco* non è in sé né vera né falsa, e diviene vera o falsa soltanto quando stabiliamo chi sia esattamente Alberto (ovvero, quando stabiliamo il referente del nome *Alberto*) e che cosa significhi esattamente essere ricco (ovvero, quando stabiliamo l'estensione del predicato *è ricco*). In altri termini, anche le lingue naturali presuppongono un'interpretazione dei termini utilizzati.

Le condizioni di verità dell'enunciato $\forall x\,(Fx \rightarrow Rx)$ sono un po' più complesse da definire, e per questo motivo ce ne occuperemo più avanti. Per il momento ci basterà sapere che, dati due predicati arbitrari *P* e *Q*, l'enunciato $\forall x\,(Px \rightarrow Qx)$ è vero in *M* se, e solo se, l'estensione di *P* è un sottoinsieme dell'estensione di *Q*, ovvero

$M \models \forall x\,(Px \rightarrow Qx)$ se, e solo se, $ext(P) \subseteq ext(Q)$.

1.5 Validità di un'argomentazione formale

Ritorniamo ora al nostro esempio, ovvero all'argomentazione formale

(1.6) $Fa,\ \forall x\,(Fx \rightarrow Rx)\ \therefore\ Ra$.

L'insieme dei predicati è costituito da $\mathbf{P} = \{F,R\}$ e l'insieme delle costanti è costituito da $\mathbf{C} = \{a\}$. Per definire un modello dobbiamo fissare un dominio non vuoto *D* e definire le funzioni *ext* e *ref*; a titolo di esempio, consideriamo il modello $M_1 = \langle D,ext,ref\rangle$ con:

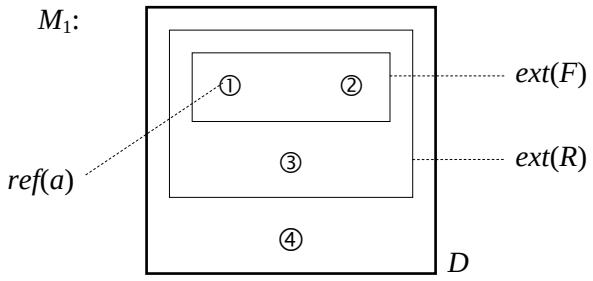
$D = \{①,②,③,④\}$,

$ext(F) = \{①,②\}$,

$ext(R) = \{①,②,③\}$,

$ref(a) = ①$.

Questo modello può essere rappresentato in modo più intuitivo con il grafico seguente:



Ricordando le interpretazioni dei simboli, possiamo dire che in questo modello gli individui ① e ② vanno in giro in Ferrari e sono ricchi, l'individuo ③ non va in giro in Ferrari ma è ricco, e l'individuo ④ non va in giro in Ferrari e non è nemmeno ricco.

È facile verificare che in M_1 sia le premesse, sia la conclusione della (6) sono vere; infatti:

$M_1 \models Fa$ perché $ref(a) = ①$, $ext(F) = \{①,②\}$, e quindi $ref(a) \in ext(F)$;

$M_1 \models \forall x\,(Fx \rightarrow Rx)$ perché $ext(F) = \{①,②\}$, $ext(R) = \{①,②,③\}$, e quindi $ext(F) \subseteq ext(R)$;

$M_1 \models Ra$ perché $ref(a) = ①$, $ext(R) = \{①,②,③\}$, e quindi $ref(a) \in ext(R)$.

Naturalmente ciò non è sufficiente a garantire che l'argomentazione sia valida. A priori, infatti, potrebbe esistere un altro modello, diciamo M_2, in cui le premesse della 1.6 sono vere mentre la conclusione è falsa; un modello del genere mostrerebbe che la verità della conclusione non è garantita dalla verità delle premesse.

Tuttavia è facile convincersi che un modello del genere non esiste. Consideriamo infatti un generico modello $M = \langle D,ext,ref\rangle$ in cui le premesse della 1.6 siano vere; tenendo conto delle condizioni di verità delle premesse, in M si dovrà comunque avere

$ref(a) \in ext(F)$,

$ext(F) \subseteq ext(R)$.

Ma in tal caso avremo anche

$ref(a) \in ext(R)$,

e quindi anche la conclusione della 1.6 è vera in M. Ne segue che la conclusione dell'argomentazione 1.6 è vera in ogni modello in cui siano vere le sue premesse: pertanto, per la definizione di validità, l'argomentazione 1.6 risulta valida.

Per stabilire che un'argomentazione è valida si procede dunque nel modo seguente:

- si riformula l'argomentazione in termini formali;
- si stabilisce che ogni modello che rende vere le premesse rende vera anche la conclusione.

Più precisamente, sia Γ un insieme arbitrario di formule logiche (vuoto o non vuoto, finito o infinito), e sia φ un'ulteriore formula logica. Se ogni modello che rende vere tutte le formule di Γ rende vera anche φ, si dice che Γ *comporta* (*entails*) φ, o che φ *è conseguenza logica di* Γ, o che φ *segue logicamente da* Γ, e si scrive

$\Gamma \models \varphi$.

Sia ora $\Gamma \therefore \varphi$ un'argomentazione formale arbitraria, dove Γ è l'insieme delle premesse e φ la conclusione. Diremo che tale argomentazione è *valida* se, e solo se,

$\Gamma \models \varphi$.

L'argomentazione formale 1.6 è quindi valida perché, come abbiamo mostrato con considerazioni di tipo insiemistico, si ha che

$Fa,\ \forall x\,(Fx \rightarrow Rx) \models Ra$.

1.5 Invalidità e controesempi

Mentre l'argomentazione 1.5 è valida, l'argomentazione

(1.7) *Alberto è ricco*

Tutti coloro che vanno in giro in Ferrari sono ricchi

$\therefore$ *Alberto va in giro in Ferrari*

non lo è, in quanto è possibile che le due premesse siano vere senza che lo sia la conclusione. Per mostrarlo in modo rigoroso passiamo alla forma logica,

(1.8) $Ra,\ \forall x\,(Fx \rightarrow Rx) \therefore Fa$,

e costruiamo un *controesempio*, ovvero un modello in cui le premesse siano vere e la conclusione falsa. Consideriamo ad esempio il modello $M_2 = \langle D,ext,ref\rangle$ con

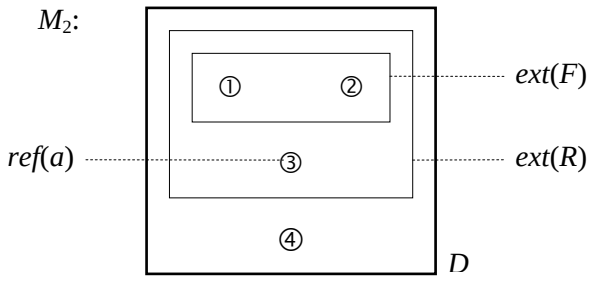
$D = \{①,②,③,④\}$,

$ext(F) = \{①,②\}$,

$ext(R) = \{①,②,③\}$,

$ref(a) = ③$,

ovvero:



Come si vede facilmente,

$M_2 \models Ra$ perché $ref(a) = ③$, $ext(R) = \{①,②,③\}$, e quindi $ref(a) \in ext(R)$;

$M_2 \models \forall x\,(Fx \rightarrow Rx)$ perché $ext(F) = \{①,②\}$, $ext(R) = \{①,②,③\}$, e quindi $ext(F) \subseteq ext(R)$;

$M_2 \not\models Fa$ perché $ref(a) = ③$, $ext(F) = \{①,②\}$, e quindi $ref(a) \notin ext(F)$.

Il modello M_2 è un controesempio per l'argomentazione 1.8, che risulta pertanto invalida.

1.6 Il calcolo

In base alle definizioni precedenti, non è difficile dimostrare che un'argomentazione è invalida: a questo scopo è sufficiente esibire un controesempio. Meno semplice è, in generale, dimostrare che un'argomentazione è valida. In questo caso dobbiamo stabilire che tutti i modelli delle premesse soddisfano la conclusione; ma, in generale, i modelli che soddisfano le premesse sono infiniti, e quindi non possono essere passati in rassegna uno per uno.

Prima della nascita della logica moderna, la mancanza di un metodo generale per dimostrare la validità di un'argomentazione ha turbato i sonni di molti pensatori. Nel Seicento, ad esempio, Leibniz (1678) auspicò la creazione di una *lingua rationalis*, costituita da una *characteristica universalis*, ovvero una notazione simbolica in grado di esprimere qualunque concetto, e da un *calculus ratiocinator*, in grado di stabilire in modo meccanico se un'argomentazione rappresentata nella notazione simbolica è o non è valida. Insomma, l'idea di Leibniz era di superare le interminabili controversie filosofiche sedendosi intorno a un tavolo e "calcolando" (*calculemus*!). Oggi sappiamo che il sogno di Leibniz, pur irrealizzabile nella sua interpretazione più generale, è realizzabile in una forma più limitata. Infatti, per poter disporre di un calcolo che verifichi meccanicamente la validità di un'argomentazione è necessario limitare l'espressività del linguaggio simbolico: non più una *characteristica universalis*, quindi, bensì un linguaggio più debole, ma pur sempre sufficiente a rappresentare la forma di argomentazioni interessanti e utili. Aver stabilito in che misura e con quali tecniche il sogno di Leibniz possa essere perseguito costituisce uno dei più spettacolari risultati della logica moderna e, in verità, dell'intera produzione matematica del Novecento.

Nell'ambito della logica un *calcolo*, o *procedura di prova*, è un algoritmo che consente di stabilire con una sequenza puramente meccanica di passi che fra un insieme di formule Γ e una formula φ sussiste una relazione di conseguenza logica. Possiamo vedere un calcolo come un programma che, ricevuti in ingresso un insieme Γ di premesse e una conclusione φ, fornisce in uscita una risposta binaria, *sì* o *no*; quando la risposta è "sì" diremo che φ *deriva da* Γ (o che φ *è deducibile da* Γ) e scriveremo

$\Gamma \vdash \varphi$.

Parlando in astratto, è un calcolo *qualunque* procedura che prenda in ingresso una coppia ordinata $\langle \Gamma,\varphi \rangle$, dove Γ è un insieme di formule e φ è un'ulteriore formula, e che fornisca in uscita una risposta binaria. Si dà il caso che sia possibile definire calcoli alquanto "patologici", come ad esempio:

- il calcolo "ottimista", che restituisce *sì* qualunque sia la coppia $\langle \Gamma,\varphi \rangle$ in ingresso;
- il calcolo "pessimista", che restituisce *no* qualunque sia la coppia $\langle \Gamma,\varphi \rangle$ in ingresso.

Calcoli del genere sarebbero in realtà del tutto inutili; ciò che desideriamo da un calcolo, infatti, è che risponda *sì* ogni volta che la coppia $\langle \Gamma, \varphi \rangle$ in ingresso è tale che $\Gamma \models \varphi$, e che risponda *no* ogni volta che la coppia $\langle \Gamma, \varphi \rangle$ in ingresso è tale che $\Gamma \not\models \varphi$. In termini più tecnici, desideriamo che un calcolo sia

- *corretto* (*sound*), nel senso che: se $\Gamma \vdash \varphi$ (ovvero, se il calcolo restituisce *sì*) allora $\Gamma \models \varphi$ (ovvero, φ è conseguenza logica di Γ);
- *completo*, nel senso che: se $\Gamma \models \varphi$ (ovvero, se φ è conseguenza logica di Γ), allora $\Gamma \vdash \varphi$ (ovvero, il calcolo restituisce *sì*).

Va notato che anche quando sussistono correttezza e completezza non sappiamo come un calcolo si comporti quando la coppia $\langle \Gamma, \varphi \rangle$ in ingresso è tale che $\Gamma \not\models \varphi$. In tal caso, vorremmo che il calcolo restituisse la risposta *no*; ciò è garantito se sappiamo che il calcolo, oltre a essere corretto e completo, termina in ogni caso in un numero finito di passi. Infatti:

- se un calcolo termina sempre in un numero finito di passi, restituirà una risposta binaria a fronte di qualunque coppia $\langle \Gamma, \varphi \rangle$ in ingresso;
- se $\Gamma \models \varphi$, dato che il calcolo è completo verrà restituito *sì*;
- se invece $\Gamma \not\models \varphi$, dato che il calcolo è corretto verrà restituito *no*.

In tal caso il calcolo è una *procedura di decisione per la conseguenza logica*, e una logica dotata di un calcolo del genere si dice *decidibile*.

La correttezza, la completezza e la decidibilità sono proprietà indubbiamente desiderabili, ma non hanno lo stesso peso. In particolare:

- La correttezza di un calcolo è una proprietà irrinunciabile, perché un calcolo che derivi formule che non sono conseguenza logica delle premesse è del tutto inutile. Come suggerisce lo stesso termine "correttezza", un calcolo non corretto è semplicemente un calcolo che fornisce risposte sbagliate!
- La completezza è una proprietà molto importante, ma non così irrinunciabile come la correttezza. Un calcolo corretto ma incompleto è un calcolo che, quando risponde, risponde correttamente; tuttavia, non sempre risponde *sì* quando $\Gamma \models \varphi$: in tal caso ovviamente non risponde (ovvero "entra in ciclo"), perché se rispondesse *no* non sarebbe corretto.
- Infine, la decidibilità è anch'essa importante, ma meno della completezza: un calcolo corretto e completo, ma che non sia una procedura di decisione, a volte non terminerà quando $\Gamma \not\models \varphi$. Tuttavia, il calcolo terminerà sempre quando $\Gamma \models \varphi$, altrimenti non sarebbe completo; per questo motivo un calcolo del genere viene chiamato *procedura di semidecisione*.

La completezza di un calcolo per la codiddetta *logica del primo ordine* (FOL, da *First Order Logic*; vedi il par. 2) è stata dimostrata per la prima volta da Gödel (1930). Nel 1928 Hilbert e Ackermann posero chiaramente il problema della decisione: esiste un calcolo che, oltre a essere corretto e completo, sia anche una procedura di decisione per FOL? Questo problema, cui si fa spesso riferimento con il nome originale di *Entscheidungsproblem*, fu all'origine di indagini molto feconde, che portarono alla formulazione del λ-calcolo (Church, 1936) e del concetto di Macchina di Turing (Turing, 1936).

La ragione per cui l'*Entscheidungsproblem* richiese l'elaborazione di un apparato teorico innovativo è facile da comprendere se si considera la domanda originale di Hilbert e Ackermann: esiste una procedura di decisione per FOL? Chiediamoci infatti come si possa rispondere a una simile domanda. Per dare risposta affermativa sarebbe sufficiente esibire uno specifico calcolo che abbia le proprietà desiderate, ovvero che sia corretto, sia completo e termini qualunque computazione in un numero finito di passi. Ma come è possibile dare una risposta negativa all'*Entscheidungsproblem*? In altre parole, come dimostrare che *non può esistere* una procedura meccanica che stabilisca in un numero finito di passi se sussiste o se non sussiste una relazione di conseguenza logica fra enunciati del primo ordine?

Nel 1936 sia Church, sia Turing dimostrarono indipendentemente (e con metodi diversi) che FOL non ammette una procedura di decisione; ma per fare questo dovettero proporre una definizione

matematica rigorosa di che cosa sia in generale una procedura meccanica. Le conseguenze del loro lavoro andarono molto al di là della semplice risposta negativa all'*Entscheidungsproblem*, in quanto gettarono le basi della teoria della computabilità e, più in generale, dell'informatica teorica.

Nel paragrafo 3 vedremo in modo dettagliato come si possa definire un calcolo corretto e completo per la logica del primo ordine.

1.7 Che cos'è la logica

A questo punto possiamo comprendere meglio che cosa sia la logica. A questo scopo è essenziale la distinzione fra forma e contenuto: la validità di un'argomentazione dipende dalle relazioni che sussistono fra le condizioni di verità delle premesse e della conclusione; a loro volta, queste relazioni dipendono dalla *forma logica delle premesse e delle conclusioni*, e non dal significato linguistico dei predicati e delle descrizioni definite contenuti nelle frasi. La logica, in effetti, non si occupa soltanto della validità delle argomentazioni; ad esempio, è sempre compito della logica mostrare che

Alberto è ricco oppure non è ricco

è un *enunciato valido*, nel senso che non può essere falso, e che

Alberto è ricco e non è ricco

è un *enunciato contraddittorio*, nel senso che non può essere vero. Anche in questo caso la dimostrazione passa per la traduzione degli enunciati nelle rispettive forme logiche, rispettivamente

$Ra \vee \neg Ra$,

dove il simbolo '$\vee$' corrisponde alla disgiunzione e il simbolo '$\neg$' corrisponde alla negazione, e

$Ra \wedge \neg Ra$,

dove il simbolo '$\wedge$' corrisponde alla congiunzione.

In conclusione, possiamo dire che *la logica è la disciplina che studia le proprietà di enunciati, e le relazioni fra enunciati, che dipendono dalla forma logica degli enunciati stessi*; *in particolare la logica si chiede in che misura tali proprietà e relazioni possano essere stabilite utilizzando procedimenti puramente meccanici*.

2. La logica predicativa del primo ordine: linguaggio e semantica

Come ogni logica, la *logica predicativa del primo ordine* o *elementare* (FOL) è costituita da tre componenti:

- un *linguaggio formale*, che modella un frammento del linguaggio naturale;
- una *semantica* del linguaggio formale, che definisce le condizioni sotto cui un'espressione del linguaggio è vera oppure è falsa;
- un *calcolo*, ovvero un algoritmo per stabilire la validità di frasi e argomentazioni in modo puramente meccanico.

2.1 Il linguaggio formale

Di seguito definiamo il *linguaggio predicativo del primo ordine* (o *linguaggio predicativo elementare*)[5], studiato per la prima volta come sistema logico a sé stante da Hilbert e Ackermann (1928). Il linguaggio è definito assegnando un *dizionario dei simboli* e una *grammatica*.

[5] Esistono anche definizioni leggermente diverse di linguaggio del primo ordine: ad esempio è possibile trascurare le costanti individuali, o al contrario considerarle come caso particolare di un più ampio insieme di *simboli funzionali* a *n* argomenti.

Dizionario dei simboli

I simboli del linguaggio appartengono a tre categorie: simboli logici, simboli descrittivi e simboli strutturali.

Simboli logici. Sono gli operatori logici (connettivi e quantificatori), il simbolo di uguaglianza e le variabili individuali (utilizzate essenzialmente assieme ai quantificatori). Più precisamente:

- il connettivo booleano di negazione, '$\neg$',
- il connettivo booleano di congiunzione, '$\wedge$',
- il quantificatore universale, '$\forall$',
- il simbolo dell'uguaglianza, '=';
- un insieme infinito numerabile $\mathbf{V} = \{x_1,...,x_n,...\}$ di variabili individuali (dette anche semplicemente *variabili* e scritte come *x*, *y*, *z*, ...).

Simboli descrittivi. I simboli descrittivi consentono di fare riferimento a individui e di predicare proprietà di individui o relazioni fra individui. Sono costituiti da:

- un insieme al più numerabile, eventualmente vuoto, $\mathbf{C} = \{a_1,...,a_n,...\}$ di *costanti individuali* (dette anche semplicemente *costanti* e scritte come *a*, *b*, *c*, ...);
- un insieme al più numerabile, eventualmente vuoto, $\mathbf{P} = \{P_1,...,P_n,...\}$ di *simboli predicativi* (detti anche semplicemente *predicati* e scritti anche come *P*, *Q*, *R*, ...), con una funzione #: $\mathbf{P} \longrightarrow \mathbf{N}$ che assegna a ogni simbolo un'*arità* (ovvero il numero di argomenti).

Simboli strutturali. Si tratta dei simboli che consentono di dare una precisa struttura a una formula: a questo scopo sono sufficienti le parentesi tonde (aperta e chiusa) e la virgola.

Sono poi chiamati *termini* i simboli utilizzati per fare riferimento a individui. L'insieme **T** dei termini è quindi definito come

$\mathbf{T} = \mathbf{V} \cup \mathbf{C}$.

Grammatica delle formule

L'insieme **A** delle *formule atomiche* è definito come segue:

- $Pt_1...t_n \in \mathbf{A}$ se $P \in \mathbf{P}$, $t_i \in \mathbf{T}$ e $\#(P) = n$;
- $(t_1 = t_2) \in \mathbf{A}$ se $t_1, t_2 \in \mathbf{T}$;
- null'altro appartiene ad **A**.

Le formule atomiche costituite da un predicato di arità zero, ovvero della forma *P* con $\#(P) = 0$, sono anche dette *proposizioni atomiche*.

L'insieme $\mathbf{\Phi}$ delle *formule* è definito come segue:

- $\mathbf{A} \subseteq \mathbf{\Phi}$;
- $\neg\varphi \in \mathbf{\Phi}$ se $\varphi \in \mathbf{\Phi}$;
- $(\varphi \wedge \psi) \in \mathbf{\Phi}$ se $\varphi, \psi \in \mathbf{\Phi}$;
- $\forall x\varphi$ se $x \in \mathbf{V}$ e $\varphi \in \mathbf{\Phi}$;
- null'altro appartiene a $\mathbf{\Phi}$.

Si noti che la definizione delle formule è ricorsiva.

Abbreviazioni

Ulteriori simboli logici vengono introdotti come abbreviazioni:

- connettivo di disgiunzione: $(\varphi \vee \psi)$ abbrevia $\neg(\neg\varphi \wedge \neg\psi)$
- connettivo condizionale: $(\varphi \rightarrow \psi)$ abbrevia $(\neg\varphi \vee \psi)$
- connettivo bicondizionale: $(\varphi \leftrightarrow \psi)$ abbrevia $((\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi))$
- quantificatore esistenziale: $\exists x\varphi$ abbrevia $\neg\forall x\neg\varphi$
- simbolo di disuguaglianza: $(t_1 \neq t_2)$ abbrevia $\neg(t_1 = t_2)$

Eliminazione delle parentesi superflue

Si può eliminare una coppia di parentesi esterna; esempio:

$((Pa \wedge \exists xQax) \rightarrow (Ra \rightarrow Qaa)) \longrightarrow (Pa \wedge \exists xQax) \rightarrow (Ra \rightarrow Qaa)$

Gli operatori ripetuti associano da destra:

$(Pa \wedge \exists xQax) \rightarrow (Ra \rightarrow Qaa) \longrightarrow (Pa \wedge \exists xQax) \rightarrow Ra \rightarrow Qaa$

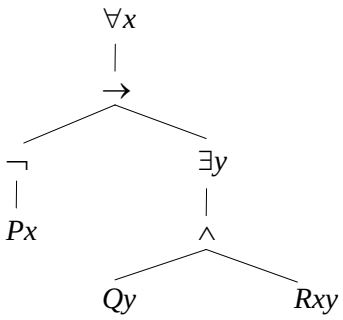
Gli operatori hanno diversa precedenza: $\{\neg, \forall x, \exists x\} > \{\wedge, \vee\} > \{\rightarrow, \leftrightarrow\}$

$(Pa \wedge \exists xQax) \rightarrow Ra \rightarrow Qaa \longrightarrow Pa \wedge \exists xQax \rightarrow Ra \rightarrow Qaa$

Struttura ad albero

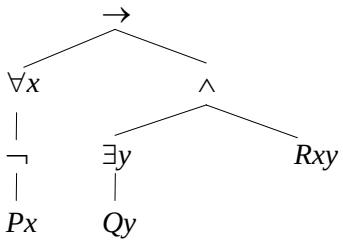
La struttura ricorsiva delle formule è messa chiaramente in luce da una notazione ad albero, in cui le foglie rappresentano formule atomiche e i nodi interni rappresentano operatori (connettivi o quantificatori); ad esempio:

(2.1) $\forall x\,(\neg Px \rightarrow \exists y\,(Qy \wedge Rxy))$

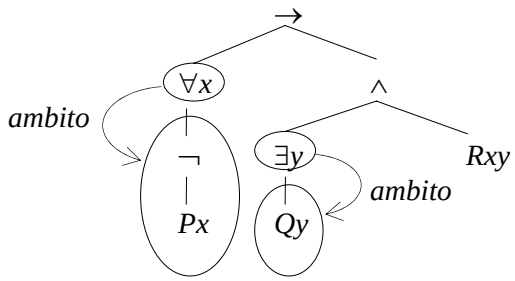


Le coppie di parentesi della scrittura lineare corrispondono a biforcazioni nell'albero. Attenzione: scrivendo la 2.1 senza alcuna parentesi si ottiene *una formula diversa*:

(2.2) $\forall x \neg Px \rightarrow \exists y Qy \wedge Rxy$

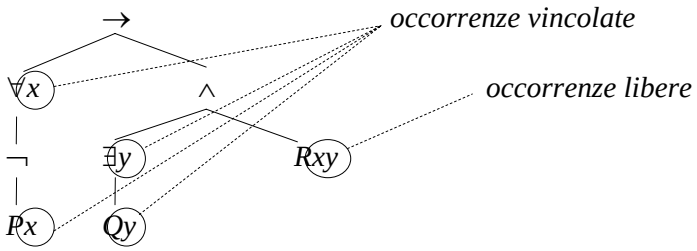


L'*ambito* di un quantificatore è la formula rappresentata dal sottoalbero che sta sotto un quantificatore:



Variabili e occorrenze di variabili

La formula 2.1 ha *due variabili*, *x* e *y*. La variabile *x* e la variabile *y* hanno *tre occorrenze ciascuna*. Ciascuna occorrenza di variabile può essere *libera* o *vincolata*: sono vincolate le occorrenze di una variabile *x* che fanno parte di un quantificatore $\forall x$ o $\exists x$, o che sono contenute nell'ambito di un quantificatore $\forall x$ o $\exists x$; ogni altra occorrenza è libera.



Formule chiuse e formule aperte

Una formula si dice *chiusa* se ogni occorrenza delle variabili nella formula è vincolata, e si dice *aperta* in caso contrario. Se φ è una formula aperta contenente occorrenze libere delle variabili x_1, ..., x_n (e soltanto di queste), si dice *chiusura universale di* φ la formula

$\forall x_1 \ldots \forall x_n \varphi$

e *chiusura esistenziale di* φ la formula

$\exists x_1 \ldots \exists x_n \varphi$

Frasi o enunciati

Una formula chiusa è detta anche *frase* o *enunciato* (*sentence*). Il motivo è che nei casi in cui una frase assertiva del linguaggio ordinario si può rappresentare con una formula del primo ordine, la rappresentazione è tipicamente una formula chiusa. Esempi:

se non piove qualcuno verrà $\neg P \rightarrow \exists x\, Vx$

tutti gli esseri umani hanno una madre $\forall x\, (Hx \rightarrow \exists y\, Myx)$

Predicati

Un *predicato* (*del primo ordine* o *elementare*) è una proprietà che un determinato individuo può possedere o non possedere, o una relazione che può sussistere o non sussistere fra coppie, terne, ecc., di individui. Con una certa improprietà di linguaggio, si chiamano "predicati" anche le formule che rappresentano predicati. Ad esempio, la formula

$Hx \wedge \exists y\, (Ay \wedge Pxy)$

può rappresentare il predicato monàdico *gli esseri umani che possiedono un'automobile*, e la formula

$Nx \wedge Ny \wedge (x < y)$

può rappresentare il predicato diàdico *le coppie di numeri naturali in cui il primo numero è minore del secondo*. Ancora più impropriamente, come abbiamo già visto, si chiamano spesso "predicati" le costanti predicative del linguaggio (*P*, *Q*, *R*, ...).

Formule sovraquantificate

In generale, un predicato è rappresentato da una formula aperta, ovvero da una formula per così dire "sottoquantificata". Esistono anche formule che potremmo chiamare "sovraquantificate", come

$\exists x\, P, \qquad \forall x\, \exists x\, Px, \qquad \forall x\, \forall y\, Px.$

A tali formule non corrispondono frasi del linguaggio ordinario. Ci si può chiedere come mai la grammatica di FOL non le escluda, considerandole semplicemente scorrette. La ragione è che in tal caso la grammatica dovrebbe essere più complessa: per l'esattezza, si dovrebbe passare da una

grammatica di tipo *non contestuale* (*context free*) a una grammatica di tipo *contestuale* (*context sensitive*), perdendo così la struttura ad albero delle formule. Conseguenza di una simile mossa sarebbe l'impossibilità di utilizzare tecniche dimostrative che sfruttano la struttura ad albero, come l'*induzione sulla struttura delle formule* (analoga alla ben nota induzione aritmetica).

Linguaggio o linguaggi?

Fino ad ora abbiamo parlato, in modo alquanto improprio, di "linguaggio" del primo ordine. In realtà, le definizioni che abbiamo dato specificano piuttosto una *famiglia di linguaggi*, parametrica negli insiemi **P** dei predicati e **C** delle costanti; dovremmo quindi, più correttamente, scrivere $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$ anziché $\mathbf{\Phi}$.

La parametricità di $\mathbf{\Phi}$ non è un fatto marginale: in generale, infatti, possiamo considerare un linguaggio logico come costituito da due componenti, una *indipendente dall'applicazione* (i simboli logici e i simboli strutturali) e una *dipendente dall'applicazione* (i simboli descrittivi). Nei linguaggi del primo ordine utilizzeremo sempre i connettivi booleani, i quantificatori e l'uguaglianza; le costanti e i predicati, invece, cambieranno a seconda dell'applicazione. Ad esempio, nell'aritmetica del primo ordine avremo una costante, 0, per denotare lo zero e due predicati, N ed S, per rappresentare rispettivamente la proprietà di essere un numero naturale e la relazione fra un numero naturale e il suo successore. Il linguaggio dell'aritmetica del primo ordine sarà quindi $\mathbf{\Phi}(\{N,S\},\{0\})$, con $\#(N) = 1$ e $\#(S) = 2$.

Si noti che anche per $\mathbf{P} = \mathbf{C} = \varnothing$ si ottiene comunque un linguaggio interessante; ad esempio, $\mathbf{\Phi}(\varnothing,\varnothing)$ contiene le formule

$\forall x\,(x = x)$,

$\forall x\,\forall y\,((x = y) \rightarrow (y = x))$,

$\forall x\,\forall y\,\forall z\,((x = y) \wedge (y = z) \rightarrow (x = z))$,

che esprimono la riflessività, la simmetria e la transitività dell'eguaglianza. Infine, si noti che la cardinalità di $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$, per qualunque scelta di **P** e **C** in accordo con le definizioni date, è sempre $\aleph_0$ (aleph-zero, cioè la cardinalità del numerabile).

2.2 La semantica

Assegnare una semantica a un linguaggio logico significa definire un criterio per poter dire, ad esempio, che un'argomentazione è valida. Nella lezione precedente abbiamo visto alcuni punti importanti, e in particolare che:

- la validità di un'argomentazione si riduce alla relazione di conseguenza logica fra formule;
- a sua volta, il concetto di conseguenza logica si basa sul concetto di verità di una formula;
- in generale una formula non è vera o falsa in sé, ma soltanto relativamente a una ben precisa interpretazione dei simboli descrittivi del linguaggio (i predicati e le costanti).

Ne segue che il passo fondamentale per specificare la semantica è definire quando una formula φ è vera in un modello M, che si scrive $M \models \varphi$.

Modelli

Un *modello del primo ordine* è una struttura matematica che modella un frammento della realtà. Tale frammento è modellato come un insieme non vuoto di individui, detto *dominio* o *universo*, fra i quali sussistono relazioni n-arie. Una relazione n-aria R su un dominio D, con $n \geq 0$, è un insieme (eventualmente vuoto) $R \subseteq D^n$, ovvero un insieme di n-ple $\langle d_1,...,d_n\rangle$, con $d_i \in D$. Nel caso particolare in cui si abbia $n = 1$ la relazione R si riduce a un sottoinsieme di D, in quanto $D^1 = D$. Nel caso particolare in cui si abbia $n = 0$ la relazione R si riduce a un *valore binario*, in quanto $D^0 = \mathbf{1}$ (l'insieme unitario, costituito da un solo elemento); poiché D^0 contiene soltanto l'insieme vuoto $\mathbf{0} = \varnothing$ e l'insieme unitario **1**, l'asserzione $R \subseteq D^0$ rappresenta una scelta binaria. Chiameremo inoltre D^* l'unione di tutte le n-ple $\langle d_1,...,d_n\rangle$, con $d_i \in D$, per ogni $n \geq 0$.

Il dominio serve da base per l'*interpretazione* dei simboli descrittivi del linguaggio (ovvero le costanti e i predicati). A ogni predicato P di arità n viene associata una relazione n-aria $ext(P)$ nel dominio, detta *estensione* del predicato, e a ogni costante c viene associato un individuo $ref(c)$ del dominio, detto *referente* della costante:

ext: $\mathbf{P} \longrightarrow D^*$, con $ext(P) \subseteq D^{\#(P)}$;

ref: $\mathbf{C} \longrightarrow D$.

Un modello M è ora, per definizione, la terna ordinata costituita da D, ext e ref:

$M = \langle D,ext,ref \rangle$.

È importante analizzare la relazione fra un linguaggio del primo ordine $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$ e i relativi modelli. Tutti i modelli di $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$ hanno qualcosa in comune in quanto, fissato il dominio D, contengono lo stesso numero di relazioni di arità corrispondente (una per ogni predicato) e lo stesso numero di "elementi selezionati" (uno per ogni costante, non necessariamente distinti fra loro). Nel seguito la notazione $\mathbf{Mod}(\mathbf{P},\mathbf{C})$ per denotare la totalità dei modelli[6] di $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$.

Assegnamento di valori alle variabili

Come abbiamo in parte già visto nella lezione precedente, la verità di una formula in un modello dipende dall'interpretazione dei simboli descrittivi nel dominio del modello, ovvero dagli individui associati alle costanti e dalle relazioni (nel senso insiemistico del termine) associate ai predicati. Un problema è costituito dalla presenza di variabili, che dovendo appunto essere lasciate libere di "variare" non possono avere un valore fissato una volta per tutte dall'interpretazione. In generale, quindi, la verità di una formula non dipende soltanto dall'interpretazione dei simboli descrittivi, ma anche dall'assegnamento di individui alle variabili. Sia D un dominio non vuoto, e per il resto arbitrario, di individui; un *assegnamento* (di individui alle variabili) è una funzione

val: $\mathbf{V} \longrightarrow D$.

Dunque anche le variabili, come le costanti, sono utilizzate per denotare individui del dominio. Se $M = \langle D,ext,ref \rangle$ è un modello e t è un termine (e quindi una costante o una variabile), definiamo nel modo seguente la *denotazione* $den(t)$ di t:

$den(t) = ref(t)$ se t è una costante,

$den(t) = val(t)$ se t è una variabile.

Come vedremo, per definire le condizioni di verità delle formule quantificate è importante considerare equivalenti due assegnamenti quando differiscono al più per il valore assegnato a una variabile prefissata x. Diciamo quindi che due assegnamenti val e val' sono x-equivalenti, e scriviamo

$val \cong_x val'$,

se, e solo se,

$val(y) = val'(y)$ per ogni variabile y non coincidente con x.

Condizioni di verità

Sfruttando il fatto che l'insieme $\mathbf{\Phi}$ delle formule è definito ricorsivamente possiamo specificare le condizioni di verità per induzione sulla struttura della formula, utilizzando le formule atomiche come base dell'induzione. Definiremo ora che cosa significa che *una formula φ è vera in un modello M per un assegnamento val*, in simboli

$M,val \models \varphi$,

[6] Se come dominio di un modello possiamo assumere *qualunque* insieme non vuoto, la totalità $\mathbf{Mod}(\mathbf{P},\mathbf{C})$ dei modelli di $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$ forma non un insieme, ma una *classe* (per approfondimenti si veda un testo di teoria assiomatica degli insiemi).

o, come anche si dice, che *M e val verificano* φ, o che *M e val soddisfano* φ. Quando *M* e *val* non soddisfano φ,

$M,val \not\models \varphi$,

diremo anche che φ *è falsa in M e val*, o che *M e val falsificano* φ. Se $M = \langle D,ext,ref\rangle$ è un modello e *val* è un assegnamento definiamo[7]

$M,val \models Pt_1...t_n$ sse $\langle den(t_1),..., den(t_n)\rangle \in ext(P)$;

$M,val \models (t_1 = t_2)$ sse $den(t_1) = den(t_2)$;

$M,val \models \neg\varphi$ sse $M,val \not\models \varphi$;

$M,val \models (\varphi \wedge \psi)$ sse $M,val \models \varphi$ e $M, val \models \psi$;

$M,val \models \forall x\varphi$ sse $M,val' \models \varphi$ per ogni assegnamento *val'* tale che $val' \cong_x val$.

La verità di una formula φ in un modello *M* si definisce ora come la verità di φ in *M* per qualsiasi assegnamento di valori alle variabili:

$M \models \varphi$ sse $M,val \models \varphi$ per ogni assegnamento *val*.

Quando φ è vera in *M* si dice anche che *M soddisfa* φ, o che *M verifica* φ o che *M è un modello di* φ. Infine, se Γ è un insieme arbitrario (vuoto, finito o infinito numerabile) di formule, diciamo che *M* soddisfa Γ,

$M \models \Gamma$,

se e solo se *M* soddisfa ciascuna formula di Γ. Si noti che se una formula chiusa φ è vera in *M* sotto un assegnamento *val*, allora φ è vera in *M* sotto qualunque assegnamento, e quindi è vera in *M* senz'altro: dunque la verità degli enunciati (formule chiuse) non dipende dall'assegnamento di valori alle variabili. Si noti inoltre che se la formula φ contiene occorrenze libere di *n* variabili x_1, ..., x_n, allora φ è vera in *M* se e solo se lo è la sua chiusura universale:

$M \models \varphi$ sse $M \models \forall x_1 ... \forall x_n \varphi$.

Questo fatto giustifica l'uso matematico di scrivere enunciati universali sottintendendo i quantificatori; ad esempio, la legge di tricotomia dei numeri reali si scrive tipicamente come:

$(x > y) \vee (x < y) \vee (x = y)$,

tralasciando i quantificatori $\forall x$ e $\forall y$.

2.3 Classificazione semantica delle formule

È facile convincersi che esistono formule che sono vere in qualsiasi modello, formule che sono false in qualsiasi modello e formule che sono vere o false a seconda del modello. Ad esempio, ogni formula di forma $\varphi \vee \neg\varphi$ è vera in qualunque modello, ogni formula di forma $\varphi \wedge \neg\varphi$ è falsa in qualunque modello, e ogni formula di forma $\exists xPx$ è vera in qualche modello e falsa in altri. Per definizione,

- una formula $\varphi \in \mathbf{\Phi}(\mathbf{P},\mathbf{C})$ si dice *valida* o *logicamente vera* se per ogni modello $M \in \mathbf{Mod}(\mathbf{P},\mathbf{C})$ e per ogni assegnamento *val* si ha

  $M,val \models \varphi$ (e quindi per ogni modello si ha $M \models \varphi$);
- una formula $\varphi \in \mathbf{\Phi}(\mathbf{P},\mathbf{C})$ si dice *soddisfacibile* se per qualche modello $M \in \mathbf{Mod}(\mathbf{P},\mathbf{C})$ e per qualche assegnamento *val* si ha

  $M,val \models \varphi$.

[7] I modelli qui definiti si dicono anche *modelli normali*, in quanto il simbolo di uguaglianza è interpretato come l'*identità* fra gli individui del dominio. È possibile interpretare l'uguaglianza, in senso più ampio, come una relazione di *congruenza*, ovvero come una relazione di equivalenza nel dominio che sia compatibile con tutte le relazioni definite nel modello.

Una formula si dice poi (vedi la tab. 2.1):

- *invalida* se non è valida, ovvero se è falsa qualche modello $M \in$ **Mod**(**P**,**C**) e per qualche assegnamento *val*;
- *insoddisfacibile* o *logicamente falsa* se non è soddisfacibile, ovvero se è falsa ogni modello $M \in$ **Mod**(**P**,**C**) e per ogni assegnamento *val*;
- *contingente* se non è né valida né insoddisfacibile, ovvero se è vera qualche modello $M \in$ **Mod**(**P**,**C**) e per qualche assegnamento *val*, e vera qualche modello $M' \in$ **Mod**(**P**,**C**) e per qualche assegnamento *val'*.

Per asserire che una formula φ è valida scriviamo

$\models \varphi$;

scriviamo poi

$\not\models \varphi$

per asserire che φ è invalida. Considerando ora la classificazione delle formule e la semantica della negazione, abbiamo che:

$\models \neg\varphi$ significa che φ è insoddisfacibile;

$\not\models \neg\varphi$ significa che φ è soddisfacibile;

$\not\models \varphi$ e $\not\models \neg\varphi$ significa che φ è contingente.

Infine, un insieme Γ di formule (vuoto, finito o infinito numerabile) si dice (*congiuntamente*) *soddisfacibile* quando esistono un modello M e una valutazione *val* che soddisfano simultaneamente ogni formula di Γ; in caso contrario Γ si dice (*congiuntamente*) *insoddisfacibile*.

È interessante notare che le frasi che si traducono in formule valide suonano come *non informative in quanto banali*, "lapalissiane" (ad es. "Fabrizio è in casa oppure non è in casa"); simmetricamente, le frasi che si traducono in formule insoddisfacibili suonano come *non informative in quanto contraddittorie* (ad es. "Fabrizio è in casa e non è in casa"). Appaiono quindi come informative solo le frasi che si traducono in formule contingenti (ad es. "Fabrizio è in casa", oppure "Fabrizio non è in casa"): questo fatto sarà approfondito nel paragrafo 2.5.

Implicazione ed equivalenza

La semantica delle formule condizionali, ovvero della forma $\varphi \rightarrow \psi$ (dove φ è detta l'*antecedente* e ψ il *conseguente* del condizionale), è spesso considerata problematica. Considerando il fatto che $\varphi \rightarrow \psi$ abbrevia $\neg\varphi \vee \psi$, abbiamo che

$M,val \models (\varphi \rightarrow \psi)$ sse $M,val \not\models \varphi$ oppure $M, val \models \psi$.

Il condizionale così definito è anche detto *condizionale materiale*. Il problema del condizionale materiale è che dà luogo a risultati apparentemente paradossali, in quanto è vero quando l'antecedente è falso oppure quando il conseguente è vero, anche se non sussiste alcun legame concettuale fra le due formule.

Tab. 2.1. Classificazione semantica delle formule.

<table>
<tr><td>condizione</td><td>vera in ogni modello per ogni assegnamento</td><td>vera in qualche modello per qualche assegnamento, e falsa in qualche modello per qualche assegnamento</td><td>falsa in ogni modello per ogni assegnamento</td></tr>
<tr><td rowspan="3">classificazione</td><td colspan="2">soddisfacibile</td><td>insoddisfacibile</td></tr>
<tr><td>valida</td><td colspan="2">invalida</td></tr>
<tr><td></td><td colspan="2">contingente</td></tr>
</table>

Ad esempio, l'enunciato condizionale

(2.3) *se Londra è in Francia, allora gli asini volano*

è vero, in quanto l'antecedente è falso. Questo risultato, in realtà, è meno paradossale di quanto possa sembrare; ad esempio, l'enunciato 2.3 è certamente accettabile come asserzione ironica, volta ad affermare che Londra non è in Francia. Ciò si spiega nel modo seguente: assumendo che il condizionale 2.3 sia vero, e dato che è noto che il suo conseguente è falso, l'unica alternativa possibile è che sia falso anche l'antecedente – il che è proprio quanto si vuole asserire. Ad ogni modo, benché l'enunciato 2.3 possa essere *vero* in certi modelli, certamente non è *valido*: se rappresentiamo *Londra è in Francia* con la proposizione atomica P, e *gli asini volano* con la proposizione atomica Q, la 2.3 diventa

$P \rightarrow Q$,

che è un enunciato contingente. In certi casi, invece, un condizionale è valido; ad esempio lo è il condizionale

(2.4) $P \wedge Q \rightarrow P$.

Un condizionale valido si chiama *implicazione*: l'enunciato 2.4 rappresenta quindi un'implicazione. Considerazioni analoghe si possono fare per gli enunciati bicondizionali. Considerando il fatto che $\varphi \leftrightarrow \psi$ abbrevia $(\varphi \rightarrow \psi) \wedge (\psi \rightarrow \varphi)$ abbiamo che

$M,val \models (\varphi \leftrightarrow \psi)$ sse $M,val \models \varphi$ e $M, val \models \psi$, oppure $M,val \not\models \varphi$ e $M, val \not\models \psi$.

Il bicondizionale così definito è anche detto *bicondizionale materiale*. Un bicondizionale valido si chiama *equivalenza*; ad esempio rappresenta un'equivalenza il bicondizionale

$(P \wedge Q \leftrightarrow Q \wedge P)$.

2.4 La conseguenza logica

Quando ci si occupa di un'argomentazione non interessa la validità "assoluta" della conclusione, bensì la validità della conclusione "relativamente alle premesse": in altre parole, non interessa sapere se la conclusione è vera in tutti i modelli, ma soltanto se la conclusione è vera in tutti quei modelli che soddisfano le premesse. Diremo quindi che un'argomentazione

$\Gamma \therefore \varphi$

è valida se, e soltanto se,

$M \models \varphi$ per ogni modello M tale che $M \models \Gamma$.

In tal caso diremo che Γ *comporta* (*entails*) φ, o che φ *è conseguenza logica di* Γ o che φ segue logicamente da Γ e scriveremo[8]

$\Gamma \models \varphi$.

La validità di una formula può ora essere vista come un caso particolare della conseguenza logica, quando l'insieme Γ delle premesse è vuoto:

$\models \varphi$ se, e solo se, $\varnothing \models \varphi$.

Dunque una formula è valida se è conseguenza logica di un insieme vuoto di premesse; viceversa, possiamo vedere la conseguenza logica $\Gamma \models \varphi$ come la validità di φ *relativizzata rispetto a* Γ.

Per la logica del primo ordine vale un importante teorema, detto *teorema della conseguenza logica*: se φ e ψ sono formule chiuse si ha che

$\Gamma, \varphi \models \psi$ se, e solo se, $\Gamma \models \varphi \rightarrow \psi$.

---

[8] Come si vede, il simbolo $\models$ è notevolmente "sovraccarico", in quanto a seconda dei casi rappresenta la verità di una formula in un modello e una valutazione, la verità di una formula in un modello, la verità di un insieme di formule in un modello, la validità di una formula, o la conseguenza logica. Tuttavia, le convenzioni sull'uso dei simboli metalinguistici (M, val, φ, Γ) rendono la notazione non ambigua.

Tenendo conto del fatto che $\varphi \rightarrow (\psi \rightarrow \chi)$ è equivalente a $(\varphi \wedge \psi) \rightarrow \chi$, questo teorema implica che quando $\Gamma = \{\varphi_1, ..., \varphi_n\}$ è un insieme finito di formule chiuse ogni relazione di conseguenza logica $\Gamma \models \varphi$ è equivalente alla validità di una formula; più precisamente,

$\varphi_1, ..., \varphi_n \models \psi$ se, e solo se, $\models \varphi_1 \wedge ... \wedge \varphi_n \rightarrow \psi$.

Ne segue che la validità di un'argomentazione con un numero finito di premesse è equivalente alla validità di una formula, e precisamente di una formula condizionale che ha la congiunzione delle premesse come antecedente e la conclusione come conseguente. Il concetto di validità di una formula è quindi sufficiente a trattare la validità di un'argomentazione con un numero finito di premesse. Ad esempio, la validità dell'argomentazione

(1.6) $Fa,\ \forall x\,(Fx \rightarrow Rx)\ \therefore\ Ra,$

può essere ricondotta alla validità della formula

$Fa \wedge \forall x\,(Fx \rightarrow Rx) \rightarrow Ra$.

Nell'analisi di problemi matematici e filosofici, tuttavia, è importante poter assumere un insieme infinito numerabile di premesse: un'argomentazione del genere non può essere rappresentata con un'unica formula, perché ciascuna formula del linguaggio $\mathbf{\Phi}$ è finita.

2.5 Un semplice modello del discorso

Prima di passare alla terza componente fondamentale di FOL, il calcolo, desideriamo discutere un semplice modello del discorso che scaturisce in modo naturale da quanto abbiamo detto finora. Un *discorso* è una successione finita di enunciati linguistici, che possiamo rappresentare come una successione finita di formule chiuse

$\Gamma = \langle \varphi_1,..., \varphi_n \rangle$.

Ci interessa ora modellare il "contributo informativo" del discorso, riducendolo alla composizione dei contributi informativi dei singoli enunciati. Cominciamo fissando uno specifico linguaggio del primo ordine, $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$; in altre parole, fissiamo l'insieme **P** dei predicati e l'insieme **C** delle costanti che potranno essere utilizzati nel discorso. In questo modo risulta fissata anche la classe $\mathbf{Mod}(\mathbf{P},\mathbf{C})$ dei modelli; questa classe gioca un ruolo molto importante, in quanto rappresenta *la totalità delle situazioni concrete descrivibili con il linguaggio* $\mathbf{\Phi}(\mathbf{P},\mathbf{C})$.

Ammettiamo che prima dell'inizio del discorso non si sappia nulla della realtà: ciò significa che, per quanto ne sappiamo, ogni modello potrebbe descrivere la realtà altrettanto bene; in altre parole, tutti gli elementi di $\mathbf{Mod}(\mathbf{P},\mathbf{C})$ sono rappresentazioni accettabili della realtà. Che cosa significa allora che un enunciato dà un certo *contributo informativo*? Significa che l'enunciato *riduce* la classe dei modelli accettabili della realtà. Più precisamente:

- all'inizio del discorso la realtà è rappresentata dall'intera classe $\mathbf{M}_0 = \mathbf{Mod}(\mathbf{P},\mathbf{C})$;
- dopo l'asserzione di φ_1 la realtà è rappresentata dalla sottoclasse $\mathbf{M}_1$ di $\mathbf{M}_0$ costituita da tutti e soli i modelli di $\mathbf{M}_0$ che soddisfano φ_1;
- in generale, dopo l'asserzione di φ_i la realtà è rappresentata dalla sottoclasse $\mathbf{M}_i$ di $\mathbf{M}_{i-1}$ costituita da tutti e soli i modelli di $\mathbf{M}_{i-1}$ che soddisfano φ_i;
- alla fine del discorso la realtà è rappresentata da tutti e soli i modelli di $\mathbf{M}_n$.

A questo punto è possibile fare alcune osservazioni:

- Se l'enunciato φ_i è valido, abbiamo che $\mathbf{M}_i = \mathbf{M}_{i-1}$ perché ovviamente tutti i modelli di $\mathbf{M}_{i-1}$ soddisfano φ_i. Quindi un enunciato valido non ha contenuto informativo, perché asserirlo *non ci dice nulla* sulla realtà.
- Anche quando l'enunciato φ_i è conseguenza logica degli enunciati precedenti abbiamo che $\mathbf{M}_i = \mathbf{M}_{i-1}$, perché per la definizione di conseguenza logica tutti i modelli di $\mathbf{M}_{i-1}$ soddisfano φ_i. Anche in questo caso l'enunciato non ha contenuto informativo, perché asserirlo *non ci dice nulla di nuovo* sulla realtà.

- Se l'enunciato φ_i è insoddisfacibile, abbiamo che $\mathbf{M}_i = \varnothing$, perché nessun modello può soddisfare un enunciato del genere. Asserire un enunciato insoddisfacibile introduce una contraddizione nel discorso, che cessa pertanto di rappresentare alcunché. Dal fatto che $\mathbf{M}_i = \varnothing$ deriva poi che $\mathbf{M}_k = \varnothing$ per ogni $i \leq k \leq n$: in altre parole, una volta introdotta una contraddizione nel discorso non si può più rimediare (in effetti, per rimediare occorrerebbe "ritornare indietro" nel discorso e ritirare l'affermazione che ha introdotto la contraddizione).
- Anche quando la *negazione* dell'enunciato φ_i è conseguenza logica degli enunciati precedenti abbiamo che $\mathbf{M}_i = \varnothing$. Quindi anche asserire un enunciato che contrasta con quanto detto precedentemente introduce una contraddizione nel discorso.
- Cambiando l'ordine di alcuni enunciati nel discorso non altera il risultato finale, $\mathbf{M}_n$. In effetti è discutibile che ciò avvenga nei discorsi in linguaggio naturale: si tratta di un tipico "difetto di approssimazione", dovuto al fatto che la nostra rappresentazione del discorso umano è molto semplificata.

Si noti che gli enunciati di un discorso possono essere classificati in tre categorie:

- enunciati "inutili" in quanto non informativi: sono gli enunciati validi e le conseguenze logiche degli enunciati precedenti;
- enunciati "deleteri" in quanto introducono una contraddizione: sono gli enunciati insoddisfacibili e le negazioni delle conseguenze logiche degli enunciati precedenti;
- enunciati "utili", in quanto riducono la classe dei modelli accettabili senza renderla vuota: si tratta degli enunciati *contingenti relativamente agli enunciati precedenti*, ovvero degli enunciati tali che né l'enunciato stesso né la sua negazione è conseguenza logica degli enunciati precedenti.

Poniamo ora attenzione al risultato ottenuto, $\mathbf{M}_n$. Se il discorso non contiene contraddizioni, $\mathbf{M}_n$ non è vuoto; abbiamo allora due possibilità:

- $\mathbf{M}_n$ contiene un solo modello (a meno di isomorfismi): in tal caso il discorso è *categorico*, nel senso che definisce la realtà in modo univoco;
- $\mathbf{M}_n$ contiene più modelli distinti (ovvero non isomorfi fra loro): in tal caso il discorso non definisce la realtà in modo univoco.

È interessante notare che qualunque frammento *finito* di realtà può essere descritto in modo categorico con un discorso del primo ordine: è noto infatti che ogni modello finito[9] (ovvero, con dominio finito, insieme **P** finito e insieme **C** finito) può essere specificato in modo categorico con un numero finito di enunciati del primo ordine[10]. Consideriamo ad esempio il seguente discorso:

Ci sono esattamente due individui; uno dei due individui è biondo; uno non lo è.

La rappresentazione più semplice possibile di questo discorso utilizza un solo predicato *B* (biondo) di arità uno. Il linguaggio è $\mathbf{\Phi}(\{B\},\varnothing)$; la classe dei modelli, $\mathbf{Mod}(\{B\},\varnothing)$, è costituita da tutti i modelli con dominio *D* non vuoto (e per il resto arbitrario) e un sottoinsieme assegnato $ext(B) \subseteq D$ (*ref* è in questo caso la funzione vuota).

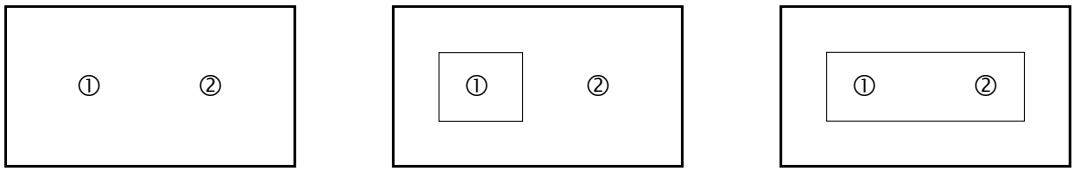
La classe iniziale dei modelli è $\mathbf{M}_0 = \mathbf{Mod}(\{B\},\varnothing)$. La prima frase del discorso si formalizza come:

$$\exists x\, \exists y\, (x \neq y \ \wedge\ \forall z\, (z = x \ \vee\ z = y)).$$

[9] Più precisamente: ogni modello *normale* finito (vedi la nota 3).

[10] Vale anche l'inversa: ogni modello normale che sia caratterizzabile in modo categorico da un insieme di enunciati del primo ordine è necessariamente finito.

La classe $\mathbf{M}_1$ è costituita da $\{M \in \mathbf{M}_0 \mid card(D) = 2\}$. Identificando le strutture fra loro isomorfe, $\mathbf{M}_1$ contiene esattamente tre modelli, rappresentabili nel modo seguente (il riquadro esterno rappresenta il dominio, il riquadro interno l'estensione di B):



M_{11} M_{12} M_{13}

La seconda frase si formalizza come

$\exists x\, B(x)$.

La classe $\mathbf{M}_2$ è costituita da $\{M \in \mathbf{M}_1 \mid ext(B) \neq \varnothing\}$. Ciò significa che il modello M_{11} viene scartato, mentre rimangono M_{12} e M_{13}. Infine, la terza frase si formalizza come

$\exists x \neg B(x)$.

La classe $\mathbf{M}_3$ è costituita da $\{M \in \mathbf{M}_2 \mid ext(B) \neq D\}$. Pertanto il modello M_{13} viene scartato, e si ha

$\mathbf{M}_3 = \{M_{12}\}$.

Come si vede il discorso che abbiamo analizzato è categorico.

3. Il calcolo

Come abbiamo già sottolineato (par. 1.6), la funzione di un calcolo è di stabilire in modo meccanico se una formula φ è conseguenza logica di un insieme Γ di formule. Come dimostrato Church (1936) e da Turing (1936), FOL è *indecidibile*: ciò significa che non esiste una procedura meccanica in grado di stabilire in un numero finito di passi se una formula è conseguenza logica di un insieme di premesse. Tuttavia FOL ammette un calcolo corretto e completo, ed è quindi *semidecidibile*: un *calcolo dei predicati del primo ordine* è appunto una procedura di semidecisione per FOL. Consultando la letteratura specializzata è facile notare che i calcoli logici possono assumere forme molto diverse: fra queste ricordiamo i calcoli "alla Hilbert", i calcoli per la deduzione naturale e i calcoli basati sui *tableaux*. Nel seguito descriviamo in dettaglio un calcolo basato sui *tableaux*.

3.4 Il calcolo degli alberi semantici

Vedremo ora una procedura di prova per la logica del primo ordine, che chiameremo *calcolo degli alberi semantici*, ispirato a un metodo proposto da Beth (1955) e noto in generale come *calcolo dei tableaux*. Per semplicità, il calcolo presentato richiede che tutte le formule utilizzate siano formule chiuse (ovvero prive di occorrenze libere di variabili); questo vincolo non è però limitativo dal punto di vista delle applicazioni.

L'idea è la seguente. Consideriamo un insieme Γ di enunciati e un ulteriore enunciato φ. Dire che

(3.1) $\Gamma \models \varphi$

equivale a dire che φ è vero in ogni modello di Γ o, in altre parole, che non esiste un modello che renda vere tutte le premesse di Γ e renda falsa la conclusione φ. Il calcolo tenta di dimostrare la 3.1 per assurdo. Ovvero: si tenta sistematicamente di costruire un modello di Γ che falsifichi φ (cioè un modello in cui tutte le premesse siano vere e la conclusione sia falsa); se il tentativo porta inevitabilmente a una contraddizione, se ne deduce che un simile modello non può esistere, e quindi che la conclusione è conseguenza logica delle premesse.

Quando il calcolo termina in un numero finito di passi portando inevitabilmente a una contraddizione scriviamo

$$\Gamma \vdash \varphi$$

e diciamo che φ *deriva da* Γ. Se poi φ deriva da un insieme vuoto di premesse scriviamo

$$\vdash \varphi$$

e diciamo che φ *è un teorema*.

Sulla falsariga della prova originale di Beth (1955) è possibile dimostrare che il calcolo è corretto e completo, anche nel caso in cui Γ contenga infinite premesse. Naturalmente la completezza non garantisce la decidibilità: infatti se φ non è conseguenza logica di Γ, la procedura può terminare fornendo la descrizione di un *controesempio* (ovvero di un modello di Γ che falsifica φ) oppure *non terminare*. Tuttavia, il calcolo degli alberi semantici si comporta come un algoritmo di decisione (in quanto termina sempre) se gli enunciati presi in considerazione sono tutti *proposizionali* (ovvero non contengono variabili né quantificatori).

Esempi introduttivi

Come si è già detto, il calcolo degli alberi semantici tenta di dimostrare che la conclusione, φ, può essere falsa anche se le premesse, Γ, sono tutte vere. A questo scopo si costruisce un albero, detto *albero semantico*; l'albero è costituito da un certo numero di *nodi*, ognuno dei quali è una struttura con quattro campi, che scriveremo su un'unica riga. I campi sono nell'ordine:

- il numero d'identificazione del nodo;
- un enunciato (ovvero, una formula chiusa)
- una giustificazione;
- una lista di *costanti introdotte* (eventualmente vuota).

Ad esempio, supponiamo di voler verificare se

$$P \rightarrow Q, P \models Q.$$

Iniziamo con i primi tre nodi dell'albero:

linea	*enunciato*	*giustificazione*	*costanti*
1.	$P \rightarrow Q$	premessa	
2.	P	premessa	
3.	$\neg Q$	$\neg$conclusione	

Ovvero:

- il nodo numero 1 contiene l'enunciato $P \rightarrow Q$, assunto come premessa;
- il nodo numero 2 contiene l'enunciato P, anch'esso assunto come premessa;
- il nodo numero 3 contiene l'enunciato $\neg Q$, assunto come negazione della conclusione, coerentemente con la strategia di dimostrare la conseguenza logica per assurdo.

La dimostrazione prosegue cercando di dimostrare che esiste un modello M che può soddisfare gli enunciati dei nodi 1, 2 e 3; ovvero, che esiste un modello M che verifica le premesse e falsifica la conclusione. Il calcolo utilizza 14 *regole di deduzione*: 7 per la negazione, 6 per le altre costanti logiche (congiunzione, disgiunzione, condizionale, bicondizionale, quantificatore universale, quantificatore esistenziale) e 1 per l'introduzione del *falsum* (che rappresenta l'esistenza di una contraddizione).

La deduzione deduzione procede in questo modo:

- per ogni regola applicabile a un enunciato viene generata un'*attivazione*, che a sua volta viene inserita in un'*agenda*;

- se l'agenda è vuota il processo di deduzione *termina*;
- se l'agenda invece contiene delle attivazioni, se ne sceglie una in base a criteri che vedremo in seguito, la si rimuove dall'agenda e la si *esegue*.

Nel caso dell'esempio l'unica regola applicabile può essere descritta nel modo seguente:

$$\rightarrow: \quad \frac{\varphi \rightarrow \psi}{\neg\varphi \mid \psi}$$

La regola si legge così:

- il *nome* della regola ($\rightarrow$) coincide con la costante logica trattata dalla regola;
- la *condizione* di applicazione della regola (al di sopra della linea orizzontale) identifica la forma generale degli enunciati che possono *attivare* la regola;
- l'*azione* (al di sotto della linea orizzontale) indica ciò che va fatto quando un'attivazione della regola viene eseguita.

Nel nostro caso, la barra verticale che compare nell'azione $\neg\varphi \mid \psi$ indica che nell'albero della dimostrazione deve essere introdotta una *biforcazione*; nel ramo sinistro viene inserito un nodo con l'enunciato $\neg\varphi$, nel ramo destro un nodo con l'enunciato ψ. Quest'azione è giustificata dal fatto che se il modello M che stiamo cercando di costruire verifica $\varphi \rightarrow \psi$, allora in M dev'essere vera $\neg\varphi$ oppure vera ψ.

La regola $\rightarrow$ è applicabile all'enunciato del nodo 1; viene quindi generata un'attivazione della regola, che descriviamo come

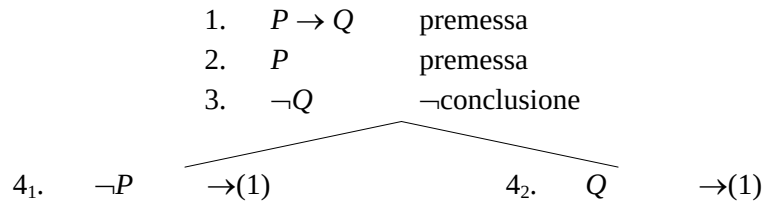
$\rightarrow(1)$

ovvero: la regola denominata '$\rightarrow$' e applicata al nodo 1; l'attivazione viene inserita nell'agenda, inizialmente vuota, che diventa quindi

$\{\rightarrow(1)\}$.

Nessun'altra regola è applicabile ai nodi 1, 2 e 3. A questo punto dobbiamo scegliere un'attivazione da eseguire. Dato che l'agenda contiene solo $\rightarrow(1)$, quest'attivazione viene scelta, rimossa dall'agenda ed eseguita. Eseguire l'attivazione significa effettuare l'azione della regola corrispondente.

L'albero diventa quindi:



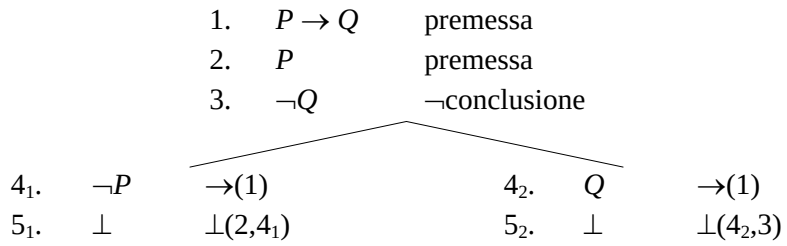
Ora consideriamo la regola, che consente di introdurre il *falsum* ($\perp$), che indica la presenza di una contraddizione in un ramo dell'albero. La regola ha due condizioni (φ e $\neg\varphi$) e un'azione:

$$\perp: \quad \frac{\varphi\,,\, \neg\varphi}{\perp}$$

Sull'albero mostrato sopra, questa regola genera due attivazioni: $\perp(2,4_1)$ nel ramo sinistro e $\perp(4_2,3)$ nel ramo destro. L'agenda è quindi

$\{\bot(2,4_1), \bot(4_2,3)\}$.

Dopo aver eseguito entrambe le attivazioni abbiamo:



Ai nostri fini, ogni ramo che contiene il *falsum* può essere scartato perché nessun modello *M* può rendere vera una contraddizione; in altre parole, ogni ramo dell'albero che contiene il falsum indica un *tentativo fallito* di costruire un modello che renda vere le premesse e falsa la conclusione. Nel nostro caso soltanto due tentativi erano giustificati (i due rami dell'albero), e sono ambuedue falliti; ne segue che non esiste alcun modello che renda le due premesse vere e la conclusione falsa. Ma ciò significa che la conclusione è conseguenza logica delle premesse. Abbiamo quindi dimostrato che:

$P \to Q, P \vdash Q$.

Poiché, come abbiamo già detto, il calcolo degli alberi semantici è *corretto* ne ricaviamo che

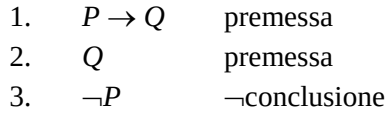
$P \to Q, P \models Q$.

Occorre chiarire un punto essenziale. Nell'esecuzione manuale della procedura ci siamo abbondantemente basati sulla nostra intuizione di che cosa significhi che un enunciato è vero o falso, e di che cosa siano un condizionale e una contraddizione. Tuttavia, il metodo è completamente meccanico, perché può essere eseguito da un calcolatore come puro processo di manipolazione di espressioni simboliche, senza la minima comprensione del significato dei simboli.

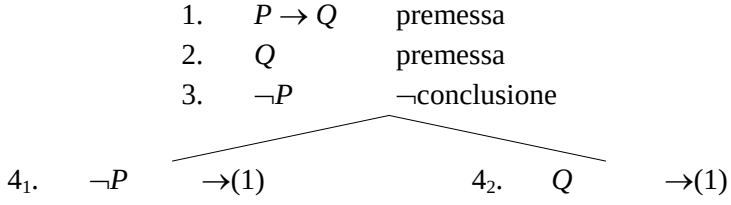
Come secondo esempio, proviamo a verificare se

$P \to Q, Q \models P$.

Iniziamo da



Proseguendo come prima arriviamo a



Ora però non è possibile introdurre il *falsum* in nessuno dei due rami; poiché nessuna regola nuova è applicabile all'albero, l'agenda delle attivazioni è vuota e la procedura termina con un insuccesso.

Ogni *ramo* dell'albero che dopo la terminazione non contenga il *falsum* ci descrive un possibile controesempio all'ipotesi iniziale (ovvero, un modello che rende vere le premesse e falsa la

conclusione). Nel nostro caso sia il ramo 1 (costituito dai nodi 1, 2, 3, 4_1), sia il ramo 2 (costituito dai nodi 1, 2, 3, 4_2) descrivono lo stesso controesempio: un modello che renda P falso ($\neg P$) e Q vero (Q). In conclusione abbiamo

$P \rightarrow Q, Q \nvdash P$.

Dato che, come abbiamo già detto, il calcolo degli alberi semantici è *completo* ne ricaviamo che

$P \rightarrow Q, Q \not\models P$.

Lo stesso metodo si applica in assenza di premesse. Ad esempio, verifichiamo se

$\models P \wedge Q \rightarrow P$.

Iniziamo con:

1.	$\neg(P \wedge Q \rightarrow P)$	$\neg$conclusione

Possiamo applicare una delle regole della negazione, e più precisamente:

$$\neg\rightarrow: \quad \frac{\neg(\varphi \rightarrow \psi)}{\varphi\,,\ \neg\psi}$$

In questo caso non ci sarà alcuna biforcazione, perché la virgola nell'azione ($\varphi\,,\ \neg\psi$) indica che vanno aggiunti due nodi allo *stesso* ramo dell'albero:

1.	$\neg(P \wedge Q \rightarrow P)$	$\neg$conclusione
2.	$P \wedge Q$	$\neg\rightarrow$(1)
3.	$\neg P$	$\neg\rightarrow$(1)

Ora possiamo applicare la regola della congiunzione:

$$\wedge: \quad \frac{\varphi \wedge \psi}{\varphi\,,\ \psi}$$

Anche questa regola non introduce biforcazioni:

1.	$\neg(P \wedge Q \rightarrow P)$	$\neg$conclusione
2.	$P \wedge Q$	$\neg\rightarrow$(1)
3.	$\neg P$	$\neg\rightarrow$(1)
4.	P	$\wedge$(2)
5.	Q	$\wedge$(2)

Ora è applicabile la regola per l'introduzione del *falsum*:

1.	$\neg(P \wedge Q \rightarrow P)$	$\neg$conclusione
2.	$P \wedge Q$	$\neg\rightarrow$(1)
3.	$\neg P$	$\neg\rightarrow$(1)
4.	P	$\wedge$(2)
5.	Q	$\wedge$(2)
6.	$\bot$	$\bot$(4,3)

Poiché l'unico ramo dell'albero fallisce, ne ricaviamo che

$$\vdash P \wedge Q \rightarrow P.$$

Le regole per i connettivi booleani

Di seguito sono elencate le nove regole per i connettivi booleani; il lettore è invitato a verificarne la correttezza in base alla semantica formale del linguaggio logico. Quattro regole riguardano i connettivi booleani binari:

$$\wedge: \quad \frac{\varphi \wedge \psi}{\varphi \,,\, \psi} \qquad\qquad \vee: \quad \frac{\varphi \vee \psi}{\varphi \mid \psi}$$

$$\rightarrow: \quad \frac{\varphi \rightarrow \psi}{\neg\varphi \mid \psi} \qquad\qquad \leftrightarrow: \quad \frac{\varphi \leftrightarrow \psi}{\varphi \,,\, \psi \mid \neg\varphi \,,\, \neg\psi}$$

Cinque regole concernono i cinque connettivi booleani quando compaiono nell'ambito di una negazione:

$$\neg\neg: \quad \frac{\neg\neg\varphi}{\varphi}$$

$$\neg\wedge: \quad \frac{\neg(\varphi \wedge \psi)}{\neg\varphi \mid \neg\psi} \qquad\qquad \neg\vee: \quad \frac{\neg(\varphi \vee \psi)}{\neg\varphi \,,\, \neg\psi}$$

$$\neg\rightarrow: \quad \frac{\neg(\varphi \rightarrow \psi)}{\varphi \,,\, \neg\psi} \qquad\qquad \neg\leftrightarrow: \quad \frac{\neg(\varphi \leftrightarrow \psi)}{\varphi \,,\, \neg\psi \mid \neg\varphi \,,\, \psi}$$

A queste va aggiunta la regola per l'introduzione del *falsum*:

$$\bot: \quad \frac{\varphi \,,\, \neg\varphi}{\bot}$$

Se ci si limita a considerare insiemi finiti di enunciati proposizionali è facile mostrare che il calcolo termina sempre; anzi, adattando una dimostrazione di Beth (1955) si può mostrare che la terminazione è assicurata anche nel caso di insiemi numerabili di enunciati proposizionali. Ciò non è più vero se utilizziamo i quantificatori.

Regole per i quantificatori

Il calcolo prevede quatto regole per la manipolazione dei quantificatori, di cui due relative alla negazione. Cominceremo con alcuni esempi. Nel seguito si fa l'ipotesi che il linguaggio offra un'infinità numerabile di costanti individuali, che chiameremo *a*, *b*, *c*, ...

Come primo esempio ci chiediamo se:

$$\forall x Px \models Pa.$$

I primi due nodi dell'albero sono:

1.	$\forall xPx$	premessa	
2.	$\neg Pa$	$\neg$conclusione	a

Nell'ultimo campo del nodo abbiamo messo in evidenza che l'enunciato introduce una costante, a. Ora, il fatto che $\forall xPx$ sia vero in M ci autorizza ad aggiungere un enunciato atomica Pc, dove c è una costante qualsiasi. D'altra parte, non dobbiamo dimenticarci che il nostro obiettivo è generare una contraddizione. Sarebbe inutile, quindi, aggiungere a un ramo dell'albero un enunciato della forma Pc se la costante c non compare altrove nel ramo: un enunciato del genere non potrebbe comunque dar luogo a una contraddizione. Possiamo quindi formulare la seguente regola per il quantificatore universale:

$$\forall: \quad \frac{\forall x\varphi}{\varphi[c/x]}$$

dove c è una costante che compare nel ramo dell'albero cui viene aggiunto il nodo

Eseguiamo l'azione con la costante a:

1.	$\forall xPx$	premessa	
2.	$\neg Pa$	$\neg$conclusione	a
3.	Pa	$\forall(1/a)$	

Ora possiamo introdurre il *falsum*:

1.	$\forall xPx$	premessa	
2.	$\neg Pa$	$\neg$conclusione	a
3.	Pa	$\forall(1/a)$	
4.	$\bot$	$\bot(3,2)$	

Pertanto:

$\forall xPx \vdash Pa_0$.

Come si vede, un enunciato del tipo $\forall x\varphi$ si comporta come un *generatore di enunciati* della forma $\varphi[a_n/x]$, per ogni costante a_n introdotta. Va precisato subito che la lista delle costanti introdotte non è fisso: oltre alle costanti presenti inizialmente, altre costanti possono essere aggiunte dalla regola per il quantificatore esistenziale (vedi oltre). Il generatore $\forall x\varphi$ si deve quindi "risvegliare" ogni volta che viene introdotta una nuova costante, per poter generare nuovi enunciati della forma $\varphi[a_n/x]$.

Proviamo ora a verificare se

$Pa \models \exists xPx$.

I primi due nodi dell'albero sono:

1.	Pa	premessa	a
2.	$\neg\exists xPx$	$\neg$conclusione	

La regola per la negazione del quantificatore esistenziale tiene conto del fatto che $\neg\exists x\varphi$ è equivalente a $\forall x\neg P\varphi$:

$$\neg\exists: \quad \frac{\neg\exists x\varphi}{\neg\varphi[c/x]}$$

dove c è una costante che compare nel ramo dell'albero cui viene aggiunto il nodo

Eseguiamo l'azione e in seguito applichiamo la regola del *falsum*:

1.	Pa_0	premessa	a
2.	$\neg\exists xPx$	¬conclusione	
3.	$\neg Pa$	$\neg\exists(2/a)$	
4.	$\bot$	$\bot(1,2)$	

Quindi:

$Pa_0 \vdash \exists xPx$.

Proviamo ora a verificare se vale

$\exists xPx \models Pa$.

I primi due nodi dell'albero sono:

1.	$\exists xPx$	premessa	
2.	$\neg Pa$	¬conclusione	a

A questo punto ragioniamo nel modo seguente. Se $\exists xPx$ è vero in M, esisterà un individuo nel dominio, chiamiamolo genericamente c, per cui si ha Pc. Dato che nulla ci consente di assumere che tale individuo sia uno di quelli cui fanno riferimento le costanti già introdotte, dobbiamo introdurre *una nuova costante*. Ecco quindi la regola per il quantificatore esistenziale:

$$\exists: \quad \frac{\exists x\varphi}{\varphi[c/x]}$$

dove c è una costante che non compare nel ramo dell'albero cui viene aggiunto il nodo

Come si vede, un enunciato del tipo $\exists x\varphi$ si comporta come *generatore di una nuova costante*. Al contrario della regola per il quantificatore universale, la regola una volta applicata non ha più bisogno di essere "risvegliata".

Applichiamo la regola:

1.	$\exists xPx$	premessa	
2.	$\neg Pa$	¬conclusione	a
3.	Pb	$\exists(1)$	b

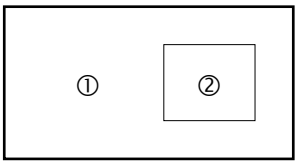
Non ci sono altre regole applicabili, per cui la procedura termina senza generare il *falsum*. Abbiamo quindi:

$\exists xPx \nvdash Pa$.

Tenendo conto della completezza del calcolo si ha che:

$\exists xPx \not\models Pa$.

Dall'albero è possibile ricavare un controesempio: introduciamo un dominio con due individui, ① e ②, da utilizzare come i referenti rispettivi di *a* e *b*; poi definiamo il modello fissando l'estensione di *P* in modo tale che *P* sia falso di *a* (il che soddisfa $\neg Pa$) e vero di *b* (il che soddisfa Pb)[11]. Graficamente:



Proviamo ora con

$Pa \vdash \forall xPx$.

I primi due nodi dell'albero sono:

1.	Pa	premessa	*a*
2.	$\neg\forall xPx$	¬conclusione	

La regola per la negazione del quantificatore universale tiene conto del fatto che $\neg\forall x\varphi$ è equivalente a $\exists x\neg P\varphi$:

$$\neg\forall: \quad \frac{\neg\forall x\varphi}{\neg\varphi[c/x]}$$ dove *c* è una costante che non compare nel ramo dell'albero cui viene aggiunto il nodo

Applichiamo la regola:

1.	Pa	premessa	*a*
2.	$\neg\forall xPx$	¬conclusione	
3.	$\neg Pb$	¬∀(2)	*b*

Non ci sono altre regole applicabili, per cui la procedura termina senza generare il *falsum*. Abbiamo quindi:

$Pa_0 \nvdash \forall xPx$.

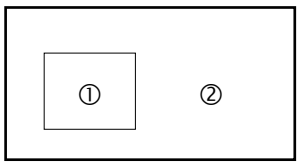
Tenendo conto della completezza del calcolo si ha che:

$Pa_0 \not\models \forall xPx$.

Anche qui, dall'albero è possibile ricavare un controesempio: introduciamo un dominio con due individui, ① e ②, da utilizzare come i referenti rispettivi di a_0 e a_1; poi definiamo il modello fissando l'estensione di *P* in modo tale che *P* sia vero di *a* (il che soddisfa Pa) e falso di *b* (il che soddisfa $\neg Pb$).

[11] Si noti che non sarebbe possibile costruire un controesempio su un dominio con meno di due individui.

Graficamente:



Resta da precisare un ultimo punto: può capitare di dover applicare la regola $\forall$ o la regola $\neg\exists$ a un nodo di un ramo in cui non è stata introdotta alcuna costante. In tal caso, applichiamo la regola generando una costante; ciò è giustificato dal fatto il dominio del modello deve comunque contenere almeno un individuo, cui possiamo fare riferimento con la costante introdotta. Verifichiamo ora la validità di:

$\forall xPx \models \exists xPx$.

Ecco la dimostrazione completa:

1.	$\forall xPx$	premessa	
2.	$\neg\exists xPx$	$\neg$conclusione	
3.	Pa	$\forall$(1)	a
4.	$\neg Pa$	$\neg\exists$(2/a)	
5.	$\bot$	$\bot$(3,4)	

Quindi $\forall xPx \vdash \exists xPx$ e pertanto, per la correttezza del calcolo,

$\forall xPx \models \exists xPx$.

Vediamo ancora due esempi. Verifichiamo che

$\exists x\forall yPxy \models \forall y\exists xPxy$.

Ecco la dimostrazione completa:

1.	$\exists x\forall yPxy$	premessa	
2.	$\neg\forall y\exists xPxy$	$\neg$conclusione	
3.	$\forall yPay$	$\exists$(1)	a
4.	$\neg\exists xPxb$	$\neg\forall$(2)	b
6.	Pab	$\forall$(3/b)	
7.	$\neg Pab$	$\neg\exists$(4/a)	
8.	$\bot$	$\bot$(6,7)	

Va notato che avremmo potuto eseguire anche altre due attivazioni di regole: $\forall$(3/a) e $\neg\exists$(4/a). I nodi generati non avrebbero però fornito informazioni utili, dato che il conflitto fra i nodi 6 e 7 è comunque sufficiente a introdurre un fallimento nell'unico ramo dell'albero.

Veniamo ora all'ultimo esempio e verifichiamo se:

$\forall y\exists xPxy \models \exists x\forall yPxy$.

Ecco i primi passi della dimostrazione:

1.	$\forall y \exists x Pxy$	premessa	
2.	$\neg \exists x \forall y Pxy$	$\neg$conclusione	
3.	$\exists x Pxa$	$\forall(1)$	a
4.	$\neg \forall y Pay$	$\neg\exists(2/a)$	
5.	Pab	$\exists(3)$	b
6.	$\neg Pac$	$\neg\forall(4)$	c

A questo punto la dimostrazione *non* è finita: infatti è possibile applicare la regola $\forall$ a $(1/a)$ e a $(1/a_2)$, e la regola $\neg\exists$ a $(2/a_1)$ e a $(2/a_2)$; l'esecuzione di queste regole introduce nuove formule, cui si può applicare nuovamente la regola $\neg\forall$:

7.	$\exists x Pxb$	$\forall(1/b)$	
8.	$\exists x Pxc$	$\forall(1/c)$	
9.	$\neg \forall y Pby$	$\neg\exists(2/b)$	
10.	$\neg \forall y Pcy$	$\neg\exists(2/c)$	
11.	Pdb	$\exists(7)$	d
12.	Pec	$\exists(8)$	e
13.	$\neg Pbf$	$\neg\forall(9)$	f
14.	$\neg Pcg$	$\neg\forall(10)$	g

È facile convincersi che il procedimento è entrato in un ciclo senza fine: dopo l'introduzione della prima costante (a), possiamo vedere la costruzione dell'albero come una sequenza infinita di fasi di elaborazione, tali che nell'n-esima fase vengono introdotte 2^n nuove costanti (nell'esempio, la fase 2 va dal nodo 7 al nodo 14). In effetti, quando il calcolo viene applicato ad argomentazioni o enunciati predicativi non validi è possibile che la procedura "vada in ciclo" e quindi non termina: ciò non è strano, dato che il calcolo non è un algoritmo di decisione. D'altra parte, è interessante vedere *perché* la procedura va in ciclo: ciò è dovuto a un circolo vizioso che si crea fra le regole $\exists$ e $\neg\forall$, che introducono costanti nuove, e le regole $\forall$ e $\neg\exists$, che si risvegliano ogni volta che viene introdotta una nuova costante.

Le regole per l'uguaglianza

Il calcolo degli alberi semantici può essere esteso al trattamento dell'uguaglianza. Per questo è sufficiente aggiungere due regole. La prima regola, detta *regola di riflessività*, consente di introdurre un enunciato del tipo $c = c$ per ogni costante c:

$$\text{R=:}\quad \frac{}{c = c}$$

dove c è una costante che compare nel ramo dell'albero cui viene aggiunto il nodo

Si noti che la condizione della regola R= è vuota. La seconda regola, detta *regola di sostituzione*, consente di sostituire ovunque lo si desideri una costante c' a una costante c quando $c' = c$:

$$\text{S=:}\quad \frac{c' = c\ ,\ \varphi}{\varphi(c'/c)}$$

Attenzione: mentre la scrittura $\varphi[c'/c]$ significa che *ogni* occorrenza di c va rimpiazzata con un'occorrenza di c', la scrittura $\varphi(c'/c)$ significa che c' rimpiazza *un numero arbitrario* di occorrenze di c (anche zero).

Utilizziamo il calcolo così esteso per dimostrare che:

$Pa, a = b \models Pb$,

$Pa, \neg Pb \models a \neq b$.

Ecco le due dimostrazioni:

1.	Pa	premessa
2.	$a = b$	premessa
3.	$\neg Pb$	¬conclusione
4.	$\neg Pa$	S=(2,3)
5.	$\bot$	⊥(1,4)

1.	Pa	premessa
2.	$\neg Pb$	premessa
3.	$\neg\neg(a = b)$	¬conclusione
4.	$a = b$	¬¬(3)
5.	$\neg Pa$	S=(4,2)
6.	$\bot$	⊥(1,5)

Dimostriamo ora le tre proprietà fondamentali dell'uguaglianza. Proprietà riflessiva:

$\models \forall x\,(x = x)$.

Dimostrazione:

1.	$\neg\forall x\,(x = x)$	¬conclusione	
2.	$\neg(a = a)$	¬∀:1	a
3.	$a = a$	R=(/a)	
4.	$\bot$	⊥(3,2)	

Proprietà simmetrica:

$\models \forall x\,\forall y\,(x = y \rightarrow y = x)$.

Dimostrazione:

1.	$\neg\forall x\forall y\,(x = y \rightarrow y = x)$	$\neg$conclusione	
2.	$\neg\forall y\,(a = y \rightarrow y = a)$	$\neg\forall(1)$	a
3.	$\neg(a = b \rightarrow b = a)$	$\neg\forall(2)$	b
4.	$a = b$	$\neg\rightarrow(3)$	
5.	$\neg(b = a)$	$\neg\rightarrow(3)$	
6.	$\neg(a = a)$	S=(4,5)	
7.	$a = a$	R=($/a_0$)	
8.	$\perp$	$\perp$(7,6)	

Proprietà transitiva:

$\models\ \forall x\ \forall y\ \forall z\ (x = y \wedge y = z \rightarrow x = z)$.

Dimostrazione:

1.	$\neg\forall x\forall y\forall z\,(x = y \wedge y = z \rightarrow x = z)$	$\neg$conclusione	
2.	$\neg\forall y\forall z\,(a = y \wedge y = z \rightarrow a = z)$	$\neg\forall(1)$	a
3.	$\neg\forall z\,(a = b \wedge b = z \rightarrow a = z)$	$\neg\forall(2)$	b
4.	$\neg(a = b \wedge b = c \rightarrow a = c)$	$\neg\forall(2)$	c
5.	$a = b \wedge b = c$	$\neg\rightarrow(4)$	
6.	$\neg(a = c)$	$\neg\rightarrow(4)$	
7.	$a = b$	$\wedge(5)$	
8.	$b = c$	$\wedge(5)$	
9.	$\neg(a = b)$	S=(8,6)	
10.	$\perp$	$\perp$(7,9)	

Criteri di gestione dell'agenda

A un passo generico della costruzione di un albero semantico sono spesso presenti nell'agenda diverse attivazioni. Se la scelta della prossima attivazione da eseguire non è oculata, è possibile che l'albero costruito non sia ottimale (ovvero, che presenti più nodi e più biforcazioni del necessario). Non è possibile formulare un insieme di criteri che assicurino in ogni caso la costruzione di un albero ottimale; tuttavia è possibile formulare alcuni *criteri euristici* (ovvero criteri che in genere portano a produrre un albero ottimale o quasi ottimale). I criteri più importanti sono:

- dare priorità alle attivazioni delle regole che introducono il *falsum* (perché permetteno di chiudere un ramo dell'albero);
- in subordine, preferire le attivazioni di regole che non producono biforcazioni (così l'albero si mantiene più semplice);
- preferire le attivazioni delle regole che producono nuove costanti a quelle che sfruttano costanti già esistenti (perché è meglio generare subito le costanti da utilizzare in seguito per altre inferenze);
- preferire le attivazioni applicate a enunciati che hanno più predicati in comune con altri enunciati del ramo corrente (in questo modo si massimizzano le opportunità di creare contraddizioni e derivare il *falsum*).

3.5 Il teorema della deduzione

Ci si può chiedere che differenza ci sia fra dimostrare che $\forall xP(x) \vdash P(a)$ e dimostrare che $\vdash \forall xP(x) \rightarrow P(a)$. Dal *teorema della conseguenza logica* (par. 2.4), dato che il calcolo è corretto e completo, ricaviamo immediatamente il *teorema della deduzione*:

$\Gamma \vdash \varphi \rightarrow \psi$ se e solo se $\Gamma, \varphi \vdash \psi$.

Nel caso in cui Γ sia finito è facile dimostrare il teorema della deduzione direttamente. Infatti, una dimostrazione di $\Gamma, \varphi \vdash \psi$ con $\Gamma = \{\gamma_1, ..., \gamma_n\}$ avrà la forma

1.	γ_1	premessa
...	...	...
n.	γ_n	premessa
n+1.	φ	premessa
n+2.	$\neg\psi$	¬conclusione
...	...	...

Da questa dimostrazione aggiungendo un nodo (*n'*) e modificando le giustificazioni dei nodi *n*+1 ed *n*+2 si può ricavare una dimostrazione di $\Gamma \vdash \varphi \rightarrow \psi$:

1.	γ_1	premessa
...	...	...
n.	γ_n	premessa
n'.	$\neg(\varphi \rightarrow \psi)$	¬conclusione
n+1.	φ	→(*n'*)
n+2.	$\neg\psi$	→(*n'*)
...	...	...

L'operazione inversa consente di passare da una prova di $\Gamma \vdash \varphi \rightarrow \psi$ a una prova di $\Gamma, \varphi \vdash \psi$.

3.6 Regole derivate

È possibile rendere le dimostrazioni più brevi introducendo ulteriori regole deduttive, dette *regole derivate*, che non aumentano la potenza del calcolo ma consentono di riutilizzare dimostrazioni già eseguite. Consideriamo ad esempio la derivazione

$\exists x\forall yP(x,y) \vdash \forall y\exists xP(x,y)$,

vista precedentemente. È facile rendersi conto che questa derivazione può essere generalizzata come:

$\exists x\forall y\varphi \vdash \forall y\exists x\varphi$.

Ecco la dimostrazione:

1.	$\exists x\forall y\varphi$	premessa	
2.	$\neg\forall y\exists x\varphi$	¬conclusione	
3.	$\forall y\varphi[a/x]$	∃(1)	*a*
4.	$\neg\exists x\varphi[b/y]$	¬∀(2)	*b*
6.	$\varphi[a/x,b/y]$	∀(3/*b*)	
7.	$\neg\varphi[a/x,b/y]$	¬∃(4/*a*)	
8.	$\perp$	⊥(6,7)	

A questo punto, se l'enunciato $\exists x \forall y \varphi$ compare in una dimostrazione, possiamo senz'altro aggiungere l'enunciato $\forall y \exists x \psi$, perché abbiamo già dimostrato una volta per tutte che il secondo enunciato è vero quando il primo enunciato è vero. Possiamo quindi formulare la regola derivata:

$$\exists\forall\text{–}\forall\exists\text{:} \quad \frac{\exists x \forall y \varphi}{\forall y \exists x \psi}$$

Utilizzando questa regola possiamo rendere più concise molte dimostrazioni.

Ogni teorema già dimostrato si può convertire in una regola derivata. Anche se l'uso di un numero eccessivo di regole derivate può avere effetti negativi sull'efficienza di un dimostratore automatico di teoremi, alcune regole derivate sono molto utili. Nel seguito ne introduciamo due; la dimostrazione per esteso della loro validità è lasciata al lettore.

La due regole che consideriamo qui, dette rispettivamente *modus pones* e *modus tollens*, concernono il connettivo condizionale:

$$\text{MP:} \quad \frac{\varphi \rightarrow \psi\,,\ \varphi}{\psi} \qquad\qquad \text{MT:} \quad \frac{\varphi \rightarrow \psi\,,\ \neg\psi}{\neg\varphi}$$

Regole analoghe consentono di evitare biforcazioni relative alla disgiunzione e alla negazione della congiunzione:

$$\vee_2\text{:} \quad \frac{\varphi \vee \psi\,,\ \neg\varphi}{\psi} \qquad\qquad \vee_3\text{:} \quad \frac{\varphi \vee \psi\,,\ \neg\psi}{\varphi}$$

$$\neg\wedge_2\text{:} \quad \frac{\neg(\varphi \wedge \psi),\ \varphi}{\neg\psi} \qquad\qquad \neg\wedge_3\text{:} \quad \frac{\neg(\varphi \wedge \psi),\ \psi}{\neg\varphi}$$

Un esempio di utilizzo di regole derivate è dato nel prossimo sottoparagrafo.

3.7 Verifica della validità di un'argomentazione

A questo punto possiamo ritornare all'idea iniziale di stabilire la validità di un'argomentazione. Consideriamo ad esempio l'argomentazione seguente:

Chi non beve in compagnia, o è un ladro o è una spia.
Alberto non beve in compagnia.
Alberto non è una spia.
Quindi Alberto è un ladro.

Il primo passo consiste nel rappresentare la forma logica dell'argomentazione. Ogni enunciato italiano viene tradotto in un enunciato logico-predicativo cercando di rendere il significato dell'enunciato originario nel modo più fedele possibile. A questo scopo introduciamo tre predicati,

Bx *x beve in compagnia*,
Lx *x è un ladro*,
Sx *x è una spia*,

e una costante individuale,

a *Alberto*.

L'argomentazione si formalizza allora così:

$\forall x\,(\neg Bx \rightarrow Lx \vee Sx)$

$\neg Ba$

$\neg Sa$

$\therefore\ La$

L'argomentazione è valida se e solo se:

$\forall x\,(\neg Bx \rightarrow Lx \vee Sx),\ \neg Ba,\ \neg Sa \models La.$

Ecco una prova che sfrutta alcune regole derivate:

1.	$\forall x\,(\neg Bx \rightarrow Lx \vee Sx)$	premessa	
2.	$\neg Ba$	premessa	a
3.	$\neg Sa$	premessa	
4.	$\neg La$	¬conclusione	
5.	$\neg Ba \rightarrow La \vee Sa$	$\forall(1/a)$	
6.	$La \vee Sa$	MP(5,2)	
7.	Sa	$\vee_2$(6,4)	
8.	$\perp$	$\perp$(7,3)	

Riferimenti bibliografici

Bencivenga, E., 1984. *Il primo libro di logica*, Bollati Boringhieri, Torino.

Beth, E.W., 1955. Semantic entailment and formal derivability. *Mededelingen van de Koninklijke Nederlandse Akademie van Wetenschappen, Afdeling Letterkunde*, 18 (13), 309–342. Ristampato in J. Hintikka, ed., 1969, *The philosophy of mathematics*, Oxford University Press, Oxford , 9–41.

Boole, G., 1854. *An investigation of the laws of thought, on which are founded the mathematical theories of logic and probabilities*, MacMillan & Co., Cambridge (scaricabile dal sito di *Project Gutenberg*, http://www.gutenberg.org/etext/15114).

Church, A., 1936. A Note on the Entscheidungsproblem. *Journal of Symbolic Logic*, 1, 40–41.

Frege, G., 1879. *Begriffsschrift – Eine der arithmetischen nachgebildete Formelsprache des reinen Denkens*, Nebert, Halle an der Saale.

Gödel, K., 1930. Die Vollständigkeit der Axiome des logischen Funkionenkalküls. *Monatshefte für Mathematik und Physik*, 37, 349–360.

Hilbert, D., and W. Ackermann, 1928. *Grundzüge der theoretischen Logik*, Springer, Berlin.

Leibniz, G.W., 1666. *Dissertatio de arte combinatoria*, Leipzig.

Mangione, C. e Bozzi, S., 1993. *Storia della logica*, Garzanti, Milano.

Turing, A.M., 1936. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society*, Series 2, 42, 230–265.