Logic for Computer Science

A.A. 2024-2025

Docente: M. Benerecetti

Dispense tratte dal corso Magistrale di Informatica
a cura dello studente **S. Cerrone**
Si ringrazia **R. Diana** per il generoso aiuto

N.B.: Le dimostrazioni in grigio **non** sono state svolte dal professore.

1. Introduzione alla Logica

Elementi di Logica

Un'**argomentazione** (o ragionamento) è una sequenza di **asserzioni** (affermazioni) dove si identificano delle premesse e si giunge ad una conclusione che è diretta conseguenza delle premesse.

1) Tutti gli uomini sono mortali [Premessa 1]
2) Socrate è un uomo [Premessa 2]
   Quindi
3) Socrate è mortale [Conclusione]

Questo schema è anche conosciuto in logica come sillogismo.
L'argomentazione è un processo di manipolazione simbolica che genera le asserzioni in modo da preservare alcune proprietà come quella di conseguenza logica.

In generale:

- Un'asserzione è una frase composta da premesse e conclusioni
- Un'argomentazione è **ben formata** o **valida** se la conclusione segue logicamente dalle premesse indipendentemente che le premesse siano vere o false. Se le premesse sono anche vere (e di conseguenza lo sarà anche la conclusione) l'argomentazione sarà anche **corretta**.
- Un sillogismo è una forma di argomentazione logica basata su preposizioni connesse in modo tale da dedurre una conclusione.

Ragionamento

Ragionamento Deduttivo

Un esempio di ragionamento deduttivo è il seguente

1) Tutti gli uomini sono mortali
2) Socrate è un uomo
   Quindi
3) Socrate è mortale

Il ragionamento deduttivo non aggiunge nessuna informazione che non sia già nelle premesse. Si parla infatti di conseguenze; la sua verità è implicita nelle premesse. Quindi i ragionamenti non deduttivi sono necessari per l'evoluzione.

Nota: il fatto che la conseguenza abbia informazione nulla non significa che sia inutile (potrebbero benissimo essere proprietà implicite che non si conoscevano).

Ragionamento Abduttivo

Nel ragionamento abduttivo le conclusioni sono dettate da un determinato livello di probabilità, cercando di estrarre ulteriori informazioni. In altre parole, le conclusioni non sono implicite nei fatti. Questo ragionamento parte da un risultato osservato e cerca di identificare la spiegazione più plausibile. Infatti, questo tipo di ragionamento è utile quando ci sono molte spiegazioni possibili per un dato fenomeno e si cerca la spiegazione più logica o probabile.

1) Gran parte degli studenti ama la logica
2) Ciro odia la logica

Quindi

3) Ciro non è uno studente

La conclusione, quindi, non è implicita nei fatti delle premesse, non è detto che sia vero ma è comunque un ragionamento accettato poiché la conclusione è **probabilmente** vera.

Ragionamento Induttivo

Il ragionamento induttivo, come quello abduttivo, cerca di trarre delle conclusioni generali non direttamente collegabili alle premesse (cioè le conclusioni non sono implicite nei fatti). In questo specifico caso, le conclusioni sono ottenute basandosi sull'esperienza, cioè se su un certo numero di casi otteniamo un certo comportamento, allora probabilmente tutti i casi simili si comporteranno allo stesso modo.

1) Tutti i bar che conosco a Napoli servono un caffè decente
2) X è un nuovo bar di Napoli

   Quindi

3) X servirà un caffè decente

Si noti l'assenza del passo induttivo; quindi, tale ragionamento differisce dal principio induttivo. Proprio per questo, prende il nome di *induzione malformata*.

Concetti fondamentali

Il concetto di consistenza, inconsistenza e validità può essere applicato sia ad un asserto che ad un insieme di asserti. Inoltre, un asserto, può essere **conseguenza logica** dell'insieme.

Inconsistenza

Un insieme di asserti (o un asserto) e **inconsistenze** se è contraddittorio; ovvero, non è esiste nessuna situazione immaginabile in cui le affermazioni sono vere. In altre parole, se in tutte le situazioni immaginabili, c'è almeno un asserto che si falsa.

Consistenza

Per **consistenza**, invece, intendiamo che nell'insieme di asserti che facciamo non c'è alcuna contraddizione interna, ovvero che possono coesistere tutte senza creare conflitti logici. In altre parole, se in un sistema di affermazioni non è possibile dedurre da esso una contraddizione.

Validità

Un asserto è **valido** se è vero da qualsiasi situazione immaginabile. Un esempio di affermazione valida può essere "un oggetto è uguale a sé stesso", che evidentemente non può essere falsificato da nessuna situazione concreta. Un esempio di asserto non valido potrebbe essere "un gesso è bianco". Nota: non confondere la validità di un asserto o insieme con la validità di una argomentazione (ovvero un'argomentazione ben formata) poiché sono concetti diversi.

Nota: Consistenza è un concetto esistenziale (esiste una situazione in cui soddisfa) mentre la validità è universale (per ogni situazione esiste) così come l'inconsistenza (per nessuna situazione esiste).

Possiamo definire il concetto di validità in maniera più precisa: **una asserzione è valida se e soltanto se la sua contraddizione è inconsistente**.

Equivalenza

Una affermazione è uguale ad un'altra se dicono la stessa cosa, se hanno lo stesso significato. Nella logica matematica due asserti sono equivalenti se sono veri in tutte e sole le stesse situazioni; ovvero, se non esiste una situazione che li può distinguere (che una è vera in una situazione mentre l'altra no).

Il concetto di equivalenza può essere espresso in conseguenza logica; ovvero: $\{x\} \vDash y \ \ e \ \ \{y\} \vDash x$ (da un insieme x discende logicamente y e da un insieme y discende logicamente x)

Esempio di concetti equivalenti:

1) Carlo non dorme o sogna
2) Se carlo dorme, allora sogna

Conseguenza logica

Il concetto di **conseguenza logica** mette in relazione due oggetti; una affermazione ed un insieme di affermazioni. Sostanzialmente dice che una certa affermazione è implicita in un insieme di affermazioni, ovvero è ottenibile da una argomentazione ben formata.

La conseguenza logica è relazionabile con l'inconsistenza; infatti, presi un insieme X di asserti e x un asserto abbiamo che (x è conseguenza logica di X) $X \vDash x \Leftrightarrow X \cup \{\neg x\}$ è inconsistente (x e ottenibile da X se e soltanto se l'unione di X con il negato di x è un insieme inconsistente).

Esempio di conseguenza logica:

1) Se Carlo è italiano e Giovanni non è spagnolo, allora Rachel è tedesca
2) Se Rachel è tedesca allora (Lucia è francese o Giovanni è spagnolo)
3) Se Lucia non è francese allora Carlo è italiano
4) Giovanni non è spagnolo
   Quindi
5) Lucia è francese

Dimostrazione: Partiamo negando la 5; E consideriamo il seguente insieme: $\{1,2,3,4\} \cup \{\text{Lucia non è francese}\}$. Assumendo che Lucia non è francese abbiamo che Carlo è italiano (la 3); a questo punto possiamo dire per la 1 e la 4 che Rachel è tedesca; a questo punto dalla 2 abbiamo che una delle due tra Lucia è francese e $\text{Giovanni è spagnolo}$ deve essere vera; non possiamo escludere la seconda per via della 4 e quindi abbiamo che $\{\text{Lucia è francese}\}$ è una asserzione vera; che contraddice $\{\text{Lucia non è francese}\}$ rendendo l'insieme inconsistente.

Strutture

Possiamo osservare che spesso le asserzioni hanno una struttura simile. Le strutture che abbiamo trattato in precedenza possono essere riassunte nei seguenti esempi:

1) Tutti i P sono Q
2) X è P
   Quindi
3) X è Q

1) Se non A allora B
2) Non A
   Quindi
3) B

L'utilizzo di variabili permette di evitare fraintendimenti e deduzioni sbagliate.

Logica

La logica è la disciplina che studia la validità degli argomenti; ma c'è una definizione tecnica di logica che si riferisce ad uno specifico linguaggio (ad esempio: logica proposizionale, logica del primo ordine, logica dei predicati, etc...).

Una logica (o linguaggio logico) consta di tre componenti (si noti che i primi due sono veri per qualsiasi linguaggio):

1) **Sintassi**. l'insieme di simboli e regole per la loro combinazione, che definiscono le formule ben formate all'interno della logica.
2) **Semantica**. Una "funzione" che associa ad una espressione ben formata del linguaggio un qualche significato. Fornisce un'interpretazione del linguaggio, cioè assegna significato alle formule ben formate (frasi).
3) **Calcolo.** Cattura il concetto di argomentazione valida; è un modo di estrarre informazioni implicite da un insieme di informazioni rispettando la semantica. Più precisamente riguarda l'insieme di regole di inferenza che permettono di derivare nuove formule (teoremi) a partire da formule date (assiomi). Il calcolo stabilisce i meccanismi formali attraverso cui si può dedurre una conclusione a partire da premesse, senza fare riferimento al significato (semantica) delle formule stesse. Tale meccanismo gode delle proprietà di completezza (se un qualcosa è corretto, allora è generabile) e di correttezza (tutto quello che genera è corretto).

2. Linguaggio Proposizionale

Un linguaggio è un insieme di parole dove ogni parola è una sequenza finita di simboli. Ogni simbolo appartiene invece ad un insieme chiamato **alfabeto**. Le parole più semplici sono le variabili proposizionali (o anche dette **proposizioni atomiche**). Andiamo adesso a formalizzarne la grammatica.

Alfabeto

Per noi l'alfabeto è l'insieme delle affermazioni logiche che rappresentiamo tramite lettere maiuscole dell'alfabeto. L'alfabeto consta in un insieme di simboli diviso in due classi:

- **Simboli non logici**. Lettere a cui daremo un valore di verità chiamate proposizioni atomiche. Come ad esempio $A_1, A_2, A_3, \dots$ Si chiamano non logici perché non è assegnato nessun significato a priori (in situazioni diverse possono avere significati diversi).
- **Simboli logici**: hanno una semantica fissata dalla logica, mettono in relazione le proposizioni (sono operatori binari, eccetto per la negazione che è unaria). Sono rappresentati dai classici operatori o connettivi logici:

$\wedge$	$\vee$	$\rightarrow$	$\neg$	$\leftrightarrow$	$\oplus$
AND	OR	$IMPLICAZIONE$	$NEGAZIONE$	SSE	$OR\ ESCLUSIVO$

  Tali simboli sono fissati; indipendentemente dalla situazione hanno lo stesso significato.

Nota: I simboli logici del livello oggetto vanno distinti da quelli del metalinguaggio; quando voglio determinare o esprimere una relazione tra una proprietà di uno o più enunciati ed altri enunciati userò i seguenti simboli ⇒ ≡ ⇔

Tra i simboli logici abbiamo anche le parentesi ( ); queste hanno il solo scopo di raggruppare un insieme di simboli per non rendere ambiguo il significato di un'affermazione.

Espressioni

Una espressione è in generale una sequenza finita di simboli dell'alfabeto; ad esempio (∧ ∨ A_1 ¬ A_2 (¬) (la quale evidentemente non è ben formata). Il concetto di "ben formato" serve a discriminare tra espressioni sensate ed espressioni non sensate.

La formula (EBF: Espressione Ben Formata) è una espressione che rispetta determinati vincoli. Poiché le espressioni che posso definire sono infinite, ragionevolmente anche le espressioni ben formate sono infinite. Infatti, se A_1 è ben formata, lo sarà anche $A_1 \wedge A_2$; così come $A_1 \wedge A_2 \wedge A_3$ e così via…

Una descrizione finita di questo insieme infinito è il seguente:

Sia $AP = \{A_1, A_2, \dots, A_k, \dots\}$. LP (EBF) è il **più piccolo insieme di espressioni** tale che:

1) $AP \subseteq LP$ [caso base]
2) Se $\phi \in LP$ allora $\neg(\phi) \in LP$
3) Se $\phi_1, \phi_2 \in LP$ allora $(\phi) \circ (\phi_2) \in LP$ dove $\circ$ è un qualsiasi operatore binario

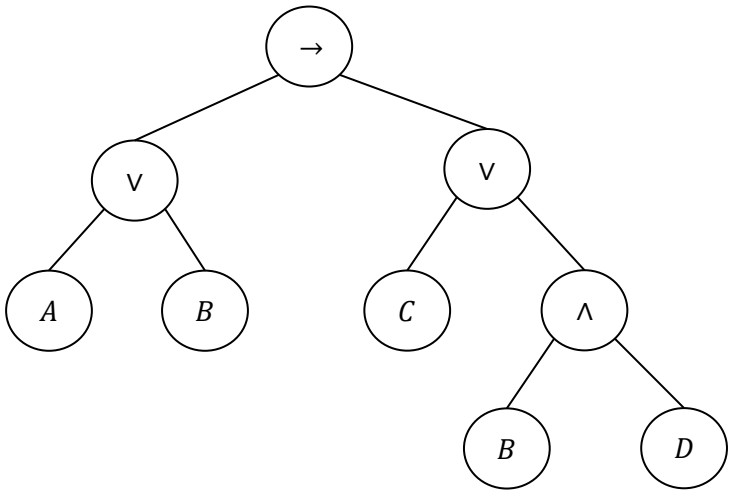
Nota: qui mi dice che <u>se alcune cose appartengono al linguaggio; allora altre cose gli appartengono</u> ma non descrive esplicitamente cosa c'è o non c'è all'interno del linguaggio. Fondamentale è l'espressione "il più piccolo insieme di espressioni" poiché se lo togliessi ammetterei anche insiemi più grandi e di conseguenza potrei ammettere anche espressioni mal formate come "se $(\wedge) \in LP$ allora $\neg(\wedge) \in LP$; il concetto "**il più piccolo**" mi forza ad escludere il vincolo descritto precedentemente. Quindi parto da un insieme e tolgo elementi finché non arrivo all'insieme dove non posso più togliere elementi e quindi avrò definito l'insieme LP.

Rappresentazione ad albero

Ne consegue che LP è l'insieme di tutte le espressioni ben formate. Questa definizione è molto simile a quella di albero binario e ci suggerisce che ogni operazione ben formata è isomorfa ad un albero binario se i connettivi logici che abbiamo sono al più binari.

Questa definizione induttiva è equivalente alla definizione matematica dove le variabili sono sostituite dai numeri e gli operatori binari da $+,\cdot,-,/$

Quindi per verificare che $(A \vee B) \rightarrow \big(C \vee (B \wedge D)\big)$ è una espressione ben formata o verifico che rispetti i vincoli 1,2 e 3 che descrivono LP; oppure rappresento un albero binario come il seguente:



Ad ogni espressione può essere associata una misura di complessità che è di fatto l'altezza della rappresentazione dell'albero sintattico (quindi nell'esempio precedente è 4).

Siccome ogni sequenza può essere rappresentata con un albero, allora ogni albero può essere rappresentato tramite una sequenza. Per linearizzare l'albero (renderlo una sequenza) le parentesi sono necessarie altrimenti non potrei definire quale sia la radice di ogni sottoalbero. Il problema dell'ambiguità può essere parzialmente ridotto definendo le precedenze. Infatti, se dico che $\wedge$ ha precedenza su $\vee$ allora la sequenza $A \wedge B \vee C$ non è più ambigua, ma comunque non posso definire senza parentesi $A \wedge (B \vee C)$. Dunque, le parentesi non sono eliminabili totalmente.

Modello

Una volta definita la sintassi, dobbiamo dare un significato alle frasi ben formate. Possiamo definire un significato di una frase partendo dai suoi componenti (tale semantica è definita composizionale). Il significato di una frase è il concetto di vero e falso; che va definito dal concetto di situazione poiché in base alla situazione una frase potrebbe essere vera o meno.

Il concetto di situazione prende il nome di **modello (**proposizionale) che è tipicamente definito con m. Per dire che una formula è vera o falsa rispetto un modello useremo il seguente simbolo: $m \vDash \phi$ (questo si legge con il modello m soddisfa ϕ oppure ϕ è vera nel modello m).

Non è compito del modello assegnare un significato agli operatori binari, l'unica cosa che può variare sono i simboli non logici; per cui l'unica informazione che un modello mi deve dare sono le condizioni di verità per cui quelle formule sono vere.

Ogni modello m è un sottoinsieme di AP. Quindi i simboli non logici contenuti in m sono veri in quel modello e tutti quelli che non sono contenuti in tal modello sono falsi. Con questa definizione un modello m definisce tutte e sole le formule vere in quel modello.

La relazione di soddisfazione di un modello può essere definita come il seguente sottoinsieme:

$$m \subseteq AP \Leftrightarrow (m \in 2^{AP})$$

Relazione di soddisfazione di un modello

A questo punto possiamo definire la **relazione di soddisfazione** $m \vDash \phi$ (si noti che il simbolo è lo stesso della conseguenza logica ma ovviamente assume significato diverso poiché mette in relazione un modello con una formula e non un insieme di formule ed una formula).

Una coppia m, ϕ appratiene alla relazione di soddisfazione ($m \vDash \phi$) se e solo se:

1) $\phi \in m$ se $\phi \in AP$
2) $m \nvDash \psi$ se $\phi = \neg\psi$ (se m soddisfa una formula ϕ non può soddisfare anche la sua negata)
3)
   a. $m \vDash \psi_1$ e $m \vDash \psi_2$ se $\phi = \psi_1 \wedge \psi_2$
   b. $m \vDash \psi_1$ oppure $m \vDash \psi_2$ se $\phi = \psi_1 \vee \psi_2$
   c. (Se $m \vDash \psi_1$ allora $m \vDash \psi_2$) se $\phi = \psi_1 \rightarrow \psi_2$
   d. ($m \vDash \psi_1$ se e solo se $m \vDash \psi_2$) se $\phi = \psi_1 \leftrightarrow \psi_2$
   e. $m \nvDash \psi_1$ oppure $m \nvDash \psi_2$ se $\phi = \psi_1 \oplus \psi_2$

Nota: Il concetto di falso verrà indicato con $\bot \triangleq A \wedge \neg A$; mentre il vero $\top \triangleq A \vee \neg A$.

Semantica

Siccome il linguaggio ha un insieme di proprietà elementari catturate dalle proposizioni atomiche (dell'insieme AP), usando gli operatori logici, possiamo costruire l'insieme di tutte le frasi ben formate scrivibili. Non è detto che tutte le frasi abbiano significati diversi (da cui il concetto di equivalenza).

Abbiamo visto che il modello è un insieme numerabile di proposizione atomiche $m \subseteq AP \Leftrightarrow m \in 2^{AP}$.

Ci sono vari modi per presentare la semantica nella logica classica (ma ovviamente tutti equivalenti); dal punto di vista computazionale potrebbe essere più utile un modo di rappresentazione del modello piuttosto di un altro. In ogni caso, per rappresentare un modello (infinito) è sufficiente avere una quantità finita di informazioni; l'informazione che basta è la definizione delle proposizioni atomiche (gli operatori logici). Ma tale definizione non permette il confronto tra sottoinsiemi diversi, poiché un modello potrebbe contenere cose che l'altro non ha e viceversa.

Potremmo quindi ridefinire la semantica introducendo un concetto di modello diverso da quello descritto nel capitolo precedente ma che; in un certo senso, contiene la stessa informazione. Questo concetto è definito come concetto di assegnamento.

Assegnamento

Un assegnamento σ per una formula ϕ è una funzione che va dalle proposizioni atomiche e restituisce o 0 (che rappresenta il falso) oppure 1 (il vero). In simboli: $\boldsymbol{\sigma: AP(\phi) \rightarrow \{0,1\}}$

È praticamente una vista sulla situazione dal punto di vista della formula: di cosa la formula sta parlando in relazione ad una certa situazione (praticamente è la tabella di verità).

Si noti che l'assegnamento da una rappresentazione parziale del mondo; a differenza del modello che è una rappresentazione completa del mondo e non solo una parte di essa.

σ può essere un assegnamento che associa ad ogni formula 0 o 1 (dato un assegnamento mi dice se quella formula è vera o falsa). Praticamente corrisponde ai valori di una riga in una tabella di verità e tale tabella è anche una possibile definizione della sua semantica.

Valore di verità

L'assegnamento σ, data la formula ϕ assume come già detto un valore di verità che può essere o vero o falso; definiremo con $val^{\sigma}(\phi)$ il **valore di verità** che σ assegna ad una formula ϕ.

Il comportamento di $val^{\sigma}(\phi)$ è definito come segue:

1) $val^{\sigma}(\phi) = \sigma(\phi)$ se $\phi \in AP(\phi)$ (posso dire $val^{\sigma}(\phi)$ solo perché vale $\phi \in AP(\phi)$)
2) $val^{\sigma}(\neg\phi) = 1 - val^{\sigma}(\phi)$ (per sapere il valore di verità di un negato prendo il valroe di verità e lo sottraggo ad uno)
3) $val^{\sigma}(\phi_1 \wedge \phi_2) = \min\big(val^{\sigma}(\phi_1), val^{\sigma}(\phi_2)\big)$
4) $val^{\sigma}(\phi_1 \vee \phi_2) = \max\big(val^{\sigma}(\phi_1), val^{\sigma}(\phi_2)\big)$ (si noti che è esattamente il duale della 3)
5) $val^{\sigma}(\phi_1 \rightarrow \phi_2) = 1$ sse $val^{\sigma}(\phi_1) \leq val^{\sigma}(\phi_2)$ (l'implicazione è una relazione di confronto debole)
6) $val^{\sigma}(\phi_1 \leftrightarrow \phi_2) = 1$ sse $val^{\sigma}(\phi_1) = val^{\sigma}(\phi_2)$

Questa definizione mi fornisce un algoritmo ricorsivo per valutare la validità di una formula. Risalgo nell'albero di ricorsione a seconda dell'operatore il caso rappresentato.

Tautologia

Dato un insieme di tutte le formule definiamo una tautologia come l'insieme che contiene tutti gli assegnamenti con valore vero. In simboli: $\mathrm{TAUT}(\phi) \Leftrightarrow val^{\sigma}(\phi) = 1 \forall \sigma$ (la tautologia è vera in tutte le viste che ho sul mondo; è simile al concetto dato informalmente di validità)

Se ho una funzione $f: A \rightarrow B$ posso costruire B^A formule; visto che il nostro f va da A all'insieme $\{0,1\}$; abbiamo 2^A dove A è il numero di proposizioni atomiche in una formula (si noti l'analogia con l'insieme delle parti) quindi questo significa che la funzione è biettiva. Una funzione di questo genere rappresenta uno ed un solo sottoinsieme perché descrive una funzione caratteristica (1 è l'appartenenza e 0 è la non appartenenza). Per questo abbiamo la definizione $m \subseteq AP \Leftrightarrow m \in 2^{AP}$.

Dunque, una tabella di verità avrà $2^{\text{numero di variabili booleane}}$. Che è una definizione equivalente a quella data al valore di verità (ovvero i punti da 1 a 6).

La tautologia significa vero in tutte le viste che ho sul mondo (si noti l'equivalenza con il concetto dato di validità)

Formalizzazione dei concetti

Data una formula ϕ, diremo che è valida (in simboli $\vDash \phi$) sse $\forall m \in 2^{\mathrm{AP}},\ m \vDash \phi$ (la proprietà atomica che mette relazione modello e formula è proprio $m \vDash \phi$ e lo abbiamo descritto come $\phi \in AP(\phi)$).

Di seguito forniamo una definizione formale dei concetti fondamentali descritti nel capitolo 1.

L'elenco segue questa sintassi: **formula è "*concetto*"**: *"simbolo sse descrizione"*

- **ϕ è valida**:
  $\vDash \phi$ sse $\forall m \in 2^{\mathrm{AP}},\ m \vDash \phi$

- **ϕ è soddisfacibile**: $sat(\phi)$ sse $\exists m \in 2^{AP},\ m \vDash \phi$
- **ϕ è insoddisfacibile** (simile al concetto di inconsistente): $unsat(\phi)$ sse non $\exists m \in 2^{AP}\ \ m \vDash \phi$ (che è equivalente a $not\ sat(\phi)$)
- **ϕ è conseguenza logica di $\Gamma \subseteq AP$**: $\Gamma \vDash \phi$ sse $\forall m \in 2^{AP}\ \ m \vDash \Gamma$ implica $m \vDash \phi$. Nota: dire che $m \vDash \Gamma$ significa che $\forall \psi \in \Gamma, m \vDash \psi$ (praticamente Γ può essere estrapolato da ϕ; quindi in ϕ c'è implicitamente Γ); tale definizione può essere semplificata con l'espressione $m \vDash \Gamma \Rightarrow m \vDash \phi$ (m soddisfa Γ).

Si noti che se $\Gamma \vDash \phi$ allora **non esiste** un modello $m \in 2^{AP}$ tale che $m \vDash \Gamma$ e $m \nvDash \phi$; che è equivalente a dire non esiste un m tale che $m \vDash \Gamma$ e $m \vDash \neg\phi$. O ancora, non esiste un modello $m \in 2^{AP}$ tale che $m \vDash \Gamma \cup \{\neg\phi\}$. In altre parole, possiamo concludere che $unsat(\phi) \Leftrightarrow \vDash \neg\phi$ (relazione tra conseguenza logica e insoddisfacibilità)

Il concetto di validità è quello che nella semantica di oggi corrisponde alla tautologia. Quindi il concetto di tautologia che viene fuori dalle tabelle di verità corrisponde (o dovrebbe) alla validità.

Teorema

$\forall \boldsymbol{\phi} \in \boldsymbol{LP}$ (per ogni formula del linguaggio) $\boldsymbol{TAUT(\phi)} \Leftrightarrow \vDash \boldsymbol{\phi}$

Dimostrazione

Cominciamo con il dimostrare che una formula è valida se vale la tautologia (implicazione ⇒).

$\forall m \in 2^{AP}$ e $\phi \in LP$ posso costruire una funzione σ_m: $AP(\phi) \to \{0,1\}$ (funzione in 0 o 1) tale che $\sigma_m(p) = 1$ sse $p \in m\big(\cap\, AP(\phi)\big)$.

Nota che $m \cap AP(\phi)$ è un sottoinsieme di AP che contiene solo le proposizioni atomiche

$$(1)\ \forall \phi \in LP, \forall m \in 2^{AP} :\ \ m \vDash \phi \Leftrightarrow val^{\sigma_m}(\phi) = 1$$

Ora supponendo la (1) vera possiamo dimostrare l'implicazione ⇒; procederemo per assurdo: supponiamo che sia una tautologia ma esiste un modello che non soddisfa la (1).

Se è falsa la validità significa che $\nvDash \phi \Leftrightarrow \exists m : m \nvDash \phi \Leftrightarrow m \vDash \neg\phi$ quindi esiste un σ^m tale che $val^{\sigma_m}(\neg\phi) = 1$; dunque, σ_m è un assegnamento che rende vera la negazione di ϕ. Andiamo ora a valutare la negazione: $val^{\sigma_m}(\neg\phi) = 1 \Leftrightarrow 1 - val^{\sigma_m}(\phi) = 1 \Leftrightarrow val^{\sigma_m}(\phi) = 0$ e quindi non può essere una tautologia; dunque, ciò contraddice l'ipotesi $TAUT(\phi)$ (ovvero l'ipotesi (1)).

L'altro verso si dimostra in maniera analoga, e lo si lascia per esercizio. Supponendo sempre la (1) vera, dimostriamo l'implicazione ⇐: Supponiamo per assurdo che esiste un modello che soddisfa la 1 ma che la tautologia non sia vera. Poiché non è vera la tautologia significa che esiste un assegnamento per cui $val^{\sigma}(\phi) = 0 \Leftrightarrow val^{\sigma}(\neg\phi) = 1 \overset{per\ la\ (1)}{\Longleftrightarrow} m \vDash \phi$; ma ciò contraddice l'ipotesi $m \vDash \phi$ (un modello non può soddisfare sia una formula che la sua negata per definizione di m).

A questo punto, ci basta dimostrare la (1) (fino ad ora l'abbiamo supposta vera). Per ogni modello esiste un unico assegnamento σ_m, ma non vale il viceversa (posso avere più modelli che contengono quella formula). Sia un σ posso costruire un m_σ semplicemente mettendo tutte le σ vere in m ed escludendo tutte le false.

Se $\sigma(p) = 1$ allora $p \in m_\sigma$ e se $\sigma(p) = 0$ allora $p \notin m$ (così ho costruito un qualsiasi modello che soddisfi tale vincolo). Ora devo dimostrare che qualsiasi modello coerente soddisfa la formula ϕ se e solo se $val^{\sigma}(\phi) = 1$.

Dimostriamo $m \vDash \phi \Leftrightarrow val^{\sigma_m}(\phi) = 1$ per induzione strutturale su ϕ (per induzione sull'altezza dell'albero sintattico di p).

- Il caso base è con $h = 0$; ovvero $\phi = p \in AP$ (è una proposizione atomica arbitraria):
  $$m \vDash p \Leftrightarrow p \in m \Leftrightarrow \sigma_m(p) = 1 \Leftrightarrow val^{\sigma_m}(p) = 1$$
  Ma una conseguenza di questa è che $p \in AP(\phi)$ e quindi $\sigma_m(p) = 1$
- Caso induttivo; va dimostrato per tutte le relazioni del modello:
  - $\phi = \neg\psi$ (è la negazione di un'altra formula)
    $m \vDash \neg\psi \Leftrightarrow m \nvDash \psi$ per ipotesi induttiva $\forall m, m \vDash \psi \Leftrightarrow val^{\sigma_m}(\psi) = 1$ quindi $m \nvDash \psi \Leftrightarrow val^{\sigma_m}(\psi) = 0 \Leftrightarrow 1 - val^{\sigma_m}(\psi) = 1 \Leftrightarrow val^{\sigma_m}(\neg\psi) = 1.$
    E quindi abbiamo concluso che $val^{\sigma_m}(\phi) = 1$
  - $\phi = \phi_1 \wedge \phi_2$: $m \vDash \phi_1 \wedge \phi_2 \Leftrightarrow m \vDash \phi_1$ e $m \vDash \phi_2 \Leftrightarrow val^{\sigma_m}(\phi_1) = 1$ e $val^{\sigma_m}(\phi_2) = 1$
    $\Leftrightarrow \min\{val^{\sigma_m}(\phi_1), val^{\sigma_m}(\phi_2)\} = 1 \Leftrightarrow val^{\sigma_m}(\phi_1 \wedge \phi_2) = 1$
  - $\phi = \phi_1 \vee \phi_2$: $m \vDash \phi_1 \vee \phi_2 \Leftrightarrow m \vDash \phi_1$ o $m \vDash \phi_2 \Leftrightarrow val^{\sigma_m}(\phi_1) = 1$ o $val^{\sigma_m}(\phi_2) = 1$
    $\Leftrightarrow \max\{val^{\sigma_m}(\phi_1), val^{\sigma_m}(\phi_2)\} = 1 \Leftrightarrow val^{\sigma_m}(\phi_1 \vee \phi_2) = 1$
  - $\phi = \phi_1 \rightarrow \phi_2$: $m \vDash \phi_1 \rightarrow \phi_2 \Leftrightarrow m \nvDash \phi_1$ o $m \vDash \phi_2 \Leftrightarrow val^{\sigma_m}(\phi_1) = 0$ o $val^{\sigma_m}(\phi_2) = 1$
    $\Leftrightarrow val^{\sigma_m}(\phi_1) \leq val^{\sigma_m}(\phi_2) \Leftrightarrow val^{\sigma_m}(\phi_1 \rightarrow \phi_2)$

Equivalenze logiche

Leggi di De Morgan

- $\boldsymbol{A \vee B \equiv \neg(\neg A \wedge \neg B)}$
  - $val^{\sigma}(A \vee B) = 1 \Leftrightarrow \mathbf{max}\{\boldsymbol{val^{\sigma}(A), val^{\sigma}(B)}\} = \mathbf{1}$
  - $val^{\sigma}\big(\neg(\neg A \wedge \neg B)\big) = 1 \Leftrightarrow 1 - val^{\sigma}(\neg A \wedge \neg B) = 1 \Leftrightarrow val^{\sigma}(\neg A \wedge \neg B) = 0$
    $\Leftrightarrow \min\{val^{\sigma}(\neg A), val^{\sigma}(\neg B)\} = 0 \Leftrightarrow \min\{1 - val^{\sigma}(A), 1 - val^{\sigma}(B)\} = 0$
    $\Leftrightarrow val^{\sigma}(A) = 1\ o\ val^{\sigma}(B) = 1 \Leftrightarrow \mathbf{max}\{\boldsymbol{val^{\sigma}(A), val^{\sigma}(B)}\} = \mathbf{1}$
- $\boldsymbol{A \wedge B \equiv \neg(\neg A \vee \neg B)}$
  - $val^{\sigma}(A \wedge B) = 1 \Leftrightarrow \mathbf{min}\{\boldsymbol{val^{\sigma}(A), val^{\sigma}(B)}\} = \mathbf{1}$
  - $val^{\sigma}\big(\neg(\neg A \vee \neg B)\big) = 1 \Leftrightarrow 1 - val^{\sigma}(\neg A \vee \neg B) = 1 \Leftrightarrow val^{\sigma}(\neg A \vee \neg B) = 0$
    $\Leftrightarrow \max\{val^{\sigma}(\neg A), val^{\sigma}(\neg B)\} = 0 \Leftrightarrow \max\{1 - val^{\sigma}(A), 1 - val^{\sigma}(B)\} = 0$
    $\Leftrightarrow val^{\sigma}(A) = 1\ e\ val^{\sigma}(B) = 1 \Leftrightarrow \mathbf{min}\{\boldsymbol{val^{\sigma}(A), val^{\sigma}(B)}\} = \mathbf{1}$

Implicazione

$\phi_1 \rightarrow \phi_2 \equiv \neg\phi_1 \vee \phi_2 \quad \forall \phi_1, \phi_2 \in LP$

Dimostrazione

$val^{\sigma}(\phi_1 \rightarrow \phi_2) = 1 \Leftrightarrow val^{\sigma}(\phi_1) \leq val^{\sigma}(\phi_2)$; quindi o $val^{\sigma}(\phi_1) = 0$ (e quindi l'altra è necessariamente vera) oppure $val^{\sigma}(\phi_2) = 1$ ma $val^{\sigma}(\phi_1) \Leftrightarrow 1 - val^{\sigma}(\phi_1) = 1$ e quindi $1 - val^{\sigma}(\phi_1) = 1$ o $val^{\sigma}(\phi_2) = 1 \Leftrightarrow \max\{1 - val^{\sigma}(\phi_1), val^{\sigma}(\phi_2)\} = 1 \Leftrightarrow \max\{val^{\sigma}(\neg\phi_1), val^{\sigma}(\phi_2)\} = 1 \Leftrightarrow val^{\sigma}(\neg\phi_1 \vee \phi_2)$ C.V.D

Concatenazioni di implicazioni

$\phi_1 \rightarrow (\phi_2 \rightarrow \phi_3) \equiv (\phi_1 \wedge \phi_2) \rightarrow \phi_3$

Dimostrazione

Visto che $val^{\sigma}(\phi_1 \rightarrow \phi_2) = 1 \Leftrightarrow val^{\sigma}(\phi_1) \leq val^{\sigma}(\phi_2)$ evidentemente abbiamo che $val^{\sigma}(\phi_1 \rightarrow \phi_2) = 0 \Leftrightarrow val^{\sigma}(\phi_1) > val^{\sigma}(\phi_2)$ (abbiamo praticamente fatto l'opposto del $\leq$); consideriamo dunque i casi in cui le implicazioni sono false.

$val^{\sigma}(\phi_1 \to (\phi_2 \to \phi_3)) = 0 \Leftrightarrow val^{\sigma}(\phi_1) > val^{\sigma}(\phi_2 \to \phi_3) \Leftrightarrow val^{\sigma}(\phi_1) = 1$ e $val^{\sigma}(\phi_2 \to \phi_3) = 0 \Leftrightarrow$
$\underbrace{val^{\sigma}(\phi_1) = 1 \text{ e } val^{\sigma}(\phi_2) = 1}$ e $val^{\sigma}(\phi_3) = 0 \Leftrightarrow \min\{val^{\sigma}(\phi_1), val^{\sigma}(\phi_2)\} = 1$ e $val^{\sigma}(\phi_3) = 0 \Leftrightarrow$
$val^{\sigma}(\phi_1 \wedge \phi_2) = 1$ e $val^{\sigma}(\phi_3) = 0 \Leftrightarrow val^{\sigma}(\phi_1 \wedge \phi_2) > val^{\sigma}(\phi_3) \Leftrightarrow val^{\sigma}((\phi_1 \wedge \phi_2) \to \phi_3) = 0$

Questo è equivalente a dire che $val^{\sigma}(\phi_1 \to (\phi_2 \to \phi_3)) = val^{\sigma}((\phi_1 \wedge \phi_2) \to \phi_3)$ poiché sono false in tutti e soli gli stessi casi (dualmente saranno vere in tutti e soli gli altri casi). C.V.D.

Questa equivalenza dice che una catena di implicazioni innestate ha lo stesso valore di verità di una catena innestata di end che implica l'ultima:

$$\phi_1 \to \big(\phi_2 \to \big(\dots(\phi_{k-1} \to \phi_k)\big)\big) \equiv (\phi_1 \wedge \phi_2 \wedge \dots \wedge \phi_{k-1}) \to \phi_k$$

Sostituzione

La validità di una formula di un linguaggio proporzionale LP è invariante rispetto a sostituzione (la validità di una formula non varia se sostituisco alle proposizioni che compaiano altre formule ad esse coerenti).

Una sostituzione è una funzione $\rho: AP \to LP$ (mappa una proposizione atomica in una formula).

Un esempio di ρ potrebbe essere il seguente: $p: \{A \leftarrow (A \vee B), B \leftarrow \neg A; C \leftarrow C, \dots\}$ (sostituisco ad una lettera una formula). Data una formula $\phi \in LP$ posso definire una nuova formula $\phi\rho$ rimpiazzando simultaneamente tutte le proposizioni atomiche p di ϕ con $\rho(p)$. Ovvero $\forall p \in AP(\phi)$ sostituisco la formula $\rho(p)$.

Esempio: $\phi = (A \wedge \neg B) \vee C$ e $p: \{A \leftarrow (A \cup B), B \leftarrow \neg A; C \leftarrow C\}$ avrò $\phi\rho = \big((A \vee B) \wedge \neg(\neg A)\big) \vee C$

Questo è il concetto di applicazione di una funzione ρ di sostituzione a una formula ϕ.

Introduciamo il concetto semantico di sostituzione (prima abbiamo dato una definizione sintattica). Sia σ un assegnamento e ρ una sostituzione allora posso definire σ^{ρ} come un assegnamento risultante da sostituzione ρ applicata a σ. Ciò significa, ad esempio che se ho $\rho: A \leftarrow A \wedge B$ vado ad assegnare ad A il valore di verità di $A \wedge B$.

Esempio:

$\sigma_1: \begin{matrix} A \leftarrow 1 \\ B \leftarrow 0 \\ C \leftarrow 1 \end{matrix}, \sigma_2: \begin{matrix} A \leftarrow 0 \\ B \leftarrow 1 \\ C \leftarrow 1 \end{matrix}$ e $\rho = \{A \leftarrow A \vee B, B \leftarrow \neg A, C \leftarrow C\}$ avremo $\sigma_1^{\rho}: \begin{matrix} A \leftarrow val^{\sigma_1^{\rho}}(A \vee B) = 1 \\ B \leftarrow val^{\sigma_1^{\rho}}(\neg A) = 0 \\ C \leftarrow val^{\sigma_1^{\rho}}(C) = 1 \end{matrix}, \sigma_2^{\rho}: \begin{matrix} A \leftarrow 1 \\ B \leftarrow 1 \\ C \leftarrow 1 \end{matrix}$

Definizione induttiva della sostituzione. $\phi\rho$ è definita come segue:

- se $\phi = p \in AP$ allora $\phi\rho = \rho(\phi)$
- se $\phi = \phi_1 \wedge \phi_2$ allora $\phi\rho = (\phi_1\rho) \wedge (\phi_2\rho)$
- se $\phi = \neg\psi$ allora $\phi\rho = \neg(\psi\rho)$

Lemma 1

$\forall \phi \in LP$, assegnamento σ per ϕ e sostituzione ρ vale che $val^{\sigma}(\phi\rho) = val^{\sigma^{\rho}}(\phi)$

Dimostrazione

Quindi dobbiamo dimostrare che $\boldsymbol{\sigma^{\rho}(p) = val^{\sigma}(\rho(p))}$ e lo faremo per induzione sulla struttura di ϕ.

- Caso base: $\phi = p \in PA(\phi)$.

$$val^{\sigma}(\phi\rho) = \boldsymbol{val^{\sigma}(\rho(p))} = \boldsymbol{\sigma^{\rho}(p)} = val^{\sigma^{\rho}}(\phi)$$

- Caso induttivo
  - $\phi = \neg\psi$:
    $val^{\sigma}(\phi\rho) = val^{\sigma}\big(\neg(\psi\rho)\big)$ che per ipotesi induttiva $val^{\sigma}(\psi\rho) = val^{\sigma^{\rho}}(\psi)$; quindi
    $val^{\sigma}(\phi\rho) = val^{\sigma}\big(\neg(\psi\rho)\big) = 1 - val^{\sigma}(\psi\rho) = 1 - val^{\sigma^{\rho}}(\psi) = val^{\sigma^{\rho}}(\neg\psi) = val^{\sigma^{\rho}}(\phi)$
  - $\phi = \phi_1 \wedge \phi_2$
    $$val^{\sigma}(\phi\rho) = val^{\sigma}\big((\phi_1 \wedge \phi_2)\rho\big) = val^{\sigma}(\phi_1\rho \wedge \phi_2\rho) = \min\{val^{\sigma}(\phi_1\rho), val^{\sigma}(\phi_2\rho)\}$$
    $$= \min\{val^{\sigma^{\rho}}(\phi_1), val^{\sigma^{\rho}}(\phi_2)\} = val^{\sigma^{\rho}}(\phi)$$
  - $\phi = \phi_1 \vee \phi_2$
    $$val^{\sigma}(\phi\rho) = val^{\sigma}\big((\phi_1 \vee \phi_2)\rho\big) = val^{\sigma}(\phi_1\rho \vee \phi_2\rho) = \max\{val^{\sigma}(\phi_1\rho), val^{\sigma}(\phi_2\rho)\}$$
    $$= \max\{val^{\sigma^{\rho}}(\phi_1), val^{\sigma^{\rho}}(\phi_2)\} = val^{\sigma^{\rho}}(\phi)$$
  - $\phi = \phi_1 \rightarrow \phi_2$
    $$val^{\sigma}(\phi\rho) = val^{\sigma}\big((\phi_1 \rightarrow \phi_2)\rho\big) = val^{\sigma}(\phi_1\rho \rightarrow \phi_2\rho) \Leftrightarrow val^{\sigma}(\phi_1\rho) \leq val^{\sigma}(\phi_2\rho)$$
    $$\Leftrightarrow val^{\sigma^{\rho}}(\phi_1) \leq val^{\sigma^{\rho}}(\phi_2) \Leftrightarrow val^{\sigma^{\rho}}(\phi_1 \rightarrow \phi_2)$$

Teorema di sostituzione

$\forall \phi \in LP$ e funzione di sostituzione ρ, vale che: $\vDash \boldsymbol{\phi} \Rightarrow \vDash \boldsymbol{\phi\rho}$

Dimostrazione

Essendo una implicazione, assumiamo falsa la condizione e dimostriamo falsa la premessa. Quindi dimostriamo che $\nvDash \phi\rho \Rightarrow \nvDash \phi$. Quindi $\exists\sigma : val^{\sigma}(\phi\rho) = 0$.

Per il Lemma 1 precedente, $val^{\sigma}(\phi\rho) = val^{\sigma^{\rho}}(\phi)$ e quindi $\exists\sigma^{\rho} : val^{\sigma^{\rho}}(\phi) = 0$ ma ciò implica che ϕ non è valida. Quindi, poiché abbiamo dimostrato che $\nvDash \phi\rho \Rightarrow \nvDash \phi$ il teorema è dimostrato.

Osservazioni

Dunque, se prendiamo una formula valida tipo $\phi = A \vee \neg A$ allora anche $(A \wedge B) \vee \neg(A \wedge B)$ è valida, così come $(A \vee \neg A) \vee \neg(A \vee \neg A)$ e così via...

La validità viene preservata ma ciò non è detto che valga per la Soddisfacibilità, se ϕ è soddisfacibile, infatti, non è detto che lo sia anche $\phi\rho$ (potrebbe esserlo ma non è affatto detto che lo sia).
Esempio: $\phi = A \wedge B$ e $\rho = \{B \leftarrow \neg A\}$ avrò $\phi\rho = A \wedge \neg A$ che non è soddisfacibile.

Correlazione con la insoddisfacibilità

Se una formula è insoddisfacibile ed applico una trasformazione questa resta insoddisfacibile. Infatti, sapendo che ϕ è $unstat \Leftrightarrow \neg\phi$ è valida, presa una sostituzione ρ abbiamo

$$\phi \text{ è } unsat \Leftrightarrow \neg\phi \text{ è valida} \Rightarrow (\neg\phi)\rho \text{ è valida} \Leftrightarrow \neg(\phi\rho) \text{ è valida} \Leftrightarrow \phi\rho \text{ è } unsat$$

Problemi di decisione

Chiedersi se una formula ϕ è soddisfacibile (sat) è un problema decisionale, così come $\phi \in VALID$ o $\phi \in UNSAT$; o ancora $(\Gamma, \phi) \in LC$ (conseguenza logica, è come chiedersi $\Gamma \vDash \phi$).

Ovviamente un algoritmo che risolva uno di questi problemi, può risolvere anche gli altri per la correlazione già discussa di $SAT, VALID, UNSAT \; e \; LC$

Teorema di Cook-Levin

SAT è un problema NP-completo

Osservazioni

I problemi NP-completi sono un sottoinsieme dei problemi NP. con la proprietà che NP-completi sono considerati difficili dal punto di vista della complessità, il che significa che sono difficili da risolvere in tempo polinomiale.

Si ricorda infatti che $P \subseteq NP$ ma non si sa se è vera l'uguaglianza. Poiché non sappiamo se $P = NP$ sussiste, non sappiamo se esiste un algoritmo che risolva SAT in tempo polinomiale per una qualsiasi formula.

Dimostrazione

Per verificare la soddisfacibilità di una formula, basterebbe provare tutti gli assegnamenti possibili e verificare la soddisfacibilità per ogni assegnamento. Quindi divido il problema di SAT in N sottoproblemi di soddisfazione.

$$\phi \in SAT \Leftrightarrow \exists \sigma : \underbrace{\sigma \vDash \phi}_{\substack{\text{problema di soddisfazione} \\ \text{o model checking}}}$$

La complessità di un algoritmo di questo tipo è almeno tempo lineare (devo visitare almeno tutte le foglie). Ma l'assegnazione ad ogni nodo (il calcolo del suo valore) avviene nel seguente modo

- Nodo foglia (tempo costante): basta leggere il valore della proposizione atomica
- Nodo interno (tempo costante): se ho già il valore dei sottoalberi basta prendere il valore nella tabella di verità in base all'operatore nel nodo (accesso ad array). Quindi il costo sarà anch'esso a tempo costante.

Dunque, in tempo lineare sulla dimensione della formula posso verificare il problema SAT (model checking).

Tuttavia, la costruzione dell'albero è esponenziale se svolta in modo deterministico; infatti, ogni volta devo prima costruire tutto l'albero e poi verificarne la soddisfacibilità e se la verifica non va a buon fine devo ripetere il processo con un'altra combinazione di valori per le proposizioni atomiche.

$$\exists \sigma . \sigma \vDash \phi ? \quad \text{con } \sigma : AP(\phi) \to \{0,1\}$$
$$|AP(\phi)| = |\{\sigma | \sigma \text{ è un assegnamento su } AP(\phi)\}| = 2^{|AP(\phi)|} \leq 2^{|\phi|}$$

Queste considerazioni hanno come conseguenza il fatto che SAT è NP-completo.

Algoritmo di decisione

Un tipo di problema SAT è il seguente: $\exists \sigma : \sigma \vDash \phi$?

Dato $AP(\phi)$, cerchiamo l'insieme $\{\sigma | \sigma \text{ è assegnamento su } AP(\phi)\} \leq 2^{|AP(\phi)|} \leq 2^{|\phi|}$ sapendo che $\sigma: AP(\phi) \to \{0,1\}$ (funzione in 0,1).

Quindi cerchiamo un algoritmo in cui la soluzione impiega non più di $2^{|\phi|}$ (che raggiunge sicuramente quando la formua non è soddisfacibile).

N.B.: Non esiste un algoritmo per SAT che è migliore di $2^{|\phi|}$ (eccetto per alcuni casi particolari che ovviamente sono validi solo per tipi di problema particolari)

Un esempio di problemi computazionalmente difficili è quello di cercare il percorso semplice massimo in un grafo.

Forma Normale

Si può dimostrare che ogni formula della logica proposizionale può essere riscritta in una formula del tutto equivalente che però ha una forma fissata (astratta ma fissata). Le **forme normali** definiscono uno schema che non contiene sintatticamente tutte le formule ma ogni formula può essere riscritta in tale formato.

Le due forme normali più diffuse sono DNF (*Disjuacntive Normal Forme*, Forma Normale Disgiuntiva in italiano) e CNF (Conjuanctive NF, Forma Normale Congiuntiva).

Una clausola congiuntiva $C^{\wedge}$ è una sequenza di congiunzioni $(l_1 \wedge l_2 \wedge \dots \wedge l_n)$ dove ogni proposizione atomica (chiamata letterale) è tale che $l_i \in \{P, \neg P\}$ (o è una proposizione atomica oppure è una negazione di una proposizione atomica). Mentre una clausola disgiuntiva sarà $C^{\vee} = (l_1 \vee l_2 \vee \dots \vee l_n)$.

Una forma normale è DNF se è una disgiunzione di clausole congiuntive:

$$\boldsymbol{DNF} = \bigvee_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{C_i^{\wedge}} = (C_1^{\wedge} \vee C_2^{\wedge} \vee \dots \vee C_n^{\wedge})$$

(è praticamente una grossa disgiunzione dove i disgiunti sono della forma $(l_1 \wedge l_2 \wedge \dots \wedge l_n)$)

Analogamente la CNF sarà il duale:

$$\boldsymbol{CNF} = \bigwedge_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{C_i^{\vee}} = (C_1^{\vee} \wedge C_2^{\vee} \wedge \dots \wedge C_n^{\vee})$$

con $C_i^{\vee} = \left(l_1^i \vee l_2^i \vee \dots \vee l_n^i\right)$ clausola disgiuntiva.

Ogni formula può essere scritta nella forma DNF o CNF.

Cominciamo con la DNF. Sappiamo che ogni formula ha una sua tabella di verità che ci dice il valore di verità di ogni formula. Quindi prendiamo una tabella di verità del seguente tipo:

$\rho_1\ \rho_2 \dots \rho_k$	ϕ
σ	0
σ'	1
$\vdots$	$\vdots$

Voglio scrivere una formula che è vera soltanto per gli assegnamenti veri nella tabella. A tal scopo ignoro gli assegnamenti che la rendono falsa; quindi, considero solo gli assegnamenti veri.

La formula può essere descritta con la seguente disgiunzione: $\Sigma_{\phi} \subset \{\sigma | \sigma \vDash \phi\}$, ad esempio consideriamo $\sigma = \begin{matrix} p_1 & p_2 & p_3 \\ 0 & 1 & 1 \end{matrix} \leftrightarrow (\neg p_1 \wedge p_2 \wedge p_3)\ \forall \sigma \in \Sigma_{\phi}$

A questo punto, la disgiunzione delle clausole di ogni Σ_{ϕ} sarà

$$\phi' = \bigvee \{C | \sigma \in \Sigma_{\phi}\}$$

Questa formula è equivalente alla tabella di verità generata. Tale formula ϕ' è tanto grande quanto il numero di assengamenti veri (poiché ogni assegnamento può essere scritto nella clausola congiuntiva come l'esempio $\sigma = (\neg p_1 \wedge p_2 \wedge p_3)$)

Si noti che la formula per DNF (così come la CNF) usa solo tre operatori: ¬,∨,∧

Questo ci mostra che ogni formula può essere messa in DNF, ma vediamo per CNF, che ovviamente avrà bisogno di una unione (congiunzione) di disgiunzioni. In questo caso gli assegnamenti che mi interessano sono quelli che rendono falsa la formula, poiché ognuno di quelli renderebbe falsa la CNF (praticamente una congiunzione di negazioni)

$$\Sigma_{\phi}^{\wedge} = \{\sigma | \sigma \in \Sigma_{\phi}^{\wedge}\} \quad \text{con} \quad \Sigma_{\phi} \subset \{\sigma | \sigma \not\models \phi\}$$

Prendiamo $\neg(l_1 \vee \ldots \vee l_n)$ che con De Morgan è pari a $(\neg l_1 \wedge \ldots \wedge \neg l_n)$ (faccio la negazione della clausula congiuntiva); quindi $C_{\sigma}^{\wedge} = (l_1 \wedge l_2 \wedge \ldots \wedge l_n) = \neg(\neg l_1 \vee \neg l_2 \vee \ldots \vee \neg l_n) = \neg C$

$$\phi' = \bigwedge_{\sigma \in \Sigma_{\phi}^{\wedge}} C_{\sigma}^{\wedge}$$

Si scrive una formula per ogni assegnamento, la si nega e infine si usa De Morgan per avere la nostra clausola di disgiunzioni.

Soddisfacibilità della DNF

La soddisfacibilità di una forma DNF la si può fare in tempo polinomiale. La validità di una DNF è difficile, così come la validità di una CNF ma è semplice dimostrarne la soddisfacibilità.

Se voglio verificare che $\phi \in DNF$ basta verificare che solo una congiunzione sia soddisfacibile (poiché sono in or), quindi posso prendere ciascun $C_i = \left(l_1^i \wedge \ldots \wedge l_m^i\right)$ e verificare la sua non soddisfacibilità.

Per verificare che $\phi = \bigvee C_i$ sia insoddisfacibile devono essere insoddisfacibili tutti i C_i e l'unica possibilità per verificarlo è che il suo negato sia valido. Ora verificare la validità di una congiunzione di letterati è polinomiale; dunque, in tempo polinomiale posso risolvere il problema descritto.

Si potrebbe pensare che questo vada contro al teorema di Cook ma bisogna considerare che si è passati dalla formula proposizionale classica alla sua forma DNF; quindi, è la traduzione a rendere il problema risolvibile in tempo polinomiale (se la traduzione è fattibile in tempo polinomiale ovviamente lo risolvo con un algoritmo polinomiale).

Esempio

$(l_1 \vee l_2) \wedge (l_3 \vee l_4)$; applichiamo la proprietà distributiva: $\left((l_1 \vee l_2) \wedge l_3\right) \vee \left((l_1 \vee l_2) \wedge l_4\right)$ che diventa (sempre per la stessa prorpeità) $(L_1 \wedge l_2) \vee (l_2 \wedge l_3) \vee (l_1 \wedge l_4) \vee (l_2 \wedge l_4)$

Ogni clausola disgiuntiva è in corrispondenza biunivoca con l'insieme delle funzioni che prendono in ingresso le due clausole della CNF e sostituisce l'indice di uno dei letterali. In altre parole, $f: \{1, .., k\} \to \{1, \ldots, n\}$ (significa che prendo una clausola e mi dà l'indice del disgiunto che scelgo) che ha complessità n^k con n numero di letterali e k numero di clausole (dunque è esponenziale sul numero di clausole). Questo significa che un problema CNF può essere esponenziale.

3. Calcolo

Deduzione

Una deduzione (vista informalmente nel capitolo 1) si formalizza partendo da un insieme di formule Γ da cui vogliamo dedurre un'altra formula ϕ tramite dei passi di manipolazione puramente simbolica.

A tal scopo individuiamo il concetto di passo elementare, poiché le formule atomiche non sono decomponibili ulteriormente ci concentriamo su formule composte, decomponibili in formule più semplici. Ovviamente rispettando la semantica degli operatori booleani, ad esempio, dalla veridicità di una congiunzione posso dedurre la veridicità di tutte le componenti. Possiamo anche effettuare il processo inverso, quello di composizione, dove da formule semplici deduco formule complesse.

Esempio: $\Gamma \vDash A$ e $\Gamma \vDash B \Leftrightarrow \Gamma \vDash A \wedge B$

Queste regole di deduzione sono conosciute anche come regole di inferenza (regole che generano nuove formule a partire da altre formule). La deduzione deve essere coerente con il concetto di conseguenza logica, quindi deve essere sempre giustificata sintatticamente.

Proprietà delle regole di inferenza

Le regole di inferenza preservano la verità.

Supponiamo di avere una regola ϕ che produce ψ: $\frac{\phi}{\psi}$. Questo significa che se è vera ϕ è vera anche ψ (il viceversa non sussiste in tutte le situazioni).
In simboli (si noti che è praticamente una forma ristretta della conseguenza logica):

$$\frac{\boldsymbol{\phi}}{\boldsymbol{\psi}} \quad \forall \boldsymbol{m} \ (\boldsymbol{m} \vDash \boldsymbol{\phi} \Rightarrow \boldsymbol{m} \vDash \boldsymbol{\psi})$$

Regole di inferenza

Un calcolo corretto si ottiene definendo delle regole. Una regola si ottiene sfruttando la semantica degli operatori. Descriveremo in seguito alcune regole di decomposizione e composizione.

Congiunzione

- And elimination left: $\frac{A \wedge B}{A}$ $\wedge$el
- And elimination right: $\frac{A \wedge B}{B}$ $\wedge$er

Implicitamente queste regole ci dicono che se in un certo punto mi trovo ad aver dedotto $A \wedge B$ allora posso produrre la veridicità di una singola proposizione. Mentre, se ho due proposizioni vere allora posso dedurre la veridicità della congiunzione.

- And introduction: $\frac{A \quad B}{A \wedge B}$ $\wedge$i

Disgiunzione

Prendiamo la regola dell'or: $\Gamma \vDash A \vee B \Leftrightarrow \Gamma \vDash A$ o $\Gamma \vDash B$.

- Or introduction left: $\frac{A}{A \vee B}$ $\vee$il
- Or introduction right: $\frac{B}{A \vee B}$ $\vee$ir

Se ho dedotto una delle due formule allora posso dedurre anche la disgiunzione. Ovviamente l'implicazione verso destra ($A \vee B$) non porta ad ottenere la certezza della veridicità delle componenti.

Supponiamo di dedurre qualcosa da A (assunto A vero), chiamiamo C questa deduzione, e supponiamo di poter ottenere C anche se assunto B. Se le assunzioni sono indipendenti posso catturare il fatto che $\frac{A \vee B}{C}$. Diciamo che se in un ragionamento deduco qualcosa in comune da due diverse assunzioni allora posso dedurre tale cosa anche solo se una delle due è vera.

Quando applico questa regola, chiamata *or elimination*, posso dimenticarmi delle due assunzioni (che vengono messe tra quadre) poiché vengono raccolte dalla disgiunzione. In simboli:

$\Gamma, A \vDash C$ (Γ ed A sono conseguenza logica di C) e $\Gamma, B \vDash C$ allora $\frac{\begin{matrix}[B] & [A] \\ \vdots & \vdots \\ C & C\end{matrix} \quad A \vee B}{C}$. Le parentesi quadre mi dicono che è possibile "scaricare" la formula all'interno (dalle assunzioni che mi hanno portato C, posso eliminare o A oppure B).

- Or elimination: $\frac{\begin{matrix}[B] & [A] \\ \vdots & \vdots \\ C & C\end{matrix} \quad A \vee B}{C}$ Ve

Le assunzioni mi permettono di dire che $\Gamma \vDash \phi \Rightarrow \Gamma \cup \Gamma' \vDash \phi$. Quindi tutto ciò che deduco nella mia situazione la suppongo vera in ogni situazione del mondo (sembra strano ma è l'unico modo per poter "allargare" le mie deduzioni).

Implicazione

$\Gamma \vDash A \to B \Leftrightarrow \Gamma \nvDash A$ o $\Gamma \vDash B$. Questo ci permette di dire che da $\frac{B}{A \to B}$ ma è legittimo anche $\frac{\begin{matrix}[A] \\ \vdots \\ B\end{matrix}}{A \to B}$ (assumendo A, arrivo a B allora AimplicaB). Praticamente $\Gamma, A \vDash B \Leftrightarrow \Gamma \vDash \mathrm{A} \to \mathrm{B}$

- Introduzione dell'implicazione (imply introduction): $\frac{\begin{matrix}[A] \\ \vdots \\ B\end{matrix}}{A \to B} \to$i

Se ho ottenuto A ed ho ottenuto che $A \to B$ allora ho ottenuto anche B

- Eliminazione dell'implicazione: $\frac{A \qquad A \to B}{B} \to$e

Questa regola ricorda quella di decomposizione: $\Gamma \vDash \psi$ e $\Gamma, \psi \vDash \phi$ quindi $\Gamma \vDash \phi$.

Negazione

L'interpretazione dominante sulla negazione è quella che la lega all'inconsistenza. Quella meno dominante la lega alla non derivabilità. Quindi il problema è capire quando si è giustificati a derivare la negazione di qualcosa. Nella definizione semantica la negazione è la falsità, quindi è legato al concetto di derivabilità (se una formula non è derivabile allora sarà derivabile la usa negata) che è un concetto non formalizzabile. Le regole per la negazione, per suddetto motivo sono legate dunque al concetto di consistenza, introducendo il simbolo logico $\bot$ ("bot" che è l'opposto di $\top$ "top") che definisce la sua falsità in ogni modello (quindi $\top$ è vero in ogni modello).
Questi connettivi logici sono descrivibili come segue: $\bot \equiv A \wedge \neg A \qquad \top \equiv A \vee \neg A$.

- Not elimination $\frac{A \quad \neg A}{\bot} \neg$e (sono praticamente in uno stato inconsistente)
- Bottom elimination $\frac{\bot}{A} \bot$e

Nel momento in cui abbiamo dedotto l'inconsistenza sono legittimato a produrre qualsiasi cosa $\frac{\bot}{A}$ (poiché tutto è implicabile da un insieme sempre falso, non abbiamo modelli che possono soddisfare le premesse)

- Riduzione all'assurdo: $\dfrac{\begin{matrix}[\neg A]\\ \vdots \\ \bot\end{matrix}}{A}$ RAA (Redutium Ad Absurdum)

Assumendo $\neg A$ arrivo ad un'inconsistenza e quindi posso dire che A sia vera.

- Not introduction: $\dfrac{\begin{matrix}[A]\\ \vdots \\ \bot\end{matrix}}{\neg A}$ ¬i (sorta di duale del precedente)

Una deduzione è intuitivamente una applicazione ripetuta di tali regole. Che ϕ è deducibila da Γ viene descritto nel seguente modo $\boldsymbol{\Gamma \vdash \phi}$ (tramite manipolazioni simboliche posso ottenere ϕ da Γ)

Teorema di deduzione (versione semantica)

$\boldsymbol{\Gamma, A \vDash B \Leftrightarrow \Gamma \vDash A \rightarrow B}$

Dimostrazione

$\forall m \quad m \vDash \bigwedge \Gamma \wedge A \Rightarrow m \vDash B$ e abbiamo dimostrato anche che $(\phi_1 \wedge \phi_2) \rightarrow \phi_3$.
Dunque, $\forall m \quad m \vDash \bigwedge \Gamma \Rightarrow m \vDash A \Rightarrow m \vDash B$. Ma $m \vDash A \Rightarrow m \vDash B$ è equivalente a $m \vDash A \rightarrow B$ ora guardando tutto insieme: $\forall m \quad m \vDash \bigwedge \Gamma \wedge A \Rightarrow m \vDash A \rightarrow B$ è praticamente l'enunciato del teorema.

Correttezza del calcolo

$\boldsymbol{\Gamma \vdash \phi \Rightarrow \Gamma \vDash \phi}$

(se ottengo ϕ da gamma tramite deduzione allora ϕ è conseguenza logica di gamma).

Se le regole preservano la verità la correttezza è sempre garantita (ciò non vale per la completezza)

Completezza del calcolo

$\boldsymbol{\Gamma \vDash \phi \Rightarrow T \vdash \phi}$

(tutto ciò che è valido può essere dimostrato).

Definizione di Deduzione

Una deduzione è una sequenza di formule. Una deduzione di ϕ da Γ (insieme di formule) è una sequenza finita (lunga n) di formule tali che

- La formula in posizione n (l'ultima formula della sequenza) è ϕ
- $\forall i < n$ (tutte le altre) $\phi_i \in \Gamma$ oppure è una conclusione di una regola di inferenza le cui premesse hanno indice $< i$

Osservazione

La deduzione è un albero le cui foglie sono le foglie di Γ, i nodi interno è ottenuto applicando una regola di inferenza con condizione i figli; la radice sarà la nostra conclusione (va letto dal basso verso l'alto)

Applicazioni delle regole di inferenza

$$\frac{A \wedge B}{A}\wedge\text{el} \qquad \frac{A \wedge B}{B}\wedge\text{er} \qquad \frac{A \quad B}{A \wedge B}\wedge\text{i} \qquad \frac{A}{A \vee B}\vee\text{il} \qquad \frac{B}{A \vee B}\vee\text{ir} \qquad \frac{\begin{matrix}[B] & [A] \\ \vdots & \vdots \\ C & C\end{matrix} \quad A \vee B}{C}\vee\text{e}$$

$$\frac{\begin{matrix}[A] \\ \vdots \\ B\end{matrix}}{A \to B}\to\text{i} \qquad \frac{A \quad A \to B}{B}\to\text{e} \qquad \frac{A \quad \neg A}{\bot}\neg\text{e} \qquad \frac{\bot}{A}\bot\text{e} \qquad \frac{\begin{matrix}[\neg A] \\ \vdots \\ \bot\end{matrix}}{A}\text{RAA} \qquad \frac{\begin{matrix}[A] \\ \vdots \\ \bot\end{matrix}}{\neg A}\neg\text{i}$$

Nota: L'introduzione dell'implicazione mi permette sempre di scaricare la premessa (formula a sinistra) per una proprietà della conseguenza logica $\Gamma \vDash C$ e $\Gamma, A \vDash C$ poiché se aggiungo premesse non posso togliere informazioni.

Oltre all'introduzione dell'implicazione ci sono altre regole che permettono di scaricare le assunzioni, e sono: RAA, ¬i e ∨e. L'idea è scaricare tutte le copie dell'assunzione che portano alla premessa, in particolare, l'or esclusivo $\frac{\begin{matrix}[B] & [A] \\ \vdots & \vdots \\ C & C\end{matrix} \quad A \vee B}{C}\vee\text{e}$ scarica le copie di A (e solo quelle) dal sottoalbero destro e le copie di B da quello sinistro. Si noti che nel caso il sottoalbero sinistro non può assumere anche A vero così come il destro non può assumere anche B vero (altrimenti sarebbero dipendenti tra loro e non potrei fare alcuna deduzione su C).

Definizione di Teorema

Dimostriamo a partire da nessuna assunzione; quindi $\Gamma \vdash \phi$; che quando $\Gamma = \emptyset$, ossia $\emptyset \vdash \phi$, (si può scrivere semplicemente con $\vdash \phi$), se ϕ è deducibile da un insieme vuote significa che è vera sempre, in deduzione ϕ viene definito teorema.

Nota: Di seguito verranno forniti vari teoremi le cui dimostrazioni avvengono tutte applicando le regole di inferenza, lo scopo è raggiungere delle proposizioni atomiche "scaricabili", così da avere l'insieme vuoto e quindi la veridicità della formula per quanto detto sopra.

Per una migliore comprensione si consiglia di leggere le regole di inferenza dal basso verso l'alto.

Teorema 1

$\vdash \big(\boldsymbol{A} \to (\boldsymbol{B} \to \boldsymbol{C})\big) \to \big((\boldsymbol{A} \to \boldsymbol{B}) \to (\boldsymbol{A} \to \boldsymbol{C})\big)$

Osservando la struttura, la radice è rappresentata da → centrale. Chiamiamo ϕ la formula, qeusta può essere prodotta da →i e quindi significa che

$$(\mathbf{1})\ \frac{(A \to B) \to (A \to C)}{\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)}\to\text{i}_1$$

Questa regola mi permette di scaricare l'assunzione $[A \to (B \to C)]_1$ poiché non è vincolante per la veridicità dell'implicazione (prendiamo tutto ciò che antecede l'implicazione poiché se assunto vero allora sarà vera anche l'implicata, dunque con una implicazione è facile "pescare" l'assunzione)

$$(\mathbf{2})\ \frac{A \to C}{(\mathbf{1})}\to\text{i}_2 \qquad \text{con cui scarico} \quad [A \to B]_2$$

La quale è ancora un'implicazione che mi porta a scaricare $[A]_3$:

$$(\mathbf{3})\ \frac{C}{(\mathbf{2})}\to\text{i}_3$$

A questo punto, se riusciamo a dimostrare vera la C, utilizzando le assunzioni scaricabili, avremmo dimostrato la validità del teorema. Con la $[A \to (B \to C)]_1$ e la $[A]_3$, sfruttando la regola dell'eliminazione dell'implicazione otteniamo:

$$\frac{[A \to (B \to C)]_1 \qquad [A]_3}{B \to C} \to\text{e}$$

Analogamente, sfruttando $[A \to B]_2$ e $[A]_3$:

$$\frac{[A \to B]_2 \qquad [A]_3}{B} \to\text{e}$$

Da queste due conclusioni, applicando un ulteriore eliminazione dell'implicazione ottengo la veridicità di C:

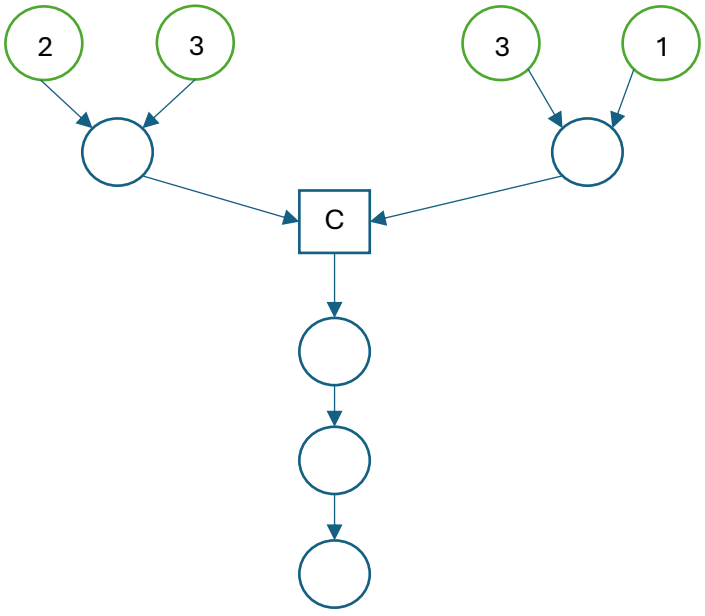
$$\frac{\dfrac{[A \to (B \to C)]_1 \qquad [A]_3}{B \to C} \to\text{e} \qquad \dfrac{[A \to B]_2 \qquad [A]_3}{B} \to\text{e}}{(3)} \to\text{e}$$

Dunque, ϕ è dimostrata, poiché le assunzioni da cui dipende non ci sono (possono essere scaricate).

Nota: La validità di questa formula si potrebbe dimostrare anche utilizzando la tabella di verità.

Di seguito riportiamo la deduzione completa con la struttura dell'albero:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[A \to (B \to C)]_1 \qquad [A]_3}{B \to C} \to\text{e} \qquad \dfrac{[A \to B]_2 \qquad [A]_3}{B} \to\text{e}}{C} \to\text{e}}{A \to C} \to\text{i}_3}{(A \to B) \to (A \to C)} \to\text{i}_2}{\big(A \to (B \to C)\big) \to \big((A \to B) \to (A \to C)\big)} \to\text{i}_1$$



Si noti che è essenziale che le assunzioni siano nelle foglie per poterle scaricare (il fatto che venga utilizzata la stessa due volte è ininfluente.

Teorema 2

$\vdash \boldsymbol{A \to (B \to A)}$

La dimostrazione è banale, infatti se A è vero $B \to A$ non può essere falso, ma, se A è falso allora la conclusione è vera. Quindi che A sia vera o falsa indipendentemente la formula sarà valida ma dimostriamolo non solo semanticamente:

$$\dfrac{\dfrac{[A]_1}{B \to A}\,{\to}\mathrm{i}}{A \to (B \to A)}\,{\to}\mathrm{i}_1$$

Questa regola mi permette di scaricare B ma in questo caso non serve poiché ci basta scaricare A per dimostrare il teorema

Teorema 3

$\vdash (\boldsymbol{A} \to \boldsymbol{B}) \to (\neg \boldsymbol{B} \to \neg \boldsymbol{A})$

Semanticamente abbiamo già dimostrato l'equivalenza (che per noi è definizione di $(A \to B) \wedge (B \to A)$) ma dimostriamolo con le regole di inferenza:

$$[A \to B]_1 \quad [\neg B]_2 \quad [A]_3$$

$$(\mathbf{1}) \quad \dfrac{\dfrac{\dfrac{\bot}{\neg A}\,\neg\mathrm{i}_3}{\neg B \to \neg A}\,{\to}\mathrm{i}_2}{(A \to B) \to (\neg B \to \neg A)}\,{\to}\mathrm{i}_1$$

Dunque, ci serve ottenere un assurdo, ma

$$\dfrac{\dfrac{[A \to B]_1 \quad [A]_3}{B}\,{\to}\mathrm{e} \qquad [\neg B]_2}{(\mathbf{1})}\,\neg\mathrm{e}$$

Teorema 4

$\vdash (\neg \boldsymbol{B} \to \neg \boldsymbol{A}) \to (\boldsymbol{A} \to \boldsymbol{B})$

Assunzioni scaricate: $[\neg B \to \neg A]_1 \quad [A]_2 \quad [\neg B]_3$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\neg B \to \neg A]_1 \quad [\neg B]_3}{\neg A}\,{\to}\mathrm{e} \qquad [A_2]}{\bot}\,\neg\mathrm{e}}{B}\,\mathrm{RAA}_3}{A \to B}\,{\to}\mathrm{i}_1}{(\neg B \to \neg A) \to (A \to B)}\,{\to}\mathrm{i}_1$$

Teorema 5

$\vdash \neg\neg \boldsymbol{A} \to \boldsymbol{A}$

$$\dfrac{\dfrac{\dfrac{[\neg\neg A]_1 \quad [\neg A]_2}{\bot}\,\neg\mathrm{e}}{A}\,\mathrm{RAA}_2}{\neg\neg A \to A}\,{\to}\mathrm{i}_1$$

Dualmente per $\vdash \boldsymbol{A} \to \neg\neg \boldsymbol{A}$

$$\dfrac{\dfrac{\dfrac{[\neg A]_1 \quad [\neg\neg A]_2}{\bot}\,\neg\mathrm{e}}{A}\,\neg\mathrm{i}_2}{\neg\neg A \to A}\,{\to}\mathrm{i}_1$$

Così abbiamo dimostrato anche che $A \equiv \neg\neg A$ in senso semantico.

Teorema 6 (distribuzione dell'or sull'and)

$\vdash \big(\boldsymbol{A} \wedge (\boldsymbol{B} \vee \boldsymbol{C})\big) \to \big((\boldsymbol{A} \wedge \boldsymbol{B}) \vee (\boldsymbol{A} \wedge \boldsymbol{C})\big)$

$$\dfrac{\dfrac{\dfrac{\dfrac{[A \wedge (B \vee C)]_1}{A}\wedge\text{el} \quad [B]_2}{A \wedge B}\wedge\text{i}}{(A \wedge B) \vee (A \wedge C)}\vee\text{il} \quad \dfrac{\dfrac{\dfrac{[A \wedge (B \vee C)]_1}{A}\wedge\text{el} \quad [C]_2}{A \wedge C}\wedge\text{i}}{(A \wedge B) \vee (A \wedge C)}\vee\text{ir} \quad \dfrac{[A \wedge (B \vee C)]_1}{B \vee C}}{(A \wedge B) \vee (A \wedge C)}\vee\text{e}_{23}}{(A \wedge (B \vee C)) \to ((A \wedge B) \vee (A \wedge C))}\to\text{i}_1$$

Teorema 7

$\vdash ((\boldsymbol{A} \wedge \boldsymbol{B}) \vee (\boldsymbol{A} \wedge \boldsymbol{C})) \to (\boldsymbol{A} \wedge (\boldsymbol{B} \vee \boldsymbol{C}))$

Dimostrare per esercizio.

Teorema 8

$\vdash \neg(\boldsymbol{A} \wedge \neg\boldsymbol{A})$

$$\dfrac{\dfrac{\dfrac{[A \wedge \neg A]_1}{A}\wedge\text{el} \quad \dfrac{[A \wedge \neg A]_1}{\neg A}\wedge\text{er}}{\bot}\neg\text{e}}{\neg(A \wedge \neg A)}\neg\text{i}_1$$

Teorema 9

$\boldsymbol{A} \vee (\boldsymbol{B} \wedge \boldsymbol{C}) \vdash (\boldsymbol{A} \vee \boldsymbol{B}) \wedge (\boldsymbol{A} \vee \boldsymbol{C})$ ossia $\vdash (A \vee (B \wedge C)) \to ((A \vee B) \wedge (A \vee C))$

$$\dfrac{\dfrac{\dfrac{(1)}{A \vee B} \quad \dfrac{(2)}{A \vee C}}{(A \vee B) \wedge (A \vee C)}\wedge\text{i}}{(A \vee (B \wedge C)) \to ((A \vee B) \wedge (A \vee C))}\to\text{i}_0$$

Dall'assunzione $[A \vee (B \wedge C)]_0$ possiamo ottenere per or introduction sia A che $B \wedge C$, dunque

$$(1)\ \dfrac{\dfrac{[A]_1}{A \vee B}\vee\text{i} \quad \dfrac{\dfrac{[B \wedge C]_2}{B}\wedge\text{i}}{A \vee B}\vee\text{i} \quad [A \vee (B \wedge C)]_0}{A \vee B}\vee\text{e}_{12}$$

E il ragionamento è duale per la (2):

$$(1)\ \dfrac{\dfrac{[A]_3}{A \vee C}\vee\text{il} \quad \dfrac{\dfrac{[B \wedge C]_4}{C}\wedge\text{i}}{A \vee C}\vee\text{ir} \quad [A \vee (B \wedge C)]_0}{A \vee C}\vee\text{e}_{34}$$

Teorema 10

$((\boldsymbol{A} \vee \boldsymbol{B}) \wedge (\boldsymbol{A} \vee \boldsymbol{C})) \vdash (\boldsymbol{A} \vee (\boldsymbol{B} \wedge \boldsymbol{C}))$

Assunzione di base: $[(A \vee B) \wedge (A \vee C)]$

$$\dfrac{\dfrac{[A]_1}{A \vee (B \wedge C)}\vee\text{i} \quad \dfrac{[B]_2 \quad [C]_3}{B \wedge C}\wedge\text{i}}{A \vee (B \wedge C)}\vee\text{i} \quad \dfrac{[(A \vee B) \wedge (A \vee C)]}{A \vee B}\wedge\text{e}\ \vee\text{e}_{12} \quad \dfrac{[A_4]}{A \vee (B \wedge C)}\vee\text{i} \quad \dfrac{[(A \vee B) \wedge (A \vee C)]}{A \vee C}\wedge\text{e}}{A \vee (B \wedge C)}\vee\text{e}_{34}$$

Teorema 11

$\vdash \boldsymbol{A} \vee \neg\boldsymbol{A}$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg A]_1}{A \vee \neg A}\vee i \qquad [\neg(A \vee \neg A)]_2}{\bot}\neg e}{A}RAA_1}{A \vee \neg A}\vee il \qquad [\neg(A \vee \neg A)]_2 \quad \dfrac{\bot}{A \vee \neg A}\neg e \quad RAA_2$$

Si noti che tale prova assume per assurdo falso il teorema, e tale assunzione mi ha portato al $\bot$ (l'ultimo) che mi permette di poter dimostrare vero $A \vee \neg A$

Di norma quando si hanno due formule (una la negata dell'altra), qualsiasi sia la formula si usa questo schema (ricordatelo che è importante).

Teorema 12

$\boldsymbol{A \to B \vdash \neg A \vee B}$

Dal teorema possiamo assumere $[A \to B]$, poi possiamo proseguire in due diversi modi:

1) Si noti che in blu c'è la dimostrazione del Teorema 11

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\neg B]_1}{B \vee \neg B}\vee i \quad [\neg(B \vee \neg B)]_2}{\bot}\wedge e}{B}RAA_1}{B \vee \neg B}\vee il \quad [\neg(B \vee \neg B)]_2 \quad \dfrac{\bot}{B \vee \neg B}\neg e \quad RAA_2 \qquad \dfrac{[B]_3}{\neg A \vee B}\vee i \qquad \dfrac{\dfrac{[A \to B]\ [A]_5}{B}\to e \quad [\neg B]_4}{\bot}\neg e \quad \dfrac{}{\neg A}\neg i_5 \quad \dfrac{}{\neg A \vee B}\vee i}{\neg A \vee B}\vee e_{34}$$

Qui si evince la necessità di decomporre le nostre dimostrazioni in più assunzioni. Infatti, in questo caso sarebbe stato più pratico fare la dimostrazione di $\neg B \wedge B$ in un'altra sezione e chiamarla magari Lemma1 così da poter alleggerire la lettura della dimostrazione totale del teorema.

2)

$$\dfrac{[\neg(\neg A \vee B)]_1 \qquad \dfrac{\dfrac{\dfrac{[\neg A]_2}{\neg A \vee B}\vee i \quad [\neg(\neg A \vee B)]_1}{\bot}\neg e}{A}RAA_2 \quad [A \to B]}{B}\to e \quad \dfrac{}{\neg A \vee B}\vee i}{\bot}\neg e \quad RAA_1$$

Teorema 13

$\boldsymbol{\neg A \vee B \vdash A \to B}$

$$\dfrac{\dfrac{\dfrac{[\neg A]_2 \quad [A]_1}{\bot}\neg e}{B}\bot e}{A \to B}\to i_1 \qquad \dfrac{[B]_3}{A \to B} \qquad [\neg A \vee B] \quad \vee e_{23}$$

Un altro modo per dimostrare questo teorema è il seguente:

$$\dfrac{\dfrac{\dfrac{[\neg A]_4 \quad [A]_1}{\bot} \qquad \dfrac{[\neg B]_2 \quad [B]_3}{\bot} \qquad [\neg A \vee B]}{\bot}\vee e_{34}}{B}RAA_2}{A \to B}\to i_1$$

Teorema 14

$\boldsymbol{\neg(A \to B) \vdash A}$

$$\dfrac{\dfrac{[\neg(A \to B)]_1 \qquad \dfrac{\dfrac{[\neg A]_1 \quad [A]_2}{\bot}\neg e}{B}\bot e}{A \to B}\to i_2}{\dfrac{\bot}{A}}RAA_1$$

Esercizi

1) $\neg(\neg A \vee B) \vdash A \wedge \neg B$

$$\dfrac{\dfrac{\dfrac{[\neg(\neg A \vee B)] \quad \dfrac{[\neg A]_1}{\neg A \vee B}\vee i}{\bot}\neg e}{A}\neg RAA_1 \qquad \dfrac{\dfrac{[\neg(\neg A \vee B)] \quad \dfrac{[B]_1}{\neg A \vee B}\vee i}{\bot}\neg e}{\neg B}\neg i_2}{A \wedge \neg B}\wedge i$$

2) $\boldsymbol{A \wedge \neg B \vdash \neg(\neg A \vee B)}$

$$\dfrac{\dfrac{[A \wedge \neg B]}{A}\wedge el \qquad \dfrac{\dfrac{\dfrac{[\neg A]_2 \quad [A]_4}{\bot}\neg e}{B}\bot e}{A \to B}\to i_4 \qquad \dfrac{[B]_3}{A \to B} \qquad [\neg A \vee B]_1}{A \to B}\vee e_{23} \qquad \dfrac{[A \wedge \neg B]}{\neg B}\wedge er}{B}\to e$$
$$\dfrac{\bot}{\neg(\neg A \vee B)}\neg e\ \neg i_1$$

Relazione tra conseguenza logica e deducibilità

Vedremo che $\vdash \subseteq \vDash$ e $\vDash \subseteq \vdash$. Ovviamente se valgono entrambe significa che ogni coppia che sta in una relazione sta anche nell'altra e viceversa.

$\vdash \subseteq \vDash$ ci dice che il calcolo è corretto mentre $\vDash \subseteq \vdash$ ci dice che il calcolo è completo (Il calcolo è $\vdash$)

La dimostrazione $\vdash \subseteq \vDash$ si basa sulla già fatta dimostrazione che le regole di inferenza preservano le verità e quindi intuitivamente preservo i modelli. Meno ovvio è dimostrare $\vDash \subseteq \vdash$ che vuol dire sostanzialmetne $\Gamma \vDash \phi \Rightarrow \Gamma \vdash \phi \quad \forall\, \Gamma \subseteq PL\ e\ \phi \in PL$

La tecnica che utilizzeremo per dimostrare la completezza del calcolo è una tecnica molto generale che ci sarà utile anche per successive dimostrazioni.

Definizione di inconsistenza

Γ è inconsistente sse $\Gamma \vdash \bot \quad \Leftrightarrow \quad \Gamma$ è inconsistente sse $\Gamma \vdash \phi \ \ \forall \phi \in PL$

Osservazioni

Se da Γ deduco bottom significa che posso dedurre qualsiasi cosa.

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \bot\end{array}}{A}\bot e$$

Se $\Gamma \vdash \phi \ \ \forall \phi$ significa che posso dedurre qualsiasi formula, quindi sia $\Gamma \vdash A$ che $\Gamma \vdash \neg A$ ma allora

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ A\end{array} \qquad \begin{array}{c}\Gamma \\ \vdots \\ \neg A\end{array}}{\bot}\neg e$$

Quindi condizione necessaria ma non sufficiente affinché Γ sia consistente è che esista almeno un ϕ non deducibile da essa.

Dimostrazione completezza del calcolo

Una formula la posso dedurre in vari modi, uno è per contrapposizione:

$$(\Gamma \vDash \phi \Rightarrow \Gamma \vdash \phi) \Leftrightarrow (\Gamma \nvdash \phi \Rightarrow \Gamma \nvDash \phi)$$

Dimostreremo quest'ultima implicazione per semplicità.

Se $\Gamma \cup \{\neg\phi\}$ è inconsistente significa che $\Gamma, \neg\phi \vdash \bot$ e quindi avendo dedotto il bottom posso dedurre ϕ per la seguente regola di inferenza:

$$\dfrac{\begin{array}{c}\Gamma, [\neg\phi] \\ \vdots \\ \bot\end{array}}{\phi}\ \text{RAA}$$

e ciò dimostra che se $\Gamma \nvDash \phi \Rightarrow \Gamma \cup \{\neg\phi\}$ è consistente.
Nota che abbiamo anche dimostrato l'equivalenza tra le seguenti relazioni:

$$(\Gamma \nvDash \phi \Rightarrow \Gamma \nvDash \phi) \Leftrightarrow (\Gamma \cup \{\neg\phi\} \text{ è consistente}) \Rightarrow \Gamma \cup \{\neg\phi\} \text{ è soddisfacibile}$$

Di conseguenza per dimostrare che $\Gamma \vDash \phi$ sse $\Gamma \cup \{\neg\phi\}$ è insoddisfacibile basta dimostrare che $\Gamma \cup \{\neg\phi\}$ è consistente $\Rightarrow \Gamma \cup \{\neg\phi\}$ è soddisfacibile (così abbiamo $\Gamma \nvDash \phi \Rightarrow \Gamma \nvDash \phi$).

Prendiamo un arbitrario $\Sigma \subseteq PL$ e dimostramio che se Σ è consistente implica che Σ è soddisfacibile (questo lo dimostriamo per ogni Σ e quindi avremmo la completezza).

Ricordiamo che un insieme di formule è soddisfacibile se esiste un modello che soddisfi tutte le formule, e quindi dobbiamo vedere che comunque sia composto Σ possiamo sempre costruire un modello che ne soddisfi le formule.

L'unica cosa che conosco di Σ è che è consistente.

Teorema di Lindenbaum

Ogni insieme Σ consistente può essere esteso ad un insieme Σ^* consistente e massimale (quindi ad un insieme non ulteriormente estendibile senza perdere la proprietà di consistenza, qualsiasi formula aggiungo in Σ^* che non c'è già lo rende inconsistente).

Osservazioni

Tale teorema è importante poiché mi dice che se parto da un insieme consistente posso allargarlo fino a renderlo consistente massimale.

Nota: il teorema non dichiara l'unicità di un insieme massimale consistente (ne posso avere diversi)

Dimostrazione del teorema

L'insieme di una sequenza infinita di simboli è numerabile. Dunque, scelgo una enumerazione infinita di formule; sia $\phi_1, \phi_2, \dots, \phi_n, \dots$ tale enumerazione.

Scegliamo un insieme $\Sigma_0 = \Sigma$, una volta definito Σ_i posso definirire un insieme Σ_{i+1} come segue

$$\Sigma_{i+1} = \begin{cases}\Sigma_i \cup \{\phi_i\} \text{ se } \Sigma_i \cup \{\phi\} \text{ è consistente} \\ \Sigma_i \qquad \text{altrimenti}\end{cases}$$

Quindi genererò per ogni ϕ_i un insieme $\Sigma_{i+1} \supseteq \Sigma_i$ (è una sequenza monotona: aggiungo qualcosa o lo lascio invariato ma non tolgo mai). Un insieme di questo tipo è sempre consistente per costruzione.

Dunque, $\forall i \geq 0 \quad \Sigma_i$ è consistente (poiché per ipotesi il caso base è consistente e rimane consistente per costruzione visto che se ϕ_i lo renderebbe inconsistente semplicemente non lo aggiungo).

A questo punto definisco $\Sigma^* = \bigcup_{i \geq 0} \Sigma_i$. Devo dimostrare la consistenza e la massimalità di Σ^*.

Dimostrazione della consistenza

Σ^* in generale non è detto che sia consistente (non necessariamente se i pezzi sono consistenti anche l'unione mantenga la proprietà), ma la proprietà di consistenza è una proprietà di tipo finitario (può essere verificata con qualcosa di finito: nel nostro caso la deduzione), quindi possiamo provare a ragionare per assurdo: assumiamo che Σ^* sia inconsistente, questo significa esiste una deduzione π tra Σ^* e $\bot$.

π è una sequenza finita di formule, quindi $\pi = \psi_1 \dots \psi_k$. Ognuna di queste formule apparirà sicuramente in un punto nell'enumerazione $\phi_1, \phi_2, \dots, \phi_n, \dots$

Sia n un indice di tale enumerazione per tutte le formule in π occorrono prima di n (quindi $\psi_1 \dots \psi_k$ vengono prima di ϕ_n). Nota che tale n esiste per la proprietà finitaria di π.

Prendiamo adesso Σ_n che sicuramente è incluso in Σ^* (per costruzione) ma se Σ^* è inconsistente avrà un bottom deducibile da π ma π è, per ipotesi, una deduzione di formule presenti già in Σ_n, dunque ciò significa che $\psi_1 \dots \psi_k$ sono anche in Σ_n e quindi anche questo insieme deduce il bottom; ne consegue l'inconsistenza (che è assurdo poiché Σ_n è consistente per costruzione).

Dimostrazione della massimalità

Resta ora da dimostrare la massimalità. Dimostriamo a tal scopo che per ogni formula $\phi \in PL$ abbiamo che $\phi \in \Sigma^*$ oppure $\neg\phi \in \Sigma^*$ (ovviamente non possono esserci entrambe poiché abbiamo appena dimostrato la consistenza).

Assumiamo $\phi \notin \Sigma^*$ e $\neg\phi \in \Sigma^*$, ma per costruzione di Σ^* se ϕ non è all'interno di Σ^* significa che esisteva già un ϕ_i per cui $\Sigma_i \cup \{\phi\}$ è inconsistente (per questo non l'ho messo all'interno di Σ^*). Il discorso è analogo per l'assunzione $\neg\phi \notin \Sigma^*$ e $\phi \in \Sigma^*$.

Assumiamo adesso per assurdo che $\phi \notin \Sigma^*$ e $\neg\phi \notin \Sigma^*$, quindi esiste un i all'interno dell'enumerazione tale che $\phi_i = \phi$ e $\Sigma_i, \phi_i \vdash \bot$ e quindi $\Sigma_{i+1} = \Sigma_i$. Allo stesso modo esiste un $j > i$ per il quale $\phi_j = \neg\phi$ e $\Sigma_j, \phi_j \vdash \bot$.

Considerata tale premessa, e considerato $\Sigma_i \subseteq \Sigma_j$ possiamo concludere

$$\dfrac{\dfrac{\begin{array}{c}\Sigma_j, [\neg\phi] \\ \vdots \\ \bot\end{array}}{\phi}\,\text{RAA} \qquad \dfrac{\begin{array}{c}\Sigma_j, [\phi] \\ \vdots \\ \bot\end{array}}{\neg\phi}\,\neg\text{i}}{\bot}\,\neg\text{e}$$

Ma tale deduzione indica che da Σ_j è deducibile il bottom; dunque, non avrei potuto dedurre Σ_j per costruzione. Quindi uno tra ϕ e $\neg\phi$ deve essere presente all'interno di Σ^*.
Questo ragionamento dimostra anche la massimalità di Σ^*

Proprietà di Σ^*

1) $\phi_1 \wedge \phi_2 \in \Sigma^* \Leftrightarrow \phi_1 \in \Sigma^*$ e $\phi_2 \in \Sigma^*$
2) $\phi_1 \vee \phi_2 \in \Sigma^* \Leftrightarrow \phi_1 \in \Sigma^*$ o $\phi_2 \in \Sigma^*$

Dimostrazione 1) $\Rightarrow$

$\phi_1 \wedge \phi_2 \in \Sigma^*$ ma $\phi_1 \notin \Sigma^*$. Esiste una Σ_j tale che

$$\dfrac{\dfrac{\begin{array}{c}\Sigma_j, [\phi_1] \\ \vdots \\ \bot\end{array}}{\neg\phi_1}\neg\text{i} \qquad \dfrac{\phi_1 \wedge \phi_2}{\phi_1}\wedge\text{i}}{\bot}\neg\text{e}$$

Ma così ho costruito una deduzione che raggiunge in bottom solo da Σ_j e quindi ho dimostrato che $\Sigma_j, \phi_1 \wedge \phi_2 \vdash \bot$ il che è assurdo.

Estrazione di un modello da Σ^*

Da Σ^* posso estrarre banalmente un modello per dimostrare che se un modello è consistente allora è soddisfacibile. Il modello lo estraggo nel seguente modo:

Dato Σ^* massimale consistente, posso definire un $M_{\Sigma^*} = \{A \in AP | A \in \Sigma^*\}$, quindi adesso ci servirebbe dimostrare che M_{Σ^*} sia un modello che soddisfa tutte le formule di Σ, e quindi che l'insieme è soddisfacibile (dimostrando dunque la completezza).

Dimostriamo $M_{\Sigma^*} \vDash \Sigma$, anzi dimostreremo che M_{Σ^*} deriva tutte le formule derivabili da Σ (che è una assunzione più forte) ovvero che se $\Sigma \vdash \phi \rightarrow M_{\Sigma^*} \vDash \phi$.

Se Σ è consistente allora esiste un modello m per cui $\forall\phi \;\; m \vDash \phi \Leftrightarrow \phi \in \Sigma^*$.

Dimostriamo per induzione sulla struttura delle formule (visto che ho $\forall\phi$). Definiamo $m = M_{\Sigma^*}$ e ricordiamo che $M_{\Sigma^*} = \{A \in AP | A \in \Sigma^*\}$ allora è ovvio che se $\phi \in AP$ l'equivalenza è valida.

Invece, se $\phi = \neg\psi$ allora procediamo per induzione. L'ipotesi induttiva è $m \vDash \psi \Leftrightarrow \psi \in \Sigma^*$; negando tale ipotesi abbiamo ovviamente

$$\begin{array}{ccc} m \nvDash \psi & \Leftrightarrow & \psi \notin \Sigma^* \\ \Updownarrow & & \Updownarrow \\ m \vDash \underbrace{\neg\psi}_{\phi} & \Leftrightarrow & \underbrace{\neg\psi}_{\phi} \in \Sigma^* \end{array}$$

Così abbiamo dimostrato la $\Rightarrow$. Nota: le $\Updownarrow$ sono vere per massimalità di Σ^*.

Se $\phi = \psi_1 \wedge \psi_2$ allora significa che $(m \vDash \psi_1 \text{ e } m \vDash \psi_2) \Leftrightarrow (\psi_1 \in \Sigma^* \text{ e } \psi_2 \in \Sigma^*) \Rightarrow m \vDash \phi$.

Se $\phi = \psi_1 \vee \psi_2$ allora significa che $(m \vDash \psi_1 \text{ o } m \vDash \psi_2) \Leftrightarrow (\psi_1 \in \Sigma^* \text{ o } \psi_2 \in \Sigma^*) \Rightarrow m \vDash \phi$.

Se $\phi = \psi_1 \rightarrow \psi_2$ allora significa che $(m \nvDash \psi_1 \text{ o } m \vDash \psi_2) \Leftrightarrow (\psi_1 \notin \Sigma^* \text{ o } \psi_2 \in \Sigma^*) \Rightarrow m \vDash \phi$.

Così per tutti i casi abbiamo dimostrato che se Σ è consistente esiste un modello che lo soddisfa.

Conclusioni

Abbiamo dimostrato che se $\Gamma \cup \{\neg\phi\}$ è consistente allora $\exists m \subseteq 2^{AP} \;\; m \vDash \Gamma \cup \{\neg\phi\}$

Il che significa che $\Gamma \cup \{\neg\phi\}$ è soddisfacibile.

Grazie al teorema di Lindenbaum partendo da questo insieme possiamo costruire una saturazione (insieme massimale consistente) che ha tutte le proprietà che ha un modello tranne quello di essere sottoinsieme 2^{AP} ma questo si raggiunge semplicemente togliendo tutto ciò che non è una proposizione atomica.

Dimostrazione della correttezza del calcolo

$$\boldsymbol{\Gamma \vdash \phi \Rightarrow \Gamma \vDash \phi} \qquad \boldsymbol{\forall \Gamma \subseteq PL \text{ e } \phi \in PL}$$

Dimostreremo in generale che una qualsiasi dimostrazione π che varia da un certo Γ a ϕ allora $\Gamma \vDash \phi$

$$\Gamma \vdash \phi \Rightarrow \forall \pi = \begin{matrix}\Gamma \\ \vdots \\ \phi\end{matrix} \Rightarrow \Gamma \vDash \phi$$

Procederemo per induzione sulla lunghezza della derivazione di π (sulla profondità della struttura dell'albero di derivazione).

Per $|\pi| = 1 \Rightarrow \phi \in \Gamma \Rightarrow \Gamma \vDash \phi$ l'ultima implicazione è ovvia, infatti per definizione: $\Gamma \vDash \phi$ vale $\forall m$ $(m \vDash \Gamma$ allora $m \vDash \phi)$.

Per $|\pi| > 1$ significa (immaginala come sequenza) che $\pi = \phi_1 \dots \phi_k$ con $k = |\pi|$, quindi $\phi_k = \phi$ che o $\phi \in \Gamma$ o $\phi \notin \Gamma$. Per $\phi \in \Gamma$ vale lo stesso ragionamento di prima (la definizione). Ma se $\phi \notin \Gamma$ allora ϕ è ottenuto da regola di inferenza e quindi bisogna ragionare regola per regola:

- $$\dfrac{\left.\begin{matrix}\Gamma \\ \vdots \\ \psi\end{matrix}\quad\begin{matrix}\Gamma \\ \vdots \\ \psi \to \phi\end{matrix}\right\}<\mathrm{k}}{\phi}\ \to\mathrm{e}$$
  ma poiché le deduzioni sono di lunghezza inferiore a ϕ_k posso procedere per induzione.
  - Ipotesi induttiva: $\Gamma \vDash \psi$ e $\Gamma \vDash \psi \to \phi$;
  - ma se queste sono vere allora per definizione della semantica dell'implicazione, ψ non può essere falsa:
    $$\Gamma \vDash \psi \equiv \forall \mathrm{m}\ \ \mathrm{m} \vDash \Gamma \Rightarrow \mathrm{m} \vDash \psi$$
    $$\Gamma \vDash \psi \to \phi \equiv \forall \mathrm{m}\ \ \mathrm{m} \vDash \Gamma \Rightarrow \mathrm{m} \vDash \psi \to \phi$$
  - preso m arbitrario tale che $m \vDash \Gamma$ abbiamo che vale $m \vDash \psi$ e $m \vDash \psi \to \phi$ e quindi, ovviamente, $m \vDash \phi$ per la semantica dell'implicazione
- $$\dfrac{\begin{matrix}\Gamma \\ \vdots \\ \psi\end{matrix}\quad\begin{matrix}\Gamma \\ \vdots \\ \neg\psi\end{matrix}}{\bot}\ \neg\mathrm{e}$$
  quindi $\phi = \bot$ e analogamente a prima poiché le condizioni sono minori di k:
  - ipotesi induttiva $\Gamma \vDash \psi$ e $\Gamma \vDash \neg\psi$
  - $\forall m\ \ m \vDash \Gamma \Rightarrow m \vDash \psi$
    $\forall m\ \ m \vDash \Gamma \Rightarrow m \vDash \neg\psi$
  - Dato $m \subseteq AP$ arbitrario se $m \vDash \Gamma \Rightarrow \begin{matrix} m \vDash \psi \\ m \vDash \neg\psi \end{matrix}$ ma ciò significa che Γ è insoddisfacibile, quindi $\Gamma \vDash \bot$
- $$\left.\dfrac{\left.\begin{matrix}\Gamma, [\neg\phi] \\ \vdots \\ \bot\end{matrix}\right\}<\mathrm{k}}{\phi}\ \mathrm{RAA}\right] = k$$
  - Per ipotesi induttiva abbiamo che $\Gamma, \neg\phi \vDash \bot$. Tale $\bot$ potrebbe dipendere solo da Γ o dal fatto che $\neg\phi$ contraddice qualsoca in Γ (soddisfacibile)
  - Caso1: $\Gamma \vDash \bot \Rightarrow \Gamma \vDash \phi$
  - Caso2: $|\{\ \exists m | m \vDash \Gamma\}| > 0$ ma nessuno di questi può soddisfare $\neg\phi$ (altrimenti avremmo il bottom) e quindi $\forall m\ \ m \vDash \Gamma \Rightarrow m \nvDash \neg\phi \Rightarrow m \vDash \phi \Rightarrow \Gamma \vDash \phi$
- Le altre si lasciano per esercizio

Teorema di compattezza

Per ogni $\Gamma \subseteq PL$ e $\phi \in PL$ vale $\Gamma \vDash \phi$ sse esiste un sottoinsieme $\Gamma' \subseteq \Gamma$ con Γ' finito tale che $\Gamma' \vDash \phi$.

Osservazioni

Questo teorema ci dice che esiste un sottoinsieme finito di premesse per cui ϕ è conseguenza logica. Quindi è sufficiente un pezzo finito dell'insieme Γ. Per verificare la conseguenza logica possiamo concentrarci su un sottoinsieme finito.

Dimostrazione

Verso $\Leftarrow$ ovvio per monotonicità di $\vDash$ (rispetto ovviamente all'insieme Γ). Mentre per il verso $\Rightarrow$ ci appoggiamo alle dimostrazioni svolte per la completezza di Σ^*.

$\Gamma \vDash \phi \Rightarrow \Gamma \vdash \phi \Rightarrow \exists \pi$ con foglia in Γ e radice ϕ; ma π è un albero di derivazione finito e dunque Γ' sarà composto esattamente dalle foglie di π e quindi $\Gamma' \vdash \phi \Rightarrow \Gamma' \vDash \phi$ (per la dimostrazione precedente)

$$\begin{array}{ccccc} \psi_1 & \dots & \psi_k & & \\ & \searrow & \swarrow & & \\ & & \phi & & \end{array} \qquad \underbrace{\{\psi_1, \dots, \psi_k\}}_{\Gamma'} \subseteq \Gamma$$

Utilizzo

Il teorema di compattezza è tipicamente usato per dire che un linguaggio non è tanto espressivo per definire determinati oggetti.

4. Risoluzione

Con calcolo di risoluzione si intende un calcolo con una sola regola di inferenza (molto più semplice) che può essere facilmente automatizzato ed è facile dimostrarne la terminazione. Inoltre, si applica a formule di una certa forma, tale forma è fissata e ogni formula può essere inserita in tale forma (come le forme normali).

La risoluzione cerca di verificare se un insieme di formule è inconsistente (quindi non verifica direttamente la conseguenza logica, ma sappiamo che questa può essere formulata in termini di insoddisfacibilità o inconsistenza). L'obiettivo è dunque ottenere il bottom da un insieme di formule.

L'idea è prendere un insieme di formule Γ e costruire la Forma Clausale di Γ.

Forma Clausale

Una forma clausale è un insieme di sottoinsiemi di letterali. Si ricorda che un letterale è o una proposizione atomica o la sua negazione: $l \in \{P, \neg P | P \in AP\}$

Praticamente si trasforma la clausola disgiuntiva della CNF in un insieme di letterali (così da far collassare le parti ridondanti) in questo modo si ha una congiunzione di insiemi di letterali (che sarà il nostro insieme di sottoinsiemi).

Ad esempio: $\phi = (p \vee \neg q \vee r) \wedge (q \vee \neg r \vee p \vee q)$
$FC(\phi) = \{\{p, \neg q, r\}, \{q, \neg r, p\}\}$

L'algoritmo di risoluzione prende questo oggetto come input e vede se è inconsistente.

Ogni Γ può essere trasformato in una FC

Il fatto che ogni insieme Γ può essere posto nella forma FC è ovvio; infatti, abbiamo già dimostrato che ogni formula può essere trasformata in una formula CNF e per ogni CNF esiste una sua traduzione in FC (si sostituiscono gli and con delle virgole e si prende l'insieme delle clausole disgiuntive mettendo le virgole al posto degli or e togliendo i doppioni, come fatto nell'esempio precedente).

Esempio di trasformazione

$\Gamma = \{(p \to (q \wedge r)), (q \vee (q \to p)), ((\neg p \to q) \vee (\neg q \to \gamma))\}$
$\phi_\Gamma = \big((p \to (q \wedge r)) \wedge (q \vee (q \to p)) \wedge ((\neg p \to q) \vee (\neg q \to \gamma))\big)$
$CNF(\phi_\Gamma) = \big(((\neg p \vee q) \wedge (\neg p \vee r)) \wedge (q \vee \neg q \vee p) \vee (p \vee q \vee q \vee r)\big)$
$FC(\Gamma) = \{\{\neg p, q\}, \{\neg p, r\}, \{p, q, r\}\}$

Per esercizio si possono esplicitare tutti i passaggi per la $CNF(\phi_\Gamma)$.

Definizioni e prime proprietà

Ciascun insieme di letterali è chiamata clausola (che corrisponde ad una clausola disgiuntiva della CNF).

Una clausola è soddisfacibile se esiste almeno un letterale soddisfacibile (così da avere un modello che soddisfa tale letterale).

Clausola unitaria

Una clausola con un solo letterale $\{l\}$ è chiamata clausola unitaria (insieme di cardinalità 1)

Clausola vuota

Una clausola vuota { } è indicata con il simbolo □ ed è la clausola sempre insoddisfacibile (non avendo nemmeno un letterale non può esistere qualcosa di soddisfacibile).

Clausola banale

Clausola banale $\{p, \neg p\}$ è la clasuola più facile da soddisfare (sempre valida) poiché contenendo p e $\neg p$ è ovviamente sempre vera.

Soddisfacibilità di una FC

Sia ϕ una forma clausale se ϕ contiene una clausola banale θ la soddisfacibilità dipenderà solo dalle altre e quindi ϕ è $sat \Leftrightarrow \phi \setminus \{\theta\}$ è sat.

Dunque, supponiamo adesso che una forma clausale non abbia una clausola banale poiché possiamo semplicemente buttarla via (si ricorda che il nostro scopo è trovare l'insoddisfacibilità).

Se ϕ contiene la clausola vuota □ allora è inconsistente poiché $\psi \vee \bot \equiv \bot$ e quindi se ottengo una clausola vuota da un insieme di clausole posso dire che tale insieme è inconsistente, se non lo ottengo allora posso dire che sia consistente.

La regola di risoluzione

Diciamo che due clausole C_1 e C_2 confliggono (*clash*) su un letterale l se $l \in C_1$ e $\neg l \in C_2$.

$C_1 = \{p, \Theta\}, C_2 = \{\neg p, \Theta'\}$ l'informazione di p ai fini dell'inconsistenza è irrilevante (poiché soddisferà o una o l'altra) quindi per completare la soddisfacibilità devo verificare che le altre proposizioni Θ o Θ' siano soddisfacibili. Dunque, posso unire le due clausole come segue: $R = \{\Theta, \Theta'\}$ (se ho un modello che soddisfa R allora avrò un modello che soddisfa C_1 e C_2)

Definizione formale

Date C_1 e C_2 che confliggono sul letterale l (assumendo $l \in C_1$ e $\neg l \in C_2$) allora

$$\frac{C_1 \qquad C_2}{R}\,\text{RES} \quad \text{dove} \quad R = (C_1 \setminus \{l\}) \cup (C_2 \setminus \{\neg l\})$$

(dalle due clausole è possibile derivare una nuova clausola R)

Dimostrazione della correttezza

Giustifichiamo la correttezza di questa regola. La dimostrazione che questa regola abbia senso si può fare in due modi; uno semantico e uno sintattico

Dimostrazione sintattica

Trasformiamo le clausole in formule, così da poter applicare le regole di deduzione che conosciamo, Trasformiamo ad esempio la seguente istanza di regola di riduzione

$$\frac{\{p, r\} \qquad \{\neg p, q\}}{\{r, q\}}$$

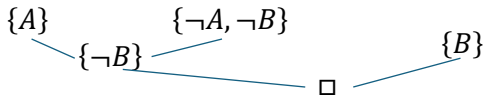
in deduzione naturale significa che da $p \vee r$ e $\neg p \vee q$ vogliamo ottenere la tesi $r \vee q$. La dimostrazione è la seguente:

$$\dfrac{\dfrac{\dfrac{p \vee r \qquad \dfrac{\pi_1}{\neg p \to r} \qquad [\neg p]_1}{r}\to e}{r \vee q}\vee i \qquad \dfrac{\dfrac{\dfrac{\neg p \vee q \qquad \dfrac{\pi_1}{\neg p \to q} \qquad [p]_1}{q}\to e}{r \vee q}\vee i \qquad \dfrac{\pi_3}{p \vee \neg p}}{r \vee q}\vee e_{12}$$

Dove π_1, π_2 e π_3 sono deduzioni corrette (vedi Teorema 13 del capitolo sulle applicazioni delle regole di inferenza); e grazie a tali deduzioni questa dimostrazione è corretta dal punto di vista sintattico. Si noti che questa dimostrazione non è nient'altro che una dimostrazione per casi; infatti, se prendo un modello che assume $\neg p$ allora ottengo r e quindi $r \vee q$, analogamente se prendo p ottengo comunque r e quindi $r \vee q$.

La conseguenza logica $\Gamma \vDash \phi \Leftrightarrow \Gamma \cup \{\neg\phi\}$ è $unsat$ che è un calcolo completo ma non è detto che succeda che se conosco Γ allora posso dedurre ϕ tramite regole di risoluzione; infatti, il calcolo di riduzione va a dimostrare l'insoddisfacibilità di una formula (cercando la clausola vuota).

Ad esempio: $\begin{cases}\Gamma = \{A, B\} \\ \phi \triangleq A \wedge B\end{cases}$ $A, N \vdash A \wedge B$ non c'è verso che si possa fare tramite il calcolo di risoluzione poiché non posso creare congiunzioni a partire da A, B (non posso ottenere la forma clausale) ma se parto da A, B e dalla congiunzione $\neg A \vee \neg B$ posso ovviamente ottenere la clausola vuota e quindi dalle clausole $\{A\}, \{B\}$ e $\{\neg A, \neg B\}$ risulta



Non è pensato per la conseguenza logica ma per la sua formalizzazione alternativa: la insoddisfacibilità

Dimostrazione semantica

Possiamo dimostrare che per ogni C_1, C_2 clausole che confliggono su qualche letterale l abbiamo

1) Se C_1 e C_2 sono soddisfatte dallo stesso modello m, allora R è soddisfatta da m (questa regola preserva la verità, quindi è corretta nel senso classico)
2) Se R è soddisfacibile, allora C_1 o C_2 sono soddisfacibili dallo stesso modello m (se parto da un insieme insoddisfacibile allora la clausola vuota deve essere ottenibile, ci da un senso di completezza)

(si noti che la 2 non è l'inverso della 1, si noti i termini soddisfatti e soddisfacibili. Infatti, la 2 non implica che il modello m soddisfi sia C_1 e C_2, o meglio, esistono modelli che soddisfano solo una o l'atra e altri modelli che li soddisfino entrambi).

Partiamo dalla **1**. Supponiamo esista un assegnamento α per C_1 e C_2 tale che $val^\alpha(C_1) = 1$ e $val^\alpha(C_2) = 1$. Supponiamo che $l \in C_1$ e $\neg l \in C_2$ (per ipotesi C_1 e C_2 confliggono), e consideriamo i due casi possibili (da cui dobbiamo ottenere R soddisfacibile da α)

a) Se $val^\alpha(l) = 1$ allora $val^\alpha(\neg l) = 0$, ma se $val^\alpha(\neg l) = 0$ allora deve esistere un letterale $l' \in C_2$ con l' diverso da $\neg l$ per cui $val^\alpha(l') = 1$ (pociché per ipostei $val^\alpha(C_2) = 1$).
   Ma $l' \in R \Rightarrow val^\alpha(R) = 1$
b) Se $val^\alpha(l) = 0$ (ragionamento duale): deve esistere un letterale $l' \in C_1$ con $l' \neq l$ tale che $val^\alpha(l') = 1$ ma essendo $l' \in R \Rightarrow val^\alpha(R) = 1$

Punto **2**. Per ipotesi esiste un α per R tale che $val^\alpha(R) = 1$, quindi per definizione $\alpha: AP(R) \to \{0,1\}$ $val^\alpha(R) = 1$ se e soltanto se $\exists l' \in R$ tale che $val^\alpha(l') = 1$; ora l' può appartenere a C_1, C_2 od a

entrambe. In ogni caso C_1 e C_2 hanno in più di R il letterale confliggente l; quindi non sono soddisfacibili da α. Dunque, generiamo un altro assegnamento α' che soddisfi sia C_1 che C_2.

Supponiamo $l \in C_1$ e $\neg l \in C_2$ e sia $\alpha' : AP(C_1 \cup C_2) \to \{0,1\}$ (per essere un assegnamento rispetto al quale posso valutare C_1 e C_2 devo assegnare ad ogni proposizione nell'insieme $C_1 \cup C_2$ e non solo di $R \subset C_1 \cup C_2$). L'uguaglianza è da escludere proprio per il letterale che confligge)

Esplicitiamo i tre casi descritti in precedenza:

a) $l' \in C_1 \cap C_2$ allora $\forall p \in AP(C_1 \cup C_2)$ vale $\alpha'(P) = \begin{cases} 0 \; o \; 1 \; se \; p \in AP(l) \\ \alpha(p) \;\; se \; p \in AP(R) \end{cases}$
(quindi se $p \in AP(R)$ mantengo quel valore, altrimenti è indifferente)

b) $l' \in C_1 \setminus C_2$ allora $\forall p \in AP(C_1 \cup C_2)$ vale $\alpha'(P) = \begin{cases} 0 \quad\; se \; l = p \\ \alpha(p) \;\; se \; p \in AP(R) \end{cases}$
(poiché $val^{\alpha'}(C_1) = 1$ perché $val^{\alpha'}(l') = 1 \; e \; l' \in C_1$, mentre $val^{\alpha'}(C_2) = 1$ perché $val^{\alpha'}(\neg l) = 1$ e $\neg l \in C_2$)
Nota che l è quello che non sta in R ma in C_2

c) $l' \in C_2 \setminus C_1$ allora $\forall p \in AP(C_1 \cup C_2)$ vale $\alpha'(P) = \begin{cases} 1 \quad\; se \; l = p \\ \alpha(p) \;\; se \; p \in AP(R) \end{cases}$

Algoritmo

Avendo una unica regola è possibile definire un algoritmo che ripetutamente applica la suddetta regola.

Se fosse soddisfacibile allora l'algoritmo non dovrebbe riuscire a trovare la clausola vuota, ma questo ovviamente non garantisce la terminazione dell'algoritmo. Perché se ottenessi la clausola vuota allora significa che la formula è insoddisfacibile. Quindi bisognerà definire una condizione di terminazione oltre alla clausola vuota.

L'algoritmo è il seguente:

```
Resolution(S) // S insieme di clausole
  E = ∅ // coppie di clausole usate
  S0 = S  // clausole a disposizione

  repeat
    Pick from S0 clausole C1 e C2 dove {C1,C2} ∉ E
    R = Resolvent(C1,C2) // regola di riduzione
    if R not trivial // non contiene letterali opposti (clausola non banale)
      S0 = S0 ∪ {R}
    E = E ∪ {C1,C2}
  until R = □ or E = S0 x S0

  if R = □ return unsat
  else return sat
```

N.B.: L'algoritmo `Resolvent` restituisce R se C_1, C_2 confliggono, altrimenti restituisce $\{p, \neg p\}$ così da non essere aggiunta in S_0 nel caso in cui C_1 e C_2 non confliggano.

Se questo algoritmo terminasse e restituisse unsat sarebbe corretto perché da un insieme di clausole soddisfacibili la risultante sarà sempre soddisfacibile e quindi non potrei avere la clausola vuota (la regola di risoluzione preserva la soddisfacibilità, quindi, per assurdo, anche la clausola vuota dovrebbe essere soddisfacibile se ricavata da un insieme soddisfacibile).

Questo dà la correttezza, ma la completezza deriva da “se l’insieme è unsat allora avrò la clausola vuota”

Esempio di algoritmo

Definiamo le seguenti proposizione atomiche:

- $C =$ “Carlo è italiano”
- $L =$ “Lucy è francese
- $J =$ ”Jonh è spagnolo”
- R = “Rachel è Tedesca”

E siano le seguenti relazioni:

1. $C \wedge \neg J \rightarrow R$
2. $R \rightarrow (L \vee J)$
3. $\neg L \rightarrow C$
4. $\neg J$

E vogliamo dimostrare che $1,2,3,4 \vDash L$ (l’abbiamo già dimostrato informalmente nel Capitolo 1 quando introducemmo la Conseguenza logica), che equivale a dire $\{1,2,3,4, \neg L\}$ è unsat.

Trasformiamo prima le relazioni in forma clausole:

$$S = \left\{ \underbrace{\{\neg C, J, R\}}_{1}, \underbrace{\{\neg R, L, J\}}_{2}, \underbrace{\{L, C\}}_{3}, \underbrace{\{\neg J\}}_{4}, \underbrace{\{\neg L\}}_{5} \right\}$$

Una possibile deduzione è

$$\dfrac{\dfrac{\dfrac{\dfrac{2 \quad 4}{\{\neg R, L\}} \quad \dfrac{\dfrac{1 \quad 4}{\{\neg C, R\}}\ 3}{\{R, L\}}}{\{L\}} \quad 5}{\square}$$

Ma abbiamo anche una deduzione più corta che è la seguente

$$\dfrac{\dfrac{\dfrac{\dfrac{1 \quad 2}{\{\neg C, J, L\}}\ 3}{\{J, L\}}\ 4}{\{L\}}\ 5}{\square}$$

Guidati dai confliggenti è facile arrivare alla clausola banale se l’insieme è insoddisfacibile, ma se è soddisfacibile allora non posso percorrere questa strada e quindi devo dimostrarne la terminazione.

Dimostrazione della terminazione

Devo garantire che lo spazio di clausole che genero non può essere infinito, perché per come è fatta la regola di riduzione ottengo una unione di due clausole che potrebbe essere più grande delle clausole che uso. Quindi se posso garantire che la riduzione decresce allora avrei la garanzia di terminazione.

Si noti che c’è un limite alla dimensione di qualsiasi clausola che non dipende dalla dimensione delle clausole che sto usando ma è un limite globale. Tale limite è che nessuna clausola può contenere più letterali delle proposizioni atomiche contenute in S:

$$\forall C \in S_0, \qquad |C| \leq |AP(S)|$$

Questa proprietà ci dice che di fatto lo spazio delle clausole è finito. Quindi dimostrare tale proprietà ci garantisce la terminazione.

Supponiamo per assurdo che esista una clausola C dove $|C| > |AP(S)|$, ma ciò significa che:

1) O C contiene due coppe dello stesso letterale (che è un modo di avere più letterali di AP), ma essendo un **insieme** è impossibile avere ripetizioni.
2) Oppure C contiene sia l che $\neg l$ per qualche l ma ciò significherebbe che C sia banale è contraddice l'ipotesi che appartenga a S_0 (l'algoritmo non aggiunge clausole banali a S_0

Questo conclude la dimostrazione. Ma si osservi che non tutte le clausole devono avere la lunghezza massima e quindi contengono addirittura meno letterati; quindi, posso dedurre che (utilizzando il binomiale)

$$|S_0| \leq \sum_{i=1}^{n} \binom{i}{n} = \underbrace{\sum_{i=0}^{n} \binom{i}{n}}_{(1+2)^n} - 1 = 3^n - 1$$

Quindi questo mi permette di dire che in questo conteggio inserisco sia le clausole di dimensione n che minori.

Nel caso peggiore, dunque, avrò costo computazionale $3^n - 1$, ma nella pratica non è tanto meglio di questo valore. Questo conferma che non esiste un algoritmo di insoddisfacibilità meglio di esponenziale.

Osservazione

Come già osservato nell'esempio, C_1 e C_2 se confliggono possono confliggere in tanti modi diversi; quindi, si potrebbe pensare di cancellare tutti gli opposti, ma in realtà cancellarne più di uno contemporaneamente diventa un problema.

Se $\{\neg A, \neg B\}$ e $\{A, B\}$ e cancellassi tutte le coppie confliggenti avrei la clausola vuota ma questa deduzione non è corretta perché avrei ottenuto un insieme insoddisfacibile da un insieme soddisfacibile. Infatti, è ovvio che esiste un modello che soddisfa sia C_1 che C_2 ed è $\alpha(A) = 0$ e $\alpha(B) = 1$.

La correttezza garantisce che se un insieme di clausole è soddisfacibile l'algoritmo dirà che è soddisfacibile.

Teorema di correttezza

Se S è un insieme di clausole e S è soddisfacibile allora $Resolution(S)$ restituiscse SAT

Teorema di completezza

S è un insieme di clausole e S è insoddisfacibile allora $Resolution(S)$ restituiscse $UNSAT$

Dimostrazione

Dimostreremo il teorema di completezza sul numero di proposizioni atomiche contenute in S.

Sia $AP(S)$ l'insieme delle propoposizioni atomiche di S poiché è un insimee finitio (essendo clausole un numero finite e ogni clausola ha un numero finito di letterali).

- $|AP(S)| = 0$ è Il **caso base**; ossia $S = \{\{\}\} \Rightarrow \exists$ deduzione vuota di $\square$ da S ($\square \in S$) che è chiaramente insoddisfacibile poiché contiene $\square$ che fa parte di S ed è $UNSAT$ per definizione.

- Nel **caso induttivo** ho la seguente ipotesi: $\forall S' \ |AP(S')| < n$ se S' è insoddisfacibile allorra □ è deducibile da S'

Sia S un insieme insoddisfacibile (arbitrario) tale che $|AP(S)| = n$ (quindi maggiori dell'insieme S') e sia $p \in AP(S)$ (prendiamo un letterale in S); definiamo due insiemi di clausole:

- $S^p = \{C \setminus \{p\} | C \in S \text{ e } \neg p \notin C\}$ (butto via p dalle clausole che non contengono $\neg p$)
- $S^{\neg p} = \{C \setminus \{\neg p\} | C \in S \text{ e } p \notin C\}$ (butto via $\neg p$ dalle clausole che non contengono p)

N.B.: una clausola in S può benissimo soddisfare entrambi gli insiemi (esempio $\{r, q\}$), ma sicuramente sta almeno in uno dei due.

Questi due insiemi così costruiti hanno la seguente proprietà:

- $S^P \neq \emptyset$ e $S^{\neg P} \neq \emptyset$
    - Entrambi questi due insiemi contengono almeno una clausola per l'ipotesi di insoddisfacibilità, infatti supponendo che $S^P = \emptyset$ significa che non esiste nessuna clausola dove $C \in S$ e $\neg p \notin C$ e quindi tutte le clausole di S contengono $\neg p$, dunque, rendendo p falsa (e quindi $\neg p$ vera) avrei una assegnazione che renderebbe S soddisfacibile (ragionamento analogo per $S^{\neg p}$).

- Sia S^P che $S^{\neg p}$ sono insoddisfacibili
    - Supponiamo per assurdo S^p soddisfacibile, quindi $\exists \alpha: AP(S^p) \to \{0,1\}$ si ha $val^\alpha(S^p) = 1$. Sia $C \in S$ (arbitrario) e costruiamo un assegnamento che le soddisfa tutte, abbiamo che C soddisfa o la condizione per stare in S^P o meno. Quindi due casi:
        - $\neg p \notin C$, quindi $C \setminus \{p\} \in S^P$
        - $\neg p \in C$, quindi $C \setminus \{\neg p\} \in S^{\neg p}$, costruendo $\alpha'(b) = \begin{cases} 0 & \text{se } b = p \\ \alpha(b) & \text{altrimenti'} \end{cases}$ questo assegnamento è coerente con α ed ha le proprietà di soddisfare tutte le clausole di S (qui è la contraddizione)

      Così S è unsat e S^P sat è contraddittorio pocihé posso costruire un α' da S^P che soddisfa anche S (gioco sul valore di verità di p)
    - Ragionamento analogo per $S^{\neg P}$

Quindi in questo modo abbiamo soddisfatto la premessa della clausola induttiva, di conseguenza esiste una deduzione $\begin{matrix} S^P \\ \pi_1 \\ \square \end{matrix}$ e una deduzione $\begin{matrix} S^{\neg P} \\ \pi_2 \\ \square \end{matrix}$

Da queste due deduzioni devo garantire che esiste una deduzione del bottom anche a partire da S.

Ci sono due casi da considerare (si ricorda che le foglie di una deduzione sono le proposizioni di partenza):

- Le foglie di π_1 appartengono a S, questo significa che la deduzione che mi ha portato da S^P a □ mi porterà anche da S a □
- Se esiste una foglia di π_1 e una di π_2 che non appartengono a S allora non sono deduzioni che partono da S e quindi devo costruire una deduzione di S. Prendo tutte le foglie di π_1 e ci aggiungo p, in particolare passo dal ramo:

$$\frac{C_1 \quad C_2}{R} \qquad\qquad \frac{C_1 \cup \{p\} \quad C_2}{R \cup \{p\}}$$

con $R \cup \{p\}$. Quindi praticamente propago p fino in fondo, dove trovo la clausula vuota, che diventer' la clausula che contiene p, ma per lo stesso motive otterrò $\neg p$ da π_2 (entrambe le deduzioni avevano foglie che non sono in S) e quindi basterà applicare la seguente deduzione:
$\frac{p \quad \neg p}{\square}$

Esempi di applicazioni

L'algoritmo di soddisfacibilità genera le possibili soluzioni scegliendole nondeterministicamente. Prima di passare agli esempi diamo una definizione di certificato.

Definizione di certificato

Un certificato è una testimonianza concreta (ad esempio, un'assegnazione di verità alle variabili) che dimostra che una formula è soddisfacibile (cioè vera sotto certe condizioni).

Sia $G = (V, E)$ grafo orientato. G è aciclico?

Codifichiamo una instanza di questo problema, in particolare riduciamo questo problema ad un problema di soddisfacibilità.

Voglio scrivere una formula ϕ_G tale che ϕ_G è $SAT \Leftrightarrow G$ contiene un ciclo.

Codifichiamo innanzitutto l'input:

$$\forall a, b \in V \quad x_{ab} \in AP \text{ per rappresentare } (a, b) \in E$$

Praticamente trasformo gli archi in proposizioni atomiche che saranno vere se l'arco esiste.

$$\phi_G \triangleq \bigwedge_{(a,b)\in E} x_{ab} \wedge \bigwedge_{(a,b)\notin E} \neg x_{ab}$$

A questo punto bisogna costruire un sottoinsieme di archi che contiene un ciclo, l'idea è codificare $S \subseteq E$ come un insieme di archi che formano cicli.

S sarà così caratterizzato come segue. $\forall a, b \in V \;\; y_{ab} \in AP$ per rappresentare $S \subseteq E$

$$\phi_S \triangleq \bigwedge_{a,b\in V} y_{ab} \rightarrow x_{ab}$$

(ogni elemento che metto in S sarà un arco in un grafo, così facendo ho solo archi del grafo)

Per dire invece che S è non vuoto utilizziamo al seguente formula:

$$\phi_{NE} \triangleq \bigwedge_{a,b\in V} y_{ab}$$

A questo punto resta da definire il modo per dire che un insieme di archi contiene un ciclo.

Se nel grafo c'è un ciclo significa che esiste il seguente sottoinsieme:

$$\phi_{CYC} \triangleq \bigwedge_{a,b\in V} \left(y_{ab} \rightarrow \bigvee_{c\in V} y_{bc} \right)$$

Di conseguenza la formula che segue è soddisfacibile se e soltanto se G è ciclico:

$$\phi_G \triangleq \phi_E \wedge \phi_S \wedge \phi_{NE} \wedge \phi_{CYC}$$

Problema NP-completo: Corrabilità di un grafo

Sia un grafo non orientato e un insieme di colori. Voglio assegnare un colore ad ogni vertice; un colore tale che nessuna coppia adiacente di vertici abbia lo stesso colore.

(utile per codificare problemi di assegnamento: assegnare ogni processo uno slot temporale dove due processi con la stessa risorsa non devono essere eseguiti nello stesso slot temporale)

Una soluzione (funzione che associa ad ogni vertice un colore) a questo problema può essere formalizzato come segue:

$$Sol \subseteq \{(v,c) | v \in V \text{ e } c \in C\} \quad \forall v \in V \ \exists! \, c \in C : (v,c) \in Sol$$

Sol è funzione totale da V a C (ovviamente possono esserci più combinazioni di colori che rispettino tale proprietà o colori non utilizzati)

Soluzione 1

Una prima soluzione è associare ad ogni coppia una variabile booleana e utilizzare queste variabili per una sorta di funzione caratteristica: Se è 0 la coppia non è nel certificato (non è una soluzione) se è 1 è una coppa appartenente al certificato

$$\forall (v,c) \in V \times C \quad x_v^c$$

Di queste variabili, che sono $|V| \times |C|$, dobbiamo stare attenti che un assegnamento non mi dia allo stesso vertice più colori (vincolo $\exists! \, c \in C$) inoltre per ogni vertice deve essere vera una condizione (deve avere almeno un colore assegnato). Tale concetto può essere formalizzato come segue:

$$\bigwedge_{v \in V} \left(\bigvee_{c \in C} x_v^c \right)$$

Questo però non esprime l'unicità che viene formalizzata come segue

$$\phi_u \triangleq \bigwedge_{v \in V} \left(\bigvee_{c \in C} \left(x_v^c \wedge \bigwedge_{\substack{c' \in C \\ c' \neq c}} x_v^{c'} \right) \right)$$

ϕ_u esprime che ogni vertice ha uno ed un solo colore.

Abbiamo un altro vincolo, la distindibilità: i vertici di un arco non devono avere lo stesso colore:

$$\phi_D \triangleq \bigwedge_{(u,v) \in E} \left(\bigwedge_{c \in C} (\neg x_u^c \vee \neg x_v^c) \right)$$

ϕ_D esclude che un colore sia assegnato ad entrambi i vertici u, v.

Quindi $\phi \triangleq \phi_U \wedge \phi_D$fa esattamente ciò che ci interessa e se ϕ è soddisfacibile allora i vincoli imposti sono soddisfacibili.

Soddisfacibilità

Garantiamo la soddisfacibilità tramite la dimensione della formula: quindi se $|\phi|$ è polinomiale allora esiste una traduzione polinomiale ed il problema è traducibile in SAT (il costo di traduzione dipenderà dalla lunghezza della formula)

$$|\phi_D| \leq |V|^2 \cdot |C|$$

$$|\phi_U| \leq |V| \cdot |C|^2$$

$$|\phi| = |E| \cdot |C| + |V| \cdot |C|^2 = \Theta(|G| \cdot |C|^2)$$

Abbiamo una traduzione polinomiale, ma possiamo fare di meglio se usiamo una scelta più furba per la codifica.

Soluzione 2

È inutile avere per ogni coppia di vertici una variabile booleana quando sappiamo che sono tutte esclusive. Quindi potremmo forzare l'unicità del colore ad essere sempre vero (quindi associamo ad ogni vertice un colore senza considerare tutti i possibili colori di un vertice).

Numeriamo i colori da 0 a $|C| - 1$ così da poter codificare un colore in $\log_2|C|$ bit, e di conseguenza ogni vertice verrà codificato da $|V| \cdot \log_2|C|$ variabili booleane: $v \to x_v^1 x_v^2 \dots x_v^{\log_2|C|}$ (questa è la codifica di un singolo vertice)

A questo punto non abbiamo più bisogno di ϕ_U ma dobbiamo modificare ϕ_D:

$$\phi'_D \triangleq \bigwedge_{(u,v)\in E} \left(\bigvee_{i=1}^{\log_2|C|} \left(x_u^i \oplus x_v^i\right) \right) \equiv \bigwedge_{(u,v)\in E} \left(\bigvee_{i=1}^{\log_2|C|} \left(\left(x_u^i \wedge \neg x_v^i\right) \vee \left(\neg x_u^i \wedge x_v^i\right) \right) \right)$$

Per ogni arco, almeno uno dei bit corrispondenti del colore deve essere diverso.

$$|\phi| = |\phi'_D| = |E| \cdot \log_2|C|$$

Questa codifica è anche quella asintoticamente ottimale perché se ad esempio avessi due colori, avremmo $\log_2|C| = 1$ e quindia vremo una sola variabile boleana per codificare i colori di un vertice (0 se è un colore e 1 se è dell'altro):

$$|C| = 2 \qquad \phi \triangleq \bigwedge_{(u,v)\in E} (x_u^1 \oplus x_v^2) \equiv \bigwedge_{(u,v)\in E} \left((x_u^1 \wedge \neg x_v^2) \vee (\neg x_u^1 \wedge x_v^2) \right)$$

$$\equiv \bigwedge_{(u,v)\in E} \left(\underbrace{(x_u^1 \vee \neg x_u^1)}_{\text{sempre true}} \wedge (x_u^1 \vee x_v^1) \wedge (\neg x_u^1 \vee \neg x_v^1) \wedge \underbrace{(x_v^1 \vee \neg x_v^1)}_{\text{sempre true}} \right)$$

Quindi è una clausola disgiuntiva con due soli letterali, ossia 2 CNF SAT (risolvibile in tempo polinomiale)

Circuito Hamiltoniano

Il certificato per questo problema è trovare un percorso semplice che contiene tutti i vertici una sola volta (il circuito Hamiltoniano è un esempio di come la logica proposizionale venga usata in informatica). Tale certificato va codificato con un assegnamento, ma poiché ha una lunghezza fissa (tanti vertici quanti ne sono in un grafo) devo creare un assegnamento che mi garantisce l'estraibilità di una sequenza di vertici fatte tutte da coppie di vertici adiacenti.

Di possibili percorse di lunghezza n io devo garantire che non abbia ripetizioni e che contenga tutti i vertici una ed una sola volta. Tale definizione è anche l'insieme delle permutazioni dei vertici (per definizione non contengono una ripetizione), quindi devo codificare una permutazione e poi lasciare scegliere all'algoritmo di SAT una permutazione che sia anche un percorso.

Una permutazione è una funzione $\Pi: V \to \{0,1,\dots,|V|-1\}$

Quindi basta associare ad ogni vertice un numero (allo stesso modo in cui abbiamo associato i colori):

$$\forall v \in V: \; x_v^1 x_v^2 \dots x_v^{\log_2 |V|}$$

Poi ho bisogno di una formula che mi dica che per ogni coppia di vertici almeno uno è diverso (esattamente come prima, così ho una funzione iniettiva).

A questo punto rimane da dire che per ogni coppia di vertici la permutazione è un percorso

$$\bigwedge_{u,v \in V} \left((\pi^u = \pi^v + 1) \rightarrow x_{vu} \right)$$

Ora per terminare bisogna verificare che l'ultimo vertice sia collegato al primo:

$$\bigwedge_{u,v \in V} \left((\pi^u = \pi^v + 1) \rightarrow x_{vu} \right) \wedge \bigwedge_{u,v \in V} \left((\pi^u = 0 \wedge \pi^v = |V|) \rightarrow x_{vu} \right)$$

5. Logica del Primo Ordine

Si differenzia dalla logica proposizionale, poiché le deduzioni non avvengono solo per combinazione di proposizione ma vengono effettuate tramite relazioni.

"Tutti gli uomini sono mortali" e "Socrate è un uomo" si arriva a "Socrate è mortale" tramite la relazione "uomo" appartenente ad entrambe le proposizioni.

La struttura degli oggetti si riflette nell'esistenza di funzioni che permette di comporre o decomporre un "individuo".

Poiché tale deduzione non può essere svolta dalla logica proposizionale bisogna arricchirne il linguaggio, il **linguaggio del primo ordine** è un arricchimento della logica proposizionale.

Definizioni

Struttura

Qui i concetti saranno più ricchi, ad esempio il modello della logica proposizionale viene arricchito e diventa il concetto di struttura, definita come $\big(D, (f_1, f_2, \dots), (R_1, R_2, \dots)\big)$

- D rappresenta il dominio (gli individui)
- $f_1, f_2, \dots$ insieme di funzioni dove una funzione può anche non ricevere input (ad esempio funzione Machete che dato un individuo mi estrapola le sue mani)
- $R_1, R_2, \dots$ insieme di relazioni che sono predicati o relazioni tra oggetti in D (se $D = N$ potremmo avere la relazione di essere primo, $R(x, y, z) \Leftrightarrow x = y + z$) (essere uomo è un altro esempio di questo insieme)

Semantica

Il concetto di semantica è molto più complesso di quello della logica proporzionale.

Un linguaggio del primo ordine (o più precisamente l'alfabeto del linguaggio) è una tupla definita come segue

$$A_{\mathcal{L}} = \big(\{c_i\}_{i\in I}, \{f_i\}_{i\in J}, \{R_i\}_{i\in K}\big)$$

- $\{c_i\}$ è insieme di **costanti** (nomi per gli elementi del dominio)
- $\{f_i\}$ insieme di **simboli funzionali**
- $\{R_i\}$ insieme di **simboli relazionali**

N.B.: tali simboli non hanno alcun significato, quindi ogni simbolo avrà una sua interpretazione che non è detto sia quella comune (ad esempio il simbolo $+$ non è detto che sia addizione); in linea di principio posso usare lo stesso alfabeto per esprimere contenuti completamente diversi.

Arietà

$\forall i \in J$ ad f_i è associata l'arietà (numero positivo che rappresenta il numero di parametri che viene associato alla funzione f_i): si indica con $arity(f)$. Se $arity(f) = n$ allora $f(c_1, \dots, c_n)$ è ben formata.

L'arietà vale anche per le relazioni (Ad Esempio R_i^1 è una relazione unaria, conosciuto come predicato, R_i^2 relazione binaria, etc...). Dire che esiste una relazione tra n oggetti $R(c_1, \dots, c_n)$ significa dare un valore di verità alla relazione di tali costanti (mentre non esiste un valore di verità per f).

Variabile

$x \in X$ sono i diversi nomi di un individuo che puoi dare applicando le funzioni, es. 1 = +(0,1) oppure +(1,0)

Termine (o individuo)

L'insieme $T_{\mathcal{L}}$ di termini t di $\mathcal{L}$ è il più piccolo insieme generato dalle seguenti regole

- $\forall i \in I, t \in T_{\mathcal{L}}$
- $\forall x \in X, x \in T_{\mathcal{L}}$

  } caso base
- se $arity(f) = n$ e $t_1, \dots, t_n \in T_{\mathcal{L}}$, allora $f(t_1, \dots, t_n) \in T_{\mathcal{L}}$ (caso induttivo)

Un esempio di termine sarebbe $+\big(5, \cdot\ (6,7)\big)$ (quindi è intuitivo che i nomi sono infiniti).

Formule

Le formule richiedono la definizione dei simboli non logici significa che non hanno una interpretazione fissata; i simboli logici sono quelli della logica proposizionale ($\wedge, \vee, \neg, \rightarrow, \leftrightarrow$) più i quantificatori esistenziali ($\exists$) e universali ($\forall$) e l'operatore di uguaglianza ($=$).

Si noti che l'operatore $=$ può essere messo sia tra i simboli logici (e quindi l'interpretazione è la stessa per tutti i linguaggi) oppure può essere messo nell'alfabeto e quindi vale lo stesso discorso per i simboli non logici.

Espressione ben formata

$\mathcal{L}$ è il più piccolo insieme di espressioni tali che

- $R(t_1, \dots, t_n) \in \mathcal{L}$ se $arity(R) = n$ e $t_i \in T_{\mathcal{L}}$ per ogni $i \in \{1, \dots, n\}$
  La sequenza di termini deve essere un simbolo del linguaggio e deve avere la stessa arità dei termini che ci metto
  - $R(t_1, \dots, t_n)$ si dice **formula atomica**
- $\phi_1\ op\ \phi_2 \in \mathcal{L}$ se $\phi_1, \phi_2 \in \mathcal{L}$ e $op \in \{\vee, \wedge, \rightarrow, \leftrightarrow\}$
- $\neg\phi \in \mathcal{L}$ se $\phi \in \mathcal{L}$
- $Qx.\ \phi \in \mathcal{L}$ se $x \in X$ e $\phi \in \mathcal{L}$ e $Q \in \{\forall, \exists\}$ (il punto è semplicemente un separatore di quantificatori, in pratica separa l'oggetto del quantificatore con la formula, si può omettere)

Nota: il primo è il caso base mentre tutti gli altri sono casi induttivi

Ora se il simbolo dell'uguale è un simbolo relazionare è già all'interno della descrizione precedente, altrimenti (se appartiene ai simboli logici) devo aggiungere il seguente caso base: $t_1 = t_2 \in \mathcal{L}$ se $= \in SL$ e $t_1, t_2 \in T_{\mathcal{L}}$ (e quindi ogni modello lo interpreta nello stesso modo)

Interpretazione

M è struttura del primo ordine $(D, \mathcal{I})$

- D è un insieme di individui
- $\mathcal{I}$ funzione di interpretazione per $A_{\mathcal{L}}$

Un esempio di struttura del primo ordine è $(\mathbb{N}, \mathcal{I})$ dove l'interpretazione di un simbolo c_{42} ad esempio è 7, ossia $\mathcal{I}(c_{42}) = 7$.

Per una struttura $M = (D, \mathcal{I})$ valgono le seguenti interpretazioni (per simboli, funzioni e relazioni)

- $\mathcal{I}(c) \in D$ per ogni $c \in \{c_1, \dots, c_k\}$

- $\mathcal{I}(f^k) \subseteq \{(d_1, \dots, d_{k+1}) | d_i \in D \ \forall i \in \{1, \dots, k+1\}\}$ se $arity(f) = k$
  Un altro modo per descriverlo è "$\mathcal{I}(f^k)$ è una funzione k-aria su D"
- $\mathcal{I}(R^k) \subseteq \{(d_1, \dots, d_k) | d_i \in D \ \forall i \in \{1, \dots, k\}\}$ se $arity(R) = k$

Ad esempi, sia + l'operatore binario di somma ($arity(+) = 2$), $\mathcal{I}(f^+) = \{\underbrace{(0,0,0)}_{0+0=0}, \underbrace{(0,1,1)}_{0+1=1}, \dots\}$
(si noti che non possono esistere due triple che condividono i primi due elementi ed hanno un terzo elemento diverso).

Semantica

Semantica dei termini

Interpretazione di un termine: $\mathcal{I}(t)$ quanto $t \in T_{\mathcal{L}}$

- $t = c \Rightarrow \mathcal{I}(t) = \mathcal{I}(c)$
- $t = x$? Con x variabile non ho una interpretazione perché $\mathcal{I}$ non dà nessuna informazione sulle variabili (si pensi all'insieme dei numeri $\{1,2,3, \dots\}$, quando dico x posso riferirmi a qualsiasi elemento, quindi $x + y = z$ non ha un valore di verità assoluto)

L'interpretazione delle variabili ha un oggetto separato della struttura che si chiama "assegnamento delle variabili" e si denota con σ. Una funzione $\sigma: X \to D$ rappresenta la corrispondenza tra variabili e valori.

Con $M = (D, \mathcal{I})$ e σ posso dare una interpretazione generale a un termine. L'interpretazione di un termine si denota con formalmente con $[\![t]\!]_\sigma^M$; di cui $t \in T_{\mathcal{L}}$ è l'elemento del dominio D di M (di cui t è il nome. Dunque, sia $[\![t]\!]_\sigma^M \in D$ con $t \in T_{\mathcal{L}}$

- $t = c \Rightarrow [\![t]\!]_\sigma^M = \mathcal{I}(c)$
- $t = x \Rightarrow [\![t]\!]_\sigma^M = \sigma(x)$
- $t = f(t_1, \dots, t_n) \Rightarrow [\![t]\!]_\sigma^M = \mathcal{I}(f)([\![t_1]\!]_\sigma^M \dots [\![t_n]\!]_\sigma^M)$
  (è praticamente l'interpretazione del termine risultante dalla funzione)

Semantica delle formule

Introduciamo la notazione $\sigma[x \leftarrow d](y) = \begin{cases} \sigma(y) \ se \ y \neq x \\ d \quad se \ y = x \end{cases}$
(prendo un assegnamento e costruisco un nuovo assegnamento che è uguale a quello di prima ma sostituisco la variabile x con un elemento di D; in pratica assegno un valore alla variabile)

Sia $M = (D, \mathcal{I})$ e $\sigma: X \to D$. Con ovviamente $D \neq \emptyset$

$M, \sigma \vDash \phi$ con $\phi \in \mathcal{L}$

- $\phi = R(t_1, \dots, t_n) \Rightarrow \big(M, \sigma \vDash \phi \Leftrightarrow ([\![t_1]\!]_\sigma^M, \dots, [\![t_n]\!]_\sigma^M) \in \mathcal{I}(R)\big)$
- $\phi = t_1 = t_2 \Rightarrow (M, \sigma \vDash \phi \Leftrightarrow [\![t_1]\!]_\sigma^M = [\![t_2]\!]_\sigma^M)$
- $\phi = \phi_1 \wedge \phi_2 \Rightarrow (M, \sigma \vDash \phi \Leftrightarrow M, \sigma \vDash \phi_1 \text{ e } M, \sigma \vDash \Phi_2)$
- $\phi = \phi_1 \vee \phi_2 \Rightarrow (M, \sigma \vDash \phi \Leftrightarrow M, \sigma \vDash \phi_1 \text{ o } M, \sigma \vDash \Phi_2)$
- $\phi = \phi_1 \to \phi_2 \Rightarrow (M, \sigma \vDash \phi \Leftrightarrow M, \sigma \nvDash \phi_1 \text{ o } M, \sigma \vDash \Phi_2)$
- Se $\phi = \forall x. \phi_1 \Rightarrow (M, \sigma \vDash \phi \Leftrightarrow M, \sigma[x \leftarrow d] \vDash \phi_1 \text{ per ogni } d \in D)$
- Se $\phi = \exists x. \phi_1 \Rightarrow (M, \sigma \vDash \phi \Leftrightarrow M, \sigma[x \leftarrow d] \vDash \phi_1 \text{ per qualche } d \in D)$

Vedremo in seguito che sono le variabili libere (x non preceduta da quantificatore) a definire la soddisfazione di una formula rispetto sia σ che M.

Esempio

Esempio di livello sintattico: $\left(\underbrace{0,1}_{D}, \underbrace{+,\cdot}_{f}, \underbrace{\leq}_{R}\right)$ Esempio di livello semantico: $\left(\underbrace{\mathbb{N}}_{D}, \underbrace{SUM, PR}_{f}, \underbrace{Less\text{-}Mean}_{R}\right)$

Le interpretazioni mappano gli elementi del livello sintattico a quelli del livello semantico (nel nostro esempio si mapperà 0,1 in $\mathbb{N}$; + in SUM)

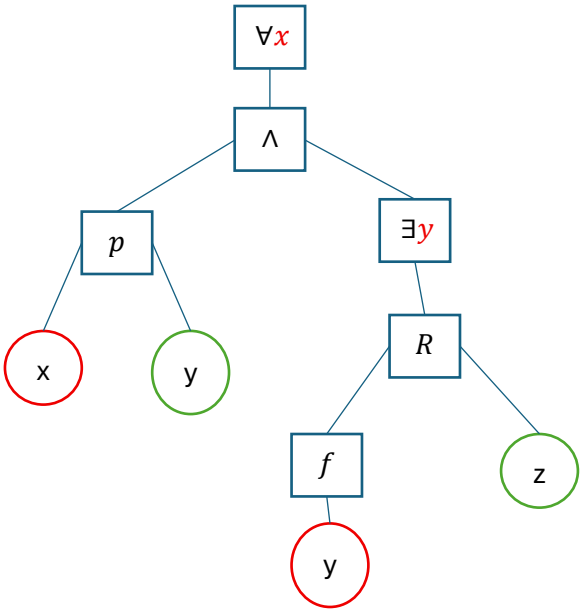
Variabile libera e legata

Una occorrenza di x in ϕ è libero se nell'albero sintattico della formula il percorso dalla radice a x non contiene nodi etichettati con Qx dove $Q \in \{\forall, \exists\}$.

Una occorrenza di x in ϕ è legata se nell'albero sintattico della formula il percorso dalla radice a x contiene nodi etichettati con Qx dove $Q \in \{\forall, \exists\}$.

Esempio

Rappresentiamo $\forall x.\big(p(x,y) \wedge \exists y.R(f(y),z)\big)$ in albero:



Se prendiamo la occorrenza di x più a sinistra e vado verso la radice trovo un quantificatore con il suo nome (che quantifica su di lei) mentre sul "fratello" y non c'è nessun quantificatore che quantifica su di lei. Quindi x è legata e y è libera. Per quanto riguarda invece le altre due foglie abbiamo y legata (quantificata da $\exists y$) mentre z è libera.

Definizione Bound Variable e Free Variable

Un modo di definire le variabili legate e libere è il seguente:

$$BV(\phi) = \{x \in Var(\phi) | x \text{ è legata in } \phi\}$$

$$FV(\phi) = \{x \in Var(\phi) | x \text{ è libera in } \phi\}$$

Nota che non è necessario che $BV(\phi) \cap FV(\phi) = \emptyset$

Formula aperta e chiusa

ϕ è una **formula chiusa** se $FV(\phi) = \emptyset$. Una formula chiusa è dunque una formula che si può valutare indipendentemente dall'assegnamento che si usa per variabili che non conosco (e quindi libere). Infatti, se dico $M, \sigma \vDash \forall x. x \geq 0 \Leftrightarrow M, \sigma' \vDash \forall x. x \geq 0$ poiché è il quantificatore che da il valore a x e non l'assegnamento.

ϕ è una **formula aperta** se $FV(\phi) \neq 0$.

$M, \sigma[x \leftarrow d] \vDash x \geq 0 \; \forall d \in D$ è un esempio di formula aperta (non è chiusa); infatti ho bisogno di un assegnamento per soddisfarla (che è quello di $x \geq 0$; si noti che il quantificatore è su d e la nostra formula invece è rispetto x). Infatti, $M, \sigma[x \leftarrow d] \vDash x \geq 0 \; \forall d \in D \Leftrightarrow M, \sigma'[x \leftarrow d] \vDash x \geq 0 \; \forall d \in D$

Le formule aperte possono essere valide in un M, σ e non valide in M, σ' poiché se due assegnamenti danno valori diversi allora anche se uso lo stesso modello il valore di verità può cambiare.

ϕ è **formula ground** se $Var(\phi) = \emptyset$ (gli assegnamenti qui non servono, infatti non ho variabili, solo termini; dunque, per fare la semantica basta la $\mathcal{I}$).

Soddisfacibilità e validità

1) ϕ è soddisfacibile in una struttura M se esiste un assegnamento σ tale che $M, \sigma \vDash \phi$
   (questo concetto di soddisfacibilità non è presente in logica proposizionale poiché lì esiste solo l'assegnamento e non la struttura)
2) ϕ è vera nella struttura M se e soltanto se per ogni assegnamento σ vale $M, \sigma \vDash \phi$
   (questa corrisponde sostanzialmente al valido nella logica proposizionale)
3) ϕ è soddisfacibilità in FOL (*First Order Logic*) se esiste una struttura M tale che ϕ è soddisfacibile in M (questo concetto di soddisfacibile include implicitamente la 1)
4) ϕ è valida in FOL se ϕ è vera in ogni struttura M
   (valida diventa per ogni modello e assegnamento, quindi abbiamo implicitamente anche la 2)

Nota: tutte le relazioni tra soddisfacibilità e validità descritte nei capitoli precedenti sono ancora soddisfatte.

Conseguenza logica

Per quanto riguarda la conseguenza logica, abbiamo che $\Gamma \vDash \phi$ se e soltanto se per ogni M e σ tale che $M, \sigma \vDash \Gamma$ vale che $M, \sigma \vDash \phi$ (è sempre la solita, solo che quantifico sia su M che su σ)

Insoddisfacibilità

$\Gamma \vDash \phi \Leftrightarrow \Gamma \cup \{\neg\phi\}$ è insoddisfacibile.

Proprietà

(1) Per i termini

Siano σ_1 e σ_2 assegnamenti, e sia $t \in T_{\mathcal{L}}$ un termine tale che $Var(t) = \{x_1, \dots, x_k\}$ e $\sigma_1(x) = \sigma_2(x) \; \forall x \in \{x_1, \dots, x_k\}$ (quindi sulle variabili di t dicono la stessa cosa, ma possono essere diversi per altre variabili) vale che per ogni M

$$[\![t]\!]^{M}_{\sigma_1} = [\![t]\!]^{M}_{\sigma_2}$$

(2) Per le formule

Questa proprietà viene ereditata anche al livello di formule, infatti date le stesse ipotesi.

Se $\phi \in \mathcal{L}$ e $Var(\phi) = \{x_1, \dots, x_n\}$ e $\sigma_1(x) = \sigma_2(x) \; \forall x \in \{x_1, \dots, x_n\}$ allora per ogni M si ha

$$\boldsymbol{M, \sigma_1 \vDash \phi \Leftrightarrow M, \sigma_2 \vDash \phi}$$

N.B.: Si lasciano le dimostrazioni per esercizio (la prima è presente nelle note del prof; mentre la seconda sfrutta la proprietà della prima)

Esempi di formule

- $\forall x. \exists y. (x < y)$ questa, ad esempio, non è vera per il modello $D = \{0,1,2\}$ poiché se scelgo $x = 2$ allora non potrei soddisfare la precedente relazione, ma è vera in tutti i domini numerici conosciuti ($\mathbb{N}, \mathbb{R}$, etc...)
- Un esempio di formula sempre valida è $\forall x. x = x$ oppure $\forall x. \Big(\big(P(x) \to M(x) \wedge P(s)\big) \to M(s)\Big)$ dove P è Persona ed M un Mortale
- $\forall x. \forall y. \exists z. (x + z = y)$ questa è falsa in $\mathbb{N}$ ma vera in $\mathbb{Z}$. Mentre$\forall x. \forall y. \exists z. (x \leq y \to x + z = y)$ è vera anche in $\mathbb{N}$.
- $\forall x. \forall y. \exists z. (x + z = y)$ non è vera in $\mathbb{N}$ e $\mathbb{Z}$ ma vera in $\mathbb{R}$ e $\mathbb{Q}$.

Esempi della logica di primo ordine

1) **"Tutti gli uomini sono mortali"**
   Lo possiamo scrivere come $\forall x. \big(U(x) \to M(x)\big)$ dove $U(\cdot)$ e $M(\cdot)$ sono simboli di relazione rispettivamente della proprietà di essere "Uomo" e di essere "Mortale".
2) $\boldsymbol{R(\cdot,\cdot)}$ **simbolo relazionale binario.**
   Vogliamo esprimere che R è una relazione transitiva, o meglio che
   $$\forall x, y, z. \Big(\big(R(x,y) \wedge R(y,z)\big) \to R(x,z)\Big)$$
   tutte quelle che soddisfano R, descriveranno una relazione transitiva.
   Nota: un insieme di formule viene chiamata **teoria**.
3) **La sequenza S è ordinata**
   Possiamo interpretarla in due modi:
   1. Simbolo funzionale $S(\cdot)$, dove $S: \mathbb{N} \to \mathbb{N}$, usiamo il $<$ come simbolo relazionale binario; quindi possiamo esprimere una sequenza ordinata come segue
      $$\forall x, y \ \big(x < y \to S(x) < S(y)\big)$$
      Questo significa che se abbiamo $\mathbb{N}$, possiamo avere una sequenza infinita, mentre se vogliamo porre un limite, ad esempio 50, possiamo scrivere
      $$\forall x, y \ \Big(\big(x < y \wedge (y < 50 \vee y = 50)\big) \to S(x) < S(y)\Big)$$
   2. Usare un simbolo relazionare binario dove $S(a, b)$ viene interpretato come b è l'elemento in posizione a in S. Dunque dobbiamo rappresentare S come relazione funzionale:
      $$\phi_f \triangleq \forall x. \forall y. \forall z. \Big(\big(S(x,y) \wedge S(x,z)\big) \to y = z\Big)$$
      (ad ogni x corrisponde un solo valore) allora possiamo scrivere
      $$\forall x, y, z, k \ \Big(\big(x < y \wedge S(x,z) \wedge S(y,k)\big) \to z < k\Big) \wedge \phi_f$$
4) **"C'è un solo cane che abbaia"**
   $$\exists x. \left(\big(C(x) \wedge A(x)\big) \wedge \underbrace{\forall y. \Big(\big(C(y) \wedge \neg(y = x)\big) \to \neg A(y)\Big)}_{\text{nessun altro cane abbaia}}\right)$$
   Oppure possiamo scrivere: $\exists x. \forall y. \big(C(x) \wedge A(x) \wedge (C(y) \to x = y)\big)$
5) **"C'è al massimo un cane"**
   Se esiste deve essere unico: $\forall x. \forall y. \Big(\big(C(x) \wedge C(y)\big) \to x = y\Big)$
6) **"Ci sono al più due cani":**

$$\forall x.\forall y.\forall z.\Big(\big(C(x)\wedge C(y)\wedge C(z)\big)\rightarrow(x=y\vee y=z\vee x=z)\Big)$$

7) "Ci sono almeno due cani"

$$\exists x,y\,\big(\neg(x=y)\wedge C(x)\wedge C(y)\big)$$

8) "Ci sono al più n cani"

$$\forall x_1,\dots,x_n,x_{n+1}\left(\bigwedge_{i=1}^{n+1}C(x_i)\rightarrow\bigvee_{i=1}^{n}\bigvee_{j=i+1}^{n+1}(x_i=x_j)\right)$$

9) "Ci sono esattamente due cani"

$$\exists x,y.\big(\neg(x=y)\wedge C(x)\wedge C(y)\big)$$

Esercizi

1. Ogni cane che abbaia è senza proprietario.
   Relazioni: $U(\cdot)$ (proprietario è umano), $C(\cdot)$, $A(\cdot)$ e $p(x,y)$ (x proprietario di y)
2. I cani che hanno un padrone non abbaiano
3. Ogni genitore ha esattamente due figli
4. Ogni vertice (relazione $V(\cdot)$) del grafo ha due archi ($E(\cdot,\cdot)$)

Esiste una formula che dice che il dominio è finito/infinito?

Nella logica di primo ordine non esiste una formula che ci permette di dimostrare la finitezza di un dominio, ossia una formula che è vera solo se il dominio è finito e falsa in dominio infinito.

Supponendo di avere la definizione di $<$ possiamo esprimere che un insieme ha minimo ed un massimo come segue

$$\phi_{INT}\triangleq\exists x,y.\forall z\big(x=z\vee y=z\vee(x<z\wedge z<y)\big)$$

Ma non abbiamo ancora dimostrato se è un insieme finito o infinito; infatti, per questo ci serve il concetto di discretezza.

- Dominio discreto si differenzia dal dominio denso (noi vogliamo una formula vera nei discreti e falsa nei densi). Prendiamo ad esempio $\mathbb{N}$ e $\mathbb{R}$, il primo ha sempre un successore o predecessore mentre il secondo no perché qualsiasi numero superiore prenda ne esiste sempre uno in mezzo.

Il concetto che x ha sempre un successore è definito come segue

$$\phi_{DISC}\triangleq\forall x.\left(\left(\exists y.x<y\right)\rightarrow\left(\exists z.\left(x<z\wedge\forall y.\left(\underbrace{(x<y)\rightarrow(z=y\vee z<y)}_{z\text{ è il successore di }x}\right)\right)\right)\right)$$

Di conseguenza $\phi_{FIN}\triangleq\phi_{INT}\wedge\phi_{DISC}$ (pezzo finito di un ordinamento lineare; ogni elemento se no è l'ultimo ha un successore). Ovviamente $\neg\phi_{FIN}$ è vera in strutture con dominio infinito.

$\neg\phi_{DISC}$ è diverso dal dire che M è denso poiché il concetto di densità non è il duale del concetto di discretezza; infatti, l'assenza del successore significa che c'è un intorno in cui è denso ma non è valido per tutto l'insieme, cosa necessaria per la densità (la negazione dice che non è vero per ogni x, quindi per alcuni potrebbe essere vero e altri falso).

$$\phi_{DENSE}\triangleq\forall x,y.\exists z\big((x<y)\rightarrow(x<z\wedge z<y)\big)$$

(se prendo due elementi esiste sempre uno che sta in mezzo)

Ma ciò non basta; infatti, ϕ_{DENSE} risulta vera anche se prendo un insieme da un elemento per mancanza di elementi (quindi implicazione sempre vera per condizione falsa).

Allora bisogna semplicemente escludere tali insiemi:

$$\phi_{DENSE} \triangleq \exists x, y.\,(x < y) \wedge \forall xy.\,\exists z\big((x < y) \rightarrow (x < z \wedge z < y)\big)$$

$\phi_{DENSE} \not\equiv \neg\phi_{DISC}$ ed è facile trovare un esempio: $\{x \in R | 0 \leq x \leq 1\} \cup \{5\}$

Insieme limitato

Sia M una struttura abbiamo che $\forall M(M \vDash \phi_{FIN} \Leftrightarrow M \text{ è un ordinamento lineare finito})$

Per dire che un modello non è limitato superiormente:

$$\phi_{UNB_D} \triangleq \forall x \exists y.\,(x < y)$$

M è un ordinamento lineare limitato a destra (non ha massimo).

Mentre M è un ordinamento lineare limitato a sinistra

$$\phi_{UNB_S} \triangleq \forall x \exists y.\,(y < x)$$

Quindi per dire che un insieme è illimitato (vale sia sui densi che discreti) basta fare la congiunzione dei due: $\phi_{UNB} \triangleq \phi_{UNB_D} \wedge \phi_{UNB_S}$.

Ordinamenti lineari

Dobbiamo interpretare ora gli ordinamenti lineari (come ad esempio $<$). Per farlo dobbiamo definire le proprietà di un ordinamento lineare:

- Antiriflessiva: $\forall x.\,\neg(x < x)$
- Antisimmetrica: $\forall xy.\,\big((x < y) \rightarrow \neg(y < x)\big)$
- Transitività: $\forall xyz.\,\big((x < y \wedge y < z) \rightarrow x < z\big)$

Le prime tre ci dicono che è una relazione d'ordine parziale. Ci manca la totalità

- $\forall xy.\,(x < y \vee y < z \vee x = y)$

Sia la congiunzione di queste quattro proprietà ϕ_{ORD} abbiamo che $\phi_{ORD} \wedge \phi_{FIN}$ è vera in tutte le strutture con ordinamenti lineari finiti.

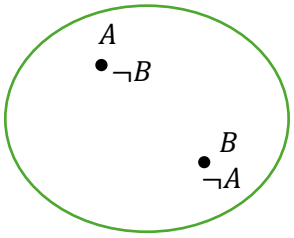
Validità di una formula

Essendo D a volte infinito e quindi $\sigma: X \rightarrow D$ ha a volte codominio infinito verificare la validità è complicato poiché bisogna dimostrarlo per ogni elemento dell'insieme ed a volte è addirittura complicato dare una definizione finita per l'espressione infinita di σ. Di contro il concetto di non validità è più facile (basta trovare un controesempio).

Nota che i quantificatori hanno priorità sulle variabili booleane. Infatti, $\exists x\big(A(x) \rightarrow B(x)\big)$ è diverso da $\exists x.\,A(x) \rightarrow \exists x.\,B(x)$). Adesso forniamo alcuni esempi di formule non valide:

- $\vDash? \big(\exists x.\,A(x) \rightarrow \exists x.\,B(x)\big) \rightarrow \forall x\big(A(x) \rightarrow B(x)\big)$
  - Prendiamo il seguente insieme: $\{\neg A, \neg A, \neg A\}$
  - Questo modello soddisfa la conclusione per insoddisfazione della premessa ma ciò non rappresenta un controesempio (bisogna sempre partire da una premessa vera).

- Voglio quindi un modello che abbia almeno un elemento soddisfatto:



Per semplicità di scrittura d'ora in poi descriverò gli elementi di un insieme mettendo i valori in colonna $\begin{matrix} p \\ q \end{matrix}$. Tali valori si riferiscono allo stesso punto dell'insieme. Di conseguenza l'insieme a sinistra si tradurrà come segue: $\left\{ \begin{matrix} A & \neg A \\ \neg B' & B \end{matrix} \right\}$

- il punto $\begin{matrix} A \\ \neg B \end{matrix}$ soddisfa la premessa ma non la conclusione.

Quindi tale formula <u>non è valida</u> poiché ho trovato un esempio che non la soddisfa.

- $\exists x\big(A(x) \to B(x)\big) \to \big(\forall x\, A(x) \to \forall x\, B(x)\big)$
  - La premessa è debole (vera in molti modelli) mentre la conclusione è forte (il vincolo è difficile da soddisfare, vera in pochi modelli)
  - Rendiamo vera la premessa: $\left\{ \begin{matrix} A & A \\ \neg B' & B \end{matrix} \right\}$;
  - Solo $\begin{matrix} A \\ B \end{matrix}$ soddisfa entrambe ma l'implicazione non vinee soddisfatta per $A, \neg B$

  Dunque, la formula non è soddisfatta
- $\big(\exists x.\, A(x) \to \forall x.\, B(x)\big) \to \forall x\big(A(x) \to B(x)\big)$
  - Il fatto che tutti gli elementi siano $\neg A$ soddisfando la conclusione per falsità della premessa. Quindi il contromodello deve contenere per forza almeno un A.
  - Il problema però è la condizione$\forall x.\, B(x)$ poiché se ho sempre B ovunque (cosa obbligata per soddisfare la premessa) allora la conclusione sarà sempre vera poiché in $A(x) \to B(x)$ abbiamo $B(x)$ sempre vera.

  Quindi in questo caso la formula è valida.
- $\big(\exists x.\, A(x) \to \exists x.\, B(x)\big) \to \exists x\big(A(x) \to B(x)\big)$
  - Valida perché come sopra abbiamo $B(x)$ che basta a soddisfare la conclusione.
  - N.B.: abbiamo il quantificatore esistenziale qui, non universale. Per questo possiamo usare questo ragionamento.
- Esercizio: $\exists x.\, \big(A(x) \to B(x)\big) \to \big(\exists x.\, A(x) \to \exists x.\, B(x)\big)$
  - si noti la dualità con l'implicazione precedente

Calcolo

Il calcolo è una estensione del calcolo proposizionale ma abbiamo bisogno di regole per i quantificatori ed eventualmente per l'uguaglianza.

Concetto di sostituzione

Ho una espressione exp con una variabile x ed un termine arbiatrario t. Voglio sostiturie in exp t ad x.

$\overline{t}_t^x$ è il termine ottenuto sostituendo ogni occorrenza di x in $\overline{t}$ con t. Esempio: $\overline{t} = f\big(g(x,y), h(z,x)\big)$ con x variabile e $t = f(y)$ (voglio sostituire tutte le occorrenze di x con $f(y)$ avrò, di conseguenza, $\overline{t}_{f(y)}^x = f\left(g(f(y),y), h\big(z, f(y)\big)\right)$ mentre $\overline{t}_y^z = f\big(g(x,y), h(y,x)\big)$. Ovviamente $\overline{t}_{f(y)}^k = \overline{t}$.

Sostituzione di un termine

Il concetto di sostituzione può essere definito induttivamente sulla struttura (i primi 2 sono casi base):

$$\overline{t}_t^x = \begin{cases} t & se\ \overline{t} = x \\ \overline{t} & se\ \overline{t} = c\ oppure\ x \notin Var(\overline{t}) \\ f\big(t_{1_t}^x \dots t_{n_t}^x\big) & se\ \overline{t} = f(t_1 \dots t_n) \end{cases}$$

Sostituzione di una formula

Adesso possiamo applicare il concetto di sostituzione all'interno di una formula:

$$\phi_t^x = \begin{cases} R(t_{1t}^x, \dots, t_{nt}^x) & se\ \phi = R(t_1, \dots, t_n) \\ \phi_{1_t}^x \circ \phi_{2_t}^x & se\ \phi = \phi_1 \circ \phi_2\ con \circ \in \{\wedge, \vee, \rightarrow, \dots\} \\ \neg(\psi_t^x) & se\ \phi = \neg\psi \\ \phi & se\ \phi = Qy.\psi\ con\ Q = \{\forall, \exists\}\ e\ x = y \\ Qy.(\psi_t^x) & se\ \phi = Qy.\psi\ con\ Q = \{\forall, \exists\}\ e\ x = y \end{cases}$$

ϕ_t^x è la formula otttenuta rimpiazzando ogni occorrenza libera di x in ϕ

N.B.: La sostituzione per aver senso può essere applicate solo alle occorrenze libere di una formula (non sostituisce qualcosa che deve variare su tutti i nomi: non avrebbe senso)

Esempio: $\phi = \exists y \left(R(x,y) \wedge \forall x.P\big(x, f(z)\big)\right)$ si ha $\phi_{f(z)}^x = \exists y \left(R(f(z),y) \wedge \forall x.P\big(x, f(z)\big)\right)$

Nota che fare ϕ_y^x porterebbe una variabile libera a diventare vincolata (avremmo infatti $\phi_y^x = \exists y \left(R(y,y) \wedge \forall x.P\big(x, f(z)\big)\right)$), e ciò stravolgerebbe il significato della formula (potrebbe diventare insoddisfacibile).

Sostituzione di una assegnazione

$\sigma[x \leftarrow d]$ sostituzione di d a x in σ

Teorema di sostituzione per i termini

Una sostituzione a livello sintattico equivale a quella a livello semantico

$$\left[\!\left[\bar{t}_t^x\right]\!\right]_\sigma^M = \left[\!\left[\bar{t}\right]\!\right]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M$$

Dimostrazione

Caso base. Se $x \notin Var(\bar{t})$ allora $\bar{t}_t^x = \bar{t}$ (t è costante o variabile diversa da x) e quindi

$$\left[\!\left[\bar{t}\right]\!\right]_\sigma^M = \left[\!\left[\bar{t}\right]\!\right]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M$$

Perché t non appartiene a $\bar{t}$ (l'unica per cui differisce non è all'interno di $\bar{t}$).

Se $\bar{t} = x$ allora $\bar{t}_t^x = t$ e quindi abbiamo:

$$\left[\!\left[\bar{t}_t^x\right]\!\right]_\sigma^M = [\![t]\!]_\sigma^M \underbrace{=}_{\substack{\text{per notazione} \\ \text{di } [\![\cdot]\!]_\sigma^M}} [\![x]\!]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M = \left[\!\left[\bar{t}\right]\!\right]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M$$

Caso Induttivo. Se $\bar{t} = f(t_1, \dots, t_n)$ per qualche funzione k-aria f e termini $t_1, .., t_k$, abbiamo per ipotesi induttiva:

$$\left[\!\left[\bar{t}_{it}^x\right]\!\right]_\sigma^M = \left[\!\left[\bar{t}_i\right]\!\right]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M \quad \forall i \in \{1, \dots, k\}$$

Dunque

$$\left[\!\left[\bar{t}_t^x\right]\!\right]_\sigma^M = [\![f(t_1, \dots, t_k)_t^x]\!]_\sigma^M = f^m\left(\left[\!\left[\bar{t}_{1t}^x\right]\!\right]_\sigma^M, \dots, \left[\!\left[\bar{t}_{kt}^x\right]\!\right]_\sigma^M\right) = f^M\left([\![\bar{t}_1]\!]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M, \dots, [\![\bar{t}_k]\!]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M\right)$$
$$= [\![f(t_1, \dots, t_k)]\!]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M = \left[\!\left[\bar{t}\right]\!\right]_\sigma^M$$

Osservazione

In generale non è vero che $M, \sigma \vDash \phi_t^x \Leftrightarrow M, \sigma[x \rightarrow [\![t]\!]_\sigma^M] \vDash \phi$. Infatti, non vale sempre che la sostituzione sintattica sia equivalente alla sostituzione semantica (anche se è quello che vorremmo).

Sostituire un termine in una variabile sarà importante per tutti i problemi di istanziazione: Se una cosa è vera per tutti gli elementi del dominio intuitivamente sarà vera anche per un elemento del dominio.

Concetto di sostituzione per una formula

Data una formula $\phi \in FOL$ e una variabile $x \in Var$ ed un termine $t \in Term$ vogliamo definire ϕ_t^x, ossia la formula ottenuta rimpiazzando ogni occorrenza libera di x in ϕ con il termine t.

Un termine t è <u>libero per x in ϕ</u> se $\exists y$ con $y \in Var(t)$ (x non compare nell'ambito di un quantificatore del tipo $\forall y$).

Questo significa che nessuna variabile del termine non può diventare legata dal quantificatore, e quinci ci evita problemi del seguente tipo: $\phi = \exists y.\, x < y$ che diventa $\phi_y^x = \exists y.\, y < y$

Teorema di sostituzione per le formule

Data $\phi \in FOL, t \in Term$ e $x \in Var$, M una interpretazione e σ un assegnametno.
Se t è libero per x in ϕ

$$\boldsymbol{M, \sigma \vDash \phi_t^x \Leftrightarrow M, \sigma[x \leftarrow [\![t]\!]_\sigma] \vDash \phi}$$

Dimostrazione

Lo dimostriamo per induzione sulla struttura di ϕ:.

Caso base: $\phi = P(t_1 \dots t_n)$. t è libero per x in ϕ quindi $\phi_t^x = P\left(t_{1_t}^x, \dots, t_{n_t}^x\right)$, dobbiamo dimostrare che $M, \sigma \vDash P\left(t_{1_t}^x, \dots, t_{n_t}^x\right) \Leftrightarrow M, \sigma[x \leftarrow [\![t]\!]_\sigma^M] \vDash P(t_1, \dots, t_n)$. Procediamo come segue:

$$\begin{array}{ccc} M, \sigma \vDash P\left(t_{1_t}^x, \dots, t_{n_t}^x\right) & \Leftrightarrow & M, \sigma[x \leftarrow [\![t]\!]_\sigma^M] \vDash P(t_1, \dots, t_n) \\ \Updownarrow & & \Updownarrow \\ \left([\![t_{1_t}^x]\!]_\sigma^M, \dots, [\![t_{n_t}^x]\!]_\sigma^M\right) \in \mathcal{I}(P) & \Leftrightarrow & \left([\![t_1]\!]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M, \dots, [\![t_1]\!]_{\sigma[x \leftarrow [\![t]\!]_\sigma^M]}^M\right) \in \mathcal{I}(P) \end{array}$$

Ora per il lemma di sostituzione dei termini le due interpretazioni sono equivalenti e quindi il caso base è dimostrato (la sostituzione sintattica è equivalente a quella semantica del termine).

Caso induttivo: $\phi = \forall y.\, \psi$

$$\phi_t^x = \begin{cases} \phi \; se \; x = y \\ \forall y, (\psi)_t^x \; se \; x \neq y \end{cases}$$

t è libero per x in ϕ implica (nel caso due) che $y \notin Vat(t)$. Per dimosrtarlo bisogna fare una serie di passaggi semantici:

$$M, \sigma \vDash \forall y.\, (\psi)_t^x \Leftrightarrow M, \sigma[x \leftarrow [\![t]\!]_\sigma^M] \vDash \forall y.\, \psi$$

Per eliminare il per ogni della condizione a sinistra possiamo usare $M, \sigma[y \leftarrow d] \vDash \psi_t^x \forall d \in D$. La condizione a destra è simile:

$$(1) \quad M, (\sigma[x \leftarrow [\![t]\!]_\sigma^M])[y \leftarrow d] \vDash \psi \; \forall d \in D$$

La prima è l'ipotesi induttiva, che ci dice che l'operazione sintattica è equivalente a quella semantica, ovvero $M, \sigma[y \leftarrow d] \vDash \psi_t^x \forall d \in D$ è equivalente a $M, \sigma[y \leftarrow d]\left[x \in [\![t]\!]_{\sigma[y \leftarrow d]}^M\right] \vDash \psi \; \forall d \in D$ (si noti che la differenza sta nell'ordine in cui applico le operazioni semantiche). Notiamo che quanto interpreto t il valore che ha y non ha importanza pociché $y \notin Var(t)$ (non ha nessun impatto) dunque possiamo scrivere:

$$M, \sigma[y \leftarrow d]\left[x \in [\![t]\!]_{\sigma[y \leftarrow d]}^M\right] \vDash \psi \; \forall d \in D \Leftrightarrow M, \sigma[y \leftarrow d][x \in [\![t]\!]_\sigma^M] \vDash \psi \; \forall d \in D$$

Che differisce dalla (1) per l'ordine, ma il valore di x non conta per y e quindi l'ordine non influisce ($x \neq y$). Dunque, $M, \sigma[y \leftarrow d][x \in [\![t]\!]^M_\sigma] \vDash \psi \; \forall d \in D \Leftrightarrow M, \sigma[x \in [\![t]\!]^M_\sigma][y \leftarrow d] \vDash \psi \; \forall d \in D$ che è esattamente la condizione che volevamo (abbiamo dimostrato che l'equivalenza sussiste).

Regole inferenza FOL

Valgono tutte le regole del calcolo proposizionale già descritte:

$$\frac{A \wedge B}{A}\wedge\text{el} \qquad \frac{A \wedge B}{B}\wedge\text{er} \qquad \frac{A \quad B}{A \wedge B}\wedge\text{i} \qquad \frac{A}{A \vee B}\vee\text{il} \qquad \frac{B}{A \vee B}\vee\text{ir} \qquad \frac{\begin{matrix}[B] & [A] \\ \vdots & \vdots \\ C & C\end{matrix} \quad A \vee B}{C}\vee\text{e}$$

$$\frac{\begin{matrix}[A] \\ \vdots \\ B\end{matrix}}{A \to B}\to\text{i} \qquad \frac{A \qquad A \to B}{B}\to\text{e} \qquad \frac{A \quad \neg A}{\bot}\neg\text{e} \qquad \frac{\bot}{A}\bot\text{e} \qquad \frac{\begin{matrix}[\neg A] \\ \vdots \\ \bot\end{matrix}}{A}\text{RAA} \qquad \frac{\begin{matrix}[A] \\ \vdots \\ \bot\end{matrix}}{\neg A}\neg\text{i}$$

A queste vanno aggiunte le regole per i quantificatori e le regole (se necessario) per l'uguaglianza.

Regole per i quantificatori

- ***Forall elimination***
  $\frac{\forall x.\phi}{\phi^x_t}\forall\text{e}$
  a patto che t sia libero per x in ϕ
- ***Forall introduction***
  $\frac{\phi^x_y}{\forall x.\phi}\forall\text{i}$
  a patto che y non occorra in ϕ e non è variabile libera di alcuna assunzione non scaricate da ϕ^x_y (praticamente non può dipendere da assunzioni che hanno y libera)

In sostanza la prima è una sorta di grande and elimination dove selezione uno dei disgiunti. Mentre la seconda è una versione finita di una infinitezza di disgiunzioni

- ***Exists introduction***
  $\frac{\phi^x_t}{\exists x.\phi}\exists\text{i}$
  a patto che t sia libero per x in ϕ
- ***Exists elimination***
  $$\frac{\begin{matrix}[\phi^x_y] \\ \vdots \\ \psi\end{matrix} \qquad \exists x.\phi}{\psi}\exists\text{e}$$
  se y non occorre in ϕ e y non è libera né in ψ né in alcuna assunzione non scaricata di ψ (si noti che è una versione dell'or elimination ed è anche l'unica delle quattro che scarica)

Esempi

1) Il seguente esempio è sbagliato perché da un $\exists x$ siamo passati a $\forall x$. L'$\exists$e non può essere applicata per <u>alcuna assunzione non scaricata da ψ</u>.

$$\frac{\dfrac{[z = 5] \qquad \exists x.(x = 5)}{z = 5}\exists\text{e}}{\forall x.x = 5}\forall\text{i}$$

2) Come sopra, anche nell'esempio di seguito siamo passati da $\exists$ al $\forall$. L'exists elimination bisogna applicarla su variabili libere e <u>presenti</u>; se una variabile non è scaricata non è detto che sia libera.

$$\dfrac{\dfrac{\dfrac{[z=5]\quad[\neg z=5]}{\bot}\neg\text{e}}{\dfrac{\bot}{z=5}\text{RAA}_1}}{\forall x.(x=5)}\forall\text{i}\qquad \exists x.(x=5)\ \exists\text{e}$$

3) $\vdash \underbrace{\forall x.\big(A(y)\to B(x)\big)\to\big(A(y)\to\forall x.B(x)\big)}_{\phi}$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\forall x.(A(y)\to B(x))]_1}{A(y)\to B(z)}\forall\text{e}\qquad [A(y)]_2}{B(z)}\to\text{e}}{\forall x.B(x)}\forall\text{i}}{\dfrac{A(y)\to\forall x.B(x)}{\phi}\to\text{i}_1}\to\text{i}_2$$

4) $\vdash \underbrace{\forall x.\big(A(x)\to B(y)\big)\to\big(\exists x.A(x)\to B(y)\big)}_{\phi}$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\forall x.(A(x)\to B(y))]_1}{A(z)\to B(y)}\forall\text{e}\qquad [A(z)]_3}{B(y)}\to\text{e}\qquad [\exists x.A(x)]_2}{B(y)}\exists\text{e}_3}{\dfrac{\exists x.A(x)\to B(y)}{\phi}\to\text{i}_1}\to\text{i}_2$$

per $\exists$e otteniamo $A(z)$ per ϕ_z^x

5) $\vdash \forall x.\big(A(x)\wedge B(x)\big)\leftrightarrow\forall x.A(x)\wedge\forall x.B(x)$

Primo verso: $\underbrace{\forall x.\big(A(x)\wedge B(x)\big)\to\forall x.A(x)\wedge\forall x.B(x)}_{\theta}$

$$\dfrac{\dfrac{\dfrac{[\forall x.(A(x)\wedge B(x))]_1}{A(y)\wedge B(y)}\forall\text{e}}{A(y)}\wedge\text{el}}{\forall x.A(x)}\forall\text{i}\qquad \dfrac{\dfrac{\dfrac{[\forall x.(A(x)\wedge B(x))]_1}{A(y)\wedge B(y)}\forall\text{e}}{B(y)}\wedge\text{er}}{\forall x.B(x)}\forall\text{i}$$

$$\dfrac{\forall x.A(x)\wedge\forall x.B(x)}{\theta}\wedge\text{i}\ \to\text{i}_1$$

Altro verso: $\underbrace{\forall x.A(x)\wedge\forall x.B(x)}_{\phi}\vdash\underbrace{\forall x.\big(A(x)\wedge B(x)\big)}_{\psi}$

$$\dfrac{\dfrac{\dfrac{\phi}{\forall x.A(x)}\wedge\text{el}}{A(y)}\forall\text{e}\qquad \dfrac{\dfrac{\phi}{\forall x.B(x)}\wedge\text{er}}{B(y)}\forall\text{e}}{A(y)\wedge B(y)}\wedge\text{i}}{\psi}\forall\text{i}$$

6) $\exists x.\big(P(x)\vee Q(x)\big)\vdash\exists x.P(x)\vee\exists x.Q(x)$

$$\dfrac{\exists x.\big(P(x)\vee Q(x)\big)\qquad \dfrac{\dfrac{[P(y)]_1}{\exists x.P(x)}\exists\text{i}}{\exists x.P(x)\vee\exists x.Q(x)}\vee\text{il}\qquad \dfrac{\dfrac{[Q(y)]_2}{\exists x.Q(x)}\exists\text{i}}{\exists x.P(x)\vee\exists x.Q(x)}\vee\text{ir}\qquad [P(y)\vee Q(y)]_3}{\exists x.P(x)\vee\exists x.Q(x)}\vee\text{e}_{12}}{\exists x.P(x)\vee\exists x.Q(x)}\exists\text{e}_3$$

Nota: la regola $\exists\text{e}_3$ è possibile dato che la variabile y non è presente né in $\exists x.\big(P(x)\vee Q(x)\big)$ tanto meno che $\exists x.P(x)\vee\exists x.Q(x)$ poiché la 1 e la 2 sono state precedentemente scaricate.

7) $\exists x.P(x)\vee\exists x.Q(x)\vdash\exists x.\big(P(x)\vee Q(x)\big)$ (l'altro verso della 6)

$$\dfrac{\dfrac{\dfrac{[P(y)]_1}{P(y)\vee Q(y)}\vee\text{il}}{\exists x.\big(P(x)\vee Q(x)\big)}\exists\text{i}\qquad [\exists x.P(x)]_3}{\exists x.\big(P(x)\vee Q(x)\big)}\exists\text{e}_1\qquad \dfrac{\dfrac{\dfrac{[Q(y)]_2}{P(y)\vee Q(y)}\vee\text{ir}}{\exists x.\big(P(x)\vee Q(x)\big)}\exists\text{i}\qquad [\exists x.Q(x)]_4}{\exists x.\big(P(x)\vee Q(x)\big)}\exists\text{e}_2\qquad \exists x.P(x)\vee\exists x.Q(x)$$

$$\exists x.\big(P(x)\vee Q(x)\big)\ \vee\text{e}_{34}$$

8) $\forall x.\, P(x) \vdash \neg\exists x \neg P(x)$ (è l'equivalente della dimostrazione di De Morgan)

$$\dfrac{\dfrac{\dfrac{\dfrac{\forall x.\, P(x)}{P(y)}\,\forall e \qquad [\neg P(y)]_2}{\bot}\,\neg e \qquad [\exists x.\, \neg P(x)]_1}{\bot}\,\exists e_2}{\neg\exists x.\, \neg P(x)}\,\neg i_1$$

9) $\neg\exists x \neg P(x) \vdash \forall x.\, P(x)$ (altro verso)

$$\dfrac{\dfrac{\neg\exists x \neg P(x) \qquad \dfrac{[\neg P(y)]_1}{\exists x \neg P(x)}\,\exists i}{\bot}\,\neg e}{\dfrac{\dfrac{P(y)}{\forall x.\, P(x)}\,\forall i}{}\,RAA_1}$$

Esercizi

1) $\exists x.\, \neg P(x) \vdash \neg\forall x.\, P(x)$

$$\dfrac{\dfrac{\dfrac{[\neg P(y)]_2 \qquad \dfrac{[\forall x.\, P(x)]_1}{P(y)}\,\forall e}{\bot}\,\neg e \qquad [\exists x.\, \neg P(x)]_{hp}}{\bot}\,\exists e_2}{\neg\forall x.\, P(x)}\,\neg i_1$$

2) $\neg\forall x.\, P(x) \vdash \exists x.\, \neg P(x)$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg P(y)]_1}{\exists x \neg P(x)}\,\exists i \qquad [\neg\exists x.\, \neg P(x)]_2}{\bot}\,\neg e}{\dfrac{P(y)}{\forall x.\, P(x)}\,\forall i}\,RAA_1 \qquad [\neg\forall x.\, P(x)]_{hp}}{\bot}\,\neg e}{\exists x.\, \neg P(x)}\,RAA_2$$

3) $\forall x.\, \neg P(x) \vdash \neg\exists x.\, P(x)$

$$\dfrac{\dfrac{\dfrac{[P(y)]_2 \qquad \dfrac{[\forall x.\, \neg P(x)]_{hp}}{\neg P(y)}\,\forall e}{\bot}\,\neg e \qquad [\exists x.\, P(x)]_1}{\bot}\,\exists e_2}{\neg\exists x.\, P(x)}\,\neg i_1$$

4) $\neg\exists x.\, P(x) \vdash \forall x.\, \neg P(x)$

$$\dfrac{\dfrac{\dfrac{[P(y)]_1}{\exists x.\, P(x)}\,\exists i \qquad [\neg\exists x.\, P(x)]_{hp}}{\bot}\,\neg e}{\dfrac{\dfrac{\neg P(y)}{}\,\neg i_1}{\forall x.\, \neg P(x)}\,\forall i}$$

Regole per l'uguaglianza

- Riflessività

  $$\dfrac{}{t = t}\,RFL$$

  per ogni termine t

  Se f ha arietà n

  $$\dfrac{t_1 = s_1 \ldots t_n = s_n}{f(t_1, \ldots, t_n) = f(s_1, \ldots, s_n)}$$

  con t_i,s_i termini

- Sostituzione
  $\dfrac{t_1 = s_1 \dots t_n = s_n}{t^{x_1,\dots,x_n}_{t_1,\dots,t_n} = t^{x_1,\dots,x_n}_{s_1,\dots,s_n}}\,\text{subT}$
  con t_i, s_i termini

  O più formalmente:
  $\dfrac{t_1 = s_1 \dots t_n = s_n}{\phi^{x_1,\dots,x_n}_{t_1,\dots,t_n} \to \phi^{x_1,\dots,x_n}_{s_1,\dots,s_n}}\,\text{subϕ}$

Esempi

- $\vdash \forall x.\, x = x$

$$\dfrac{\dfrac{}{y = y}\,\text{RFL}}{\forall x.\, x = x}$$

- $\vdash \exists x.\, \exists y.\, x = y$

$$\dfrac{\dfrac{\dfrac{}{z = z}\,\text{RFL}}{\exists y.\, z = y}\,\exists\text{i}}{\exists x.\, \exists y.\, z = y}\,\exists\text{i}$$

  Lecito perché nell'exists l'unica condizione che deve essere garantita è che z sia libera per y in ϕ; in questo caso abbiamo $\phi \triangleq z = y$ e $\phi^y_z \triangleq y = y$ (diciamo che la sostituzione è unica ma le ϕ che possono validarla sono molte cosa che non vale con il forall). Si rimarca anche la differenza con il forall che rispetto a exists aggiunte: "e non è variabile libera di alcuna assunzione non scaricate da ϕ^k_y".

- $\boldsymbol{P(t_1), \neg P(t_2) \vdash \neg(t_1 = t_2)}$

$$\dfrac{\dfrac{\dfrac{[t_1 = t_2]_1}{P(t_1) \to P(t_2)}\,\text{subϕ} \qquad [P(t_1)]_{hp}}{P(t_2)}\,\to\text{e} \qquad [\neg P(t_2)]_{hp}}{\dfrac{\bot}{\neg(t_1 = t_2)}\,\neg\text{i}_1}\,\neg\text{e}$$

- $\boldsymbol{x = y \vdash y = x}$

$$\dfrac{\dfrac{[x = y]_{hp} \qquad \dfrac{}{x = x}\,\text{RFL}}{x = x \to y = x}\,\text{subϕ} \qquad \dfrac{}{x = x}\,\text{RFL}}{y = x}\,\to\text{e}$$

- $\boldsymbol{x = y \wedge y = z \vdash x = z}$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{x = x}\,\text{RFL} \qquad \dfrac{[x = y \wedge y = z]_{hp}}{x = y}\,\wedge\text{el}}{x = x \to y = x}\,\text{subϕ} \qquad \dfrac{}{x = x}\,\text{RFL}}{y = x}\,\to\text{e} \qquad \dfrac{[x = y \wedge y = z]_{hp}}{y = z}\,\wedge\text{er}}{y = y \to x = z}\,\text{subϕ} \qquad \dfrac{}{y = y}\,\text{RFL}}{x = z}\,\to\text{e}$$

Esercizi

Suggerimento: si ricordi che $\phi \vee \neg\phi$ è un teorema già dimostrato e quindi c'è una derivazione di questo senza assunzioni. Il che significa che tutte le sue istanze come $\forall x.\, A(x) \vee \neg\forall x.\, A(x)$ sono dimostrabili.

1) $\vdash \neg\exists x.\, \neg(x = x)$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{y = y}\,\text{RFL} \qquad [\neg(y = y)]_2}{\bot} \qquad [\exists x.\, \neg(x = x)]_1}{\bot}\,\exists\text{e}_2}{\neg\exists x.\, \neg(x = x)}\,\neg\text{i}_1$$

2) $\vdash y = z \to \forall x.\, (x = y \to x = z)$

$$\dfrac{\dfrac{\dfrac{\dfrac{}{c=c}\,\mathrm{RFL} \qquad [y=z]_1}{c=y\to c=z}\,\mathrm{sup}\phi}{\forall x.\,(x=y\to x=z)}\,\forall \mathrm{i}}{y=z\to\forall x.\,(x=y\to x=z)}\,{\to}\mathrm{i}_1$$

3) $\vdash \forall x.\,(x=y\to x=z)\to y=z$

$$\dfrac{\dfrac{\dfrac{}{y=y}\,\mathrm{RFL} \qquad \dfrac{[\forall x.\,(x=y\to x=z)]_1}{y=y\to y=z}\,\forall \mathrm{e}}{y=z}\,{\to}\mathrm{e}}{\forall x.\,(x=y\to x=z)\to y=z}\,{\to}\mathrm{i}_1$$

4) $\vdash \neg\big(\forall x.\,P(x)\wedge\neg\neg\exists x.\,\neg P(x)\big)$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\forall x.\,P(x)]_{1\wedge er}}{P(y)}\,\forall \mathrm{e} \qquad [\neg P(y)]_3}{\bot}\,\neg\mathrm{e} \qquad \dfrac{[\neg\neg\exists x.\,\neg P(x)]_{1\wedge el} \qquad [\neg\exists x.\,\neg P(x)]_2}{\bot}\,\neg\mathrm{e}}{\exists x.\,\neg P(x)}\,\mathrm{RAA}_2}{\bot}\,\exists\mathrm{e}_3}{\neg\big(\forall x.\,P(x)\wedge\neg\neg\exists x.\,\neg P(x)\big)}\,\neg\mathrm{i}_1$$

5) $\neg\forall x.\,R(x,x)\vdash\neg\forall x.\,\forall y.\,R(x,y)$

$$\dfrac{\dfrac{[\neg\forall x.\,R(x,x)]_{hp} \qquad \dfrac{\dfrac{\dfrac{[\forall x.\,\forall y.\,R(x,y)]_1}{\forall y.\,R(c,y)}\,\forall \mathrm{e}}{R(c,c)}\,\forall \mathrm{e}}{\forall x.\,R(x,x)}\,\forall \mathrm{i}}{\bot}\,\neg\mathrm{e}}{\neg\forall x.\,\forall y.\,R(x,y)}\,\neg\mathrm{i}_1$$

6) $\vdash \forall x.\,\big(P(x)\to Q(x)\big)\to(\exists x.\,P(x)\to\exists y.\,Q(y))$

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[P(z)]_3 \qquad \dfrac{[\forall x.\,(P(x)\to Q(x))]_1}{P(z)\to Q(z)}\,\forall \mathrm{e}}{Q(z)}\,{\to}\mathrm{e}}{\exists y.\,Q(y)}\,\exists\mathrm{i} \qquad [\exists x.\,P(x)]_2}{\exists y.\,Q(y)}\,\exists\mathrm{e}_3}{\exists x.\,P(x)\to\exists y.\,Q(y)}\,{\to}\mathrm{i}_2}{\forall x.\,\big(P(x)\to Q(x)\big)\to(\exists x.\,P(x)\to\exists y.\,Q(y))}\,{\to}\mathrm{i}_1$$

7) $\vdash \big(\forall x.\,P(x)\to\exists x.\,Q(x)\big)\to\exists x.\,\big(P(x)\to Q(x)\big)$

$$\dfrac{\begin{array}{ccccc}[\forall x.\,P(x)\to\exists x.\,Q(x)]_1,[\forall x.\,P(x)]_2 & & [\neg\forall x.\,P(x)]_3 & & \\ \pi_1 & & \pi_2 & & \pi_3 \\ \exists x.\,\big(P(x)\to Q(x)\big) & & \exists x.\,\big(P(x)\to Q(x)\big) & & \forall x.\,P(x)\vee\neg\forall x.\,P(x)\end{array}}{\exists x.\,\big(P(x)\to Q(x)\big)}\,\vee\mathrm{e}_{23}}{\big(\forall x.\,P(x)\to\exists x.\,Q(x)\big)\to\exists x.\,\big(P(x)\to Q(x)\big)}\,{\to}\mathrm{i}_1$$

- Vedi Teorema 11 (Capitolo 3) per π_3 e fine appunti per π_1 e π_2.

8) $(a=b),(c=b)\vdash\forall x.\,P(a,x)\to\forall xP(c,x)$

$$\dfrac{\dfrac{\dfrac{}{b=b}\,\mathrm{RFL} \qquad \dfrac{\dfrac{[a=b]_{hp} \qquad \dfrac{}{a=a}\,\mathrm{RFL}}{a=a\to b=a}\,\mathrm{sub}\phi \qquad \dfrac{}{a=a}\,\mathrm{RFL}}{b=a}\,{\to}\mathrm{e} \qquad \dfrac{\dfrac{[c=b]_{hp} \qquad \dfrac{}{c=c}\,\mathrm{RFL}}{c=c\to b=c}\,\mathrm{sub}\phi \qquad \dfrac{}{c=c}\,\mathrm{RFL}}{b=c}\,{\to}\mathrm{e}}{(b=b)\to(a=c)}\,\mathrm{sub}\phi}{a=c}\,{\to}\mathrm{e}}{\forall x.\,P(a,x)\to\forall x.\,P(c,x)}\,\mathrm{sub}\phi$$

9) $(a=b),(c=b)\vdash\exists x.\,P(a,x)\to\exists x.\,P(c,x)$

$$\dfrac{\begin{array}{cc}[a=b]_{hp} & [c=b]_{hp} \\ & \pi \\ a=c & \end{array}}{\exists x.\,P(a,x)\to\exists x.\,P(c,x)}\,\mathrm{sub}\phi$$

Dimostrazione della correttezza del calcolo

$$\Gamma \vdash \phi \Rightarrow \Gamma \vDash \phi$$

Si dimostra per induzione sulla lunghezza delle prove (qualsiasi sia la lunghezza della prova la conseguenza logica vale).

Caso base

Lunghezza 0. Quindi $\phi \in \Gamma$ e la conseguenza logica è immediata.

Caso induttivo

Caso induttivo si svolge uno per ogni regola di inferenza conosciuta. Sulle regole di inferenza per le regole proposizionali si rimanda al paragrafo Dimostrazione correttezza del calcolo del capitolo 3.

Induzione sulla lunghezza $h > 0$ (almeno una regola di inferenza è stata applicata) della prova ϕ:

- *forall elimination*

$$\frac{\pi' \begin{cases} \Gamma \\ \vdots \\ \forall x.\phi \end{cases}}{\phi_t^x} \forall \text{e}$$

con t libero per x in ϕ. Quindi la deduzione π' dimostra che $\Gamma \vdash \forall x.\phi \Rightarrow \Gamma \vDash \forall x.\phi$ (ha lunghezza minore h ed è l'ipotesi induttiva)

$\Gamma \vdash \forall x.\phi \Leftrightarrow \Gamma \vDash \forall x.\phi \Leftrightarrow (\forall M\ \forall \sigma\ \ M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \forall x.\phi) \Leftrightarrow M, \sigma[x \leftarrow d] \vDash \phi\ \ \forall d \in D$

Sappiamo che se t è libero (per il vincolo della regola ∀e) allora possiamo applicare il teorema di sostituzione

$$M, \sigma[x \leftarrow d] \vDash \phi\ \ \forall d \in D \xrightarrow{[\![t]\!]_\sigma^M \in D} M, \sigma[x \leftarrow\ [\![t]\!]_\sigma^M] \vDash \phi \Leftrightarrow M, \sigma \vDash \phi_t^x$$

A questo punto:

$$\forall M, \sigma.\, (M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \phi_t^x) \Leftrightarrow \Gamma \vDash \phi_t^x$$

- *forall introduction*

$$\frac{\pi' \begin{cases} \Gamma \\ \vdots \\ \phi_y^x \end{cases}}{\forall x.\phi} \forall \text{i}$$

con y che non occorre in ϕ e non occorre libera in Γ (l'insieme delle assunzioni non scaricare).

$$\left(\Gamma \vdash \phi_y^x \Rightarrow \Gamma \vDash \phi_y^x\right) \Leftrightarrow \left(\forall M, \sigma\ \ M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \phi_y^x\right)$$

Ma poiché y non occorre libera in Γ:

$$(1)\ \forall M, \sigma\ \ M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \phi_y^x$$
$$(2)\ \ \forall \bar{\sigma}\ \ M, \bar{\sigma} \vDash \Gamma \Rightarrow M, \bar{\sigma}[y \leftarrow d] \vDash \Gamma\ \forall d \in D$$

Dalla (1) e dalla (2) implica

$$M, \bar{\sigma}[y \leftarrow d] \vDash \phi_y^x\ \ \forall d \in D$$

Ogni occorrenza di y in ϕ_y^x è un'occorrenza di x in ϕ (non vero in generale ma vero se y non sta dentro ϕ). E quindi $M, \bar{\sigma}[y \leftarrow d] \vDash \phi_y^x\ \ \forall d \in D \Leftrightarrow M, \bar{\sigma}[y \leftarrow d] \vDash \phi\ \forall d \in D$

Abbiamo a questo punto

$$\forall \bar{\sigma}\ \ M, \bar{\sigma} \vDash \Gamma \quad \Rightarrow \quad M, \bar{\sigma}[x \leftarrow d] \vDash \phi\ \forall d \in D$$
$$\Updownarrow$$
$$M, \bar{\sigma} \vDash \forall x.\phi$$

In questo modo abbiamo dimostrato che $\forall M, \sigma.\, (M, \sigma \vDash \Gamma \Rightarrow \mathrm{M}, \sigma \vDash \forall \mathrm{x}.\, \phi) \Leftrightarrow \Gamma \vDash \forall x.\phi$

- *exists elimination*

$$\dfrac{\begin{array}{cc}\Gamma, [\phi_y^x] & \Gamma \\ \vdots & \vdots \\ \psi & \exists x.\phi\end{array}}{\psi}\exists e$$

con y che non occorre in ϕ e non occorre libera in ψ o Γ. Per ipotesi induttiva, $\Gamma \vDash \exists x.\phi$ e $\Gamma, \phi_y^x \vDash \psi$, il che significa $\forall M, \sigma \quad M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \exists x.\phi$ e $\forall M, \sigma \quad M, \sigma \vDash \Gamma \cup \{\phi_y^x\} \Rightarrow M, \sigma \vDash \psi$. Consideriamo M e σ arbitrari, con $M, \sigma \vDash \Gamma$. Siccome $M, \sigma \vDash \exists x.\phi$, sia $d \in M$ il valore per cui x sia il testimone per $\exists x.\phi$, quindi $M, \sigma[x \leftarrow d] \vDash \phi$. Poiché y non è libera in Γ o in ψ abbiamo ($\forall d \in D$):

$$M, \sigma \vDash \Gamma \Leftrightarrow M, \sigma[y \leftarrow d] \vDash \Gamma \text{ e } M, \sigma \vDash \psi \Leftrightarrow M, \sigma[y \leftarrow d] \vDash \psi$$

Siccome y non occorre in ϕ, si ha anche $M, \sigma[x \leftarrow d] \vDash \phi \Leftrightarrow M, \sigma[y \leftarrow d] \vDash \phi_y^x$ (sempre $\forall d \in D$). Possiamo concludere che per puro ragionamento proposizionale, $\forall M, \sigma$, esiste un $d \in M$ tale che

$$M, \sigma[y \leftarrow d] \vDash \Gamma \Rightarrow M, \sigma[y \leftarrow d] \vDash \phi_y^x \text{ e } M, \sigma[y \leftarrow d] \vDash \Gamma \cup \{\phi_y^x\} \Rightarrow M, \sigma[y \leftarrow d] \vDash \psi$$

Di nuovo, siccome y non è libera in Γ o ψ, abbiamo $\forall M, \sigma \quad M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \psi \Rightarrow \Gamma \vDash \psi$

- *exists introduction*

$$\dfrac{\begin{array}{c}\Gamma \\ \vdots \\ \phi_t^x\end{array}}{\exists x.\phi}\exists i$$

con t libera per x in ϕ. Per ipotesi induttiva, $\Gamma \vDash \phi_t^x$ significa che $\forall M, \sigma \quad M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \phi_t^x$. Siccome t è libera per x in ϕ, per il teorema di sostituzione si ha:

$$M, \sigma \vDash \phi_t^x \Leftrightarrow M, \sigma[x \leftarrow [\![t]\!]_\sigma^M] \vDash \phi \overset{d=[\![t]\!]_\sigma^M}{\Longleftrightarrow} M, \sigma[x \leftarrow d] \vDash \phi \overset{\text{per la semantica di } \exists x}{\Longleftrightarrow} M, \sigma \vDash \exists x.\phi$$

Abbiamo concluso che $\forall M, \sigma$ si ha $M, \sigma \vDash \Gamma \Rightarrow M, \sigma \vDash \exists x.\phi$ che è per definizione $\Gamma \vDash \exists x.\phi$.

Dimostrazione della completezza del calcolo

Dato insieme Γ di formule chiuse e ϕ formula chiusa:

$$\begin{array}{ccc} \boldsymbol{\Gamma \vDash \phi} & \Rightarrow & \boldsymbol{\Gamma \vdash \phi} \\ \Updownarrow & & \Updownarrow \\ \Gamma \cup \{\neg\phi\} \text{ è insoddisfacibile} & \Rightarrow & \Gamma \cup \{\neg\phi\} \vdash \bot \text{ (inconsistente)} \end{array}$$

Quindi per contrapposizione abbiamo che $\Gamma \cup \{\neg\phi\}$ consistente deve implicare $\Gamma \cup \{\neg\phi\}$ è soddisfacibile (tenteremo di dimostrare ciò).

In generale è sufficiente dimostrare che ogni insieme Γ di formule chiuse consistente è soddisfacibile.

Partiamo dall'assunzione Γ consistente. Costruiamo il suo insieme massimale, ossia:

$$\Gamma \text{ consistente} \Rightarrow \Gamma^* \text{ massimale e consitstente} \Rightarrow M^* \text{ tale che } M^* \vDash \Gamma^*$$

A differenza di come fatto con la dimostrazione di completezza per la logica proposizionale, in questo caso, il modello ha necessità di un dominio, di simboli relazionali e di simboli di funzioni.

Intuitivamente un dominio di una struttura di primo ordine contiene degli elementi che nella sintassi sono rappresentati da nomi. Infatti, sappiamo che presa qualsiasi struttura $M = (D, \mathcal{I})$ abbiamo che $f(c_1, h(c_2)) = \mathcal{I}(f)(\mathcal{I}(c_1), \mathcal{I}(h)(\mathcal{I}(c_2)))$; quindi, ogni termine ground (costante) è in corrispondenza con un elemento del domino. Dunque, dobbiamo far coincidere i termini ground con il dominio.

Non è detto che $L \in \Gamma$ contenga sufficienti termini ground, ad esempio se $L = \langle\{c\}, \emptyset, \{P\}\rangle$ e $\Gamma = \{\exists x.P(x), \neg P(c)\}$ poiché ho solo c non esiste nessuna struttura che possa soddisfare l'insieme di formule Γ (che sarebbe soddisfacibile se non contenesse un solo elemento).

Quindi abbiamo un problema legato ai quantificatori che potrebbero potenzialmente obbligarmi ad aggiungere un numero infinito di elementi al linguaggio (l'esistenziale mi dice che deve esserci un elemento che lo soddisfa, quindi se non c'è lo devo aggiungere).

Insieme di Henkin

Γ insieme di formule chiuse di L è un insieme di Henkin se per ogni formula di L del tipo $\exists x.\phi$ esiste un termine t tale che $\exists x.\phi \to \phi_t^x \in \Gamma$ (detti assiomi di Henkin)

Questo teorema ci dice che se dovessi avere un esistenziale nel mio insieme dovrei ad un certo punto avere $\exists x.\phi \to \phi_t^x$ che per imply elimination avrei ϕ_t^x (quindi esiste un termine t che soddisfa la formula x). In altre parole, un insieme di Henkin garantisce che esiste un testimone per ogni formula esistenziale.

Costruzione di Γ^*

Quindi la costruzione di Γ^* è decomponibile in due fasi:

1) estendiamo il linguaggio L in un linguaggio L' con nuove costanti (che saranno i nostri testimoni per gli esistenziali) ed estendiamo Γ ad un Γ' che è consistente e di Henkin se Γ è consistente
2) da Γ' costruire Γ^* aggiungendo iterativamente le formule consistenti (questo è quello che abbiamo fatto anche in logica proposizionale)

Fase 1

Definiamo $\{L_i\}_{i\geq 0}$ e $\{\Gamma_i\}_{i\geq 0}$; Sia $L_0 = L$ e $\Gamma_0 = 0$:

$$L_{i+1} = L_i \cup \{c_i | i \in \mathbb{N} \text{ e } c_i \notin L_i\}$$

$$\Gamma_{i+1} = \Gamma_i \cup \left\{ \exists x.\phi_j \to \phi^*_{j_{c_j}} \middle| \exists x.\phi_j \in L_i \text{ e } \begin{matrix} \phi_j \text{è la } j\text{-esima formula nella enumerazione delle} \\ \text{formule di } L_i \text{e } \exists x.\phi_j \text{ ha solo } x \text{ come variabile libera} \end{matrix} \right\}$$

Si noti che il motivo di dover fare una sequenza è per il problema dell'annidamento degli esistenziali; infatti, ogni fase si occupa dell'esistenziale più esterno.

- Esempio
  $L_0 = (\emptyset, \emptyset, \{P, R\})$ con P^2 e R^1 e prendiamo la seguente formula: $\exists x.\exists y.\big(P(x,y) \wedge R(y)\big) \in L_0$. Tale formula la possiamo vedere come $\exists x.\phi(x)$ dove $\phi(x) = \exists y.\big(P(x,y) \wedge R(y)\big)$ (sarà il nostro j)
  Con Γ' insieme consistente e di Henkin poiché composto partendo da Γ consistente.
  A questo punto il dominio del modello è esattamente i termini ground.
  Preso L_1 inseriamo in Γ_1 l'assioma di Henkin $\underbrace{\exists x.\exists y.\big(P(x,y) \wedge R(y)\big)}_{\exists x.\phi(x)} \to \underbrace{\exists y.\Big(P\big(c_j,y\big) \wedge R(y)\Big)}_{\phi^x_{c_j}}$

  $$\exists y, \Big(P\big(c_j,y\big) \wedge R(y)\Big) \in L_1 \setminus L_0$$
  $$L_2 = L_1 \cup \{\bar{c}_i | i \in \mathbb{N} \text{ e } \bar{c}_i \notin L_1\}$$

  Mentre Γ_2 inserisco l'assioma di Henkin $\exists y.\Big(P\big(c_j,y\big) \wedge R(y)\Big) \to P\big(c_j,c_k\big) \wedge R(c_k)$.
  Dunque, T_2 conterrà quindi sia quella in Γ_1 che quella appena definita.

Al termine della fase 1 abbiamo

$$L' = \bigcup_{i\geq 0} L_i \qquad \Gamma' = \bigcup_{i\geq 0} \Gamma_i$$

con $L \subseteq L'$ e $\Gamma \subseteq \Gamma'$

Fase 2

Ogni Γ' consistente e di Henkin può essere esteso ad un Γ^* massimale consistente e di Henkin (teorema di Lindenbaum per la logica del primo ordine)

In questa fase non aggiungeremo più costanti ed è praticamente la stessa per la logica proposizionale.

Data una enumerazione delle formule di L' costruiamo una sequenza $\{\Gamma'_i\}_{i\geq 0}$ come segue:

$$\Gamma'_0 = \Gamma'$$

$$\Gamma'_{i+1} = \begin{cases} \Gamma'_i \cup \{\phi_i\} & \text{se } \Gamma'_i \cup \{\phi_i\} \text{ è consistente} \\ \Gamma'_i \cup \{\neg\phi_i\} & \text{altrimenti} \end{cases}$$

Dunque,

$$\Gamma^* = \bigcup_{i\geq 0} \Gamma'_i$$

E per gli stessi motivi di quelli visti per la completezza nella logica proposizionale (capitolo "Calcolo") Γ^* è consistente, è di Henkin (perché ne ha tutti gli assiomi) ed è massimale (per costruzione).

Al termine della seconda fase è già dimostrabile che per ogni formula chiusa ϕ di L' si ha
$\phi \in \Gamma^* \Leftrightarrow \Gamma^* \vdash \phi$ *(che significa che* Γ^* *è saturo)*

- Verso $\Rightarrow$:
  supponiamo che $\phi \notin \Gamma^*$, ciò significa che per massimalità $\neg\phi \in \Gamma^*$ e assumiamo per ipotesi di contraddizione che $\Gamma^* \vdash \phi$, ossia $\Gamma^* \vdash \phi \Leftrightarrow \Gamma^* \cup \{\neg\phi\}$ è inconsistente. Ma questo, con l'ipotesi $\phi \notin \Gamma^*$, significa che Γ' è inconsistente. Quindi, abbiamo dimostrato che
  $$\phi \notin \Gamma^* \Rightarrow \Gamma^* \nvdash \phi \equiv \Gamma^* \vdash \phi \Rightarrow \phi \in \Gamma^*$$
- Altro verso:
  $$\phi \in \Gamma^* \Rightarrow \neg\phi \notin \Gamma^* \Rightarrow \Gamma^* \cup \{\neg\phi\} \text{ è inconsistente} \Leftrightarrow \Gamma^* \cup \{\neg\phi\} \vdash \bot$$
  e quindi per reduzium ad absurdum otteniamo $\Gamma^* \vdash \phi$

Costruzione del modello

Da Γ^* costruiamo un modello M^* tale che $M^* \vDash \Gamma^*$ nel seguente modo:

$M^* = (D^*, \mathcal{I}^*)$ dove $D^* = \{t | t \text{ è termine ground di } L'\}$

Siccome nel dominio ci sono già i termini ground, possiamo interpretare ogni costante in sé stessa e quindi $\mathcal{I}(c) = c \; \forall c$ costante di L'

Le funzioni sono le stesse perché L' aggiunge solo costanti e dunque le funzioni di M sono le stesse di L e quindi $\mathcal{I}^*(f) = f^{M^*} : (g^*)^k \to D^*$ tale che $f^{M^*}(t_1, \dots, t_k) = f(t_1, \dots, t_k)$

Infine, per i simboli relazionali abbiamo $\mathcal{I}^*(R) = R^{M^*} = \{(t_1, \dots, t_k) | R(t_1, \dots, t_k) \in \Gamma^*\}$

Dimostrazione che $M^* \vDash \Gamma^*$ *(conclude il tutto)*

Dimostrazione per induzione sulla struttura di ϕ: per ogni formula $\phi \in L'$ abbiamo $\phi \in \Gamma^* \Leftrightarrow M^* \vDash \phi$

Caso base:

- ϕ è atomica e quindi $\phi = R(t_1, \dots, t_n)$ che per ipotesi è chiusa e quindi i termini t_i sono tutti ground; dunque, la formula atomica è già in Γ^* (e per come abbiamo definito il modello è ovvio per costruzione di M^*)

Caso induttivo:

- $\phi = \neg\alpha$ per qualche formula chiusa α; segue per ipotesi induttiva che $\alpha \in \Gamma^* \Leftrightarrow M^* \vDash \alpha$.
  Ma se tale equivalenza vale allora vale anche che $\alpha \notin \Gamma^* \Leftrightarrow M^* \nvDash \alpha \Leftrightarrow \mathrm{M}^* \vDash \neg\alpha$ e poiché M^* è massimale $\alpha \notin \Gamma^* \Leftrightarrow \neg\alpha \in \Gamma^*$ e $\phi \in \Gamma^* \Leftrightarrow \alpha \notin \Gamma^* \Leftrightarrow M^* \nvDash \alpha \Leftrightarrow \mathrm{M}^* \vDash \neg\alpha \Leftrightarrow M^* \vDash \Gamma^*$
- $\phi = \alpha \vee \beta$ per qualche formula chiusa α e β. Abbiamo che $\phi \in \Gamma^*$ sse $\alpha \in \Gamma^*$ o $\beta \in \Gamma^*$. Per ipotesi induttiva, $\alpha \in \Gamma^*$ o $\beta \in \Gamma^* \Leftrightarrow M^* \vDash \alpha$ o $M^* \vDash \beta \Leftrightarrow M^* \vDash \alpha \vee \beta$ che è vero sse $M^* \vDash \phi$
- Supponiamo $\phi = \forall x.\psi$ con ψ formula aperta con x unica variabile libera.
  Dimostriamo entrambi i versi di $\forall x.\psi \in \Gamma^* \Leftrightarrow M^* \vDash \forall x.\psi$.
  - Assumiamo $\forall x.\psi \in \Gamma^*$ allora possiamo dimostrare che per ogni $t \in D^*$ si ha $\psi_t^x \in \Gamma^*$. Per assurdo supponiamo che esista una $\bar{t} \in D^*$ tale che $\neg\psi_{\bar{t}}^x$ ma se ciò fosse vero $\forall x.\psi \in \Gamma^*$ avrò
    $$\frac{\forall x.\psi}{\psi_{\bar{t}}^x} \forall\text{e con } \bar{t} \text{ libero per } x \text{ in } \psi$$
    ma siccome $\bar{t}$ è ground è sempre libero e quindi per *forall eliminiation* dedurrei $\frac{\forall x.\psi}{\psi_{\bar{t}}^x}$ con $\psi_{\bar{t}}^x \in \Gamma^*$ rendendolo inconsistente; dunque, segue $\neg\psi_{\bar{t}}^x \notin \Gamma^*$ per ogni $\bar{t} \in D^*$
    Ciò per massimalità di Γ^* significa che $\psi_t^x \in \Gamma^*$ per ogni $\bar{t} \in D^*$; inoltre, ψ_t^x è chiusa (l'unica varibaile libera era x ma l'ho sostituita con il termine ground $\bar{t}$) e ciò significa che $M^* \vDash \psi_t^x$ per ogni $t \in D^*$ che equivale a $M^*, \sigma \vDash \psi_t^x$ per ogni $t \in D^*$ e per ogni σ.
    Ma per il teorema di sostituzione ho
    $$M^*, \sigma \vDash \psi_t^x \Leftrightarrow M^*, \sigma[x \leftarrow t] \vDash \psi\ \forall t \in D \Leftrightarrow M^*, \sigma \vDash \forall x.\psi \Leftrightarrow M^* \vDash \forall x.\psi$$
  - Assumiamo $M^* \vDash \forall x.\psi$; questo equivale a $M^* \vDash \psi_t^x$ per tutti gli elementi di t nel dominio di M^*. Per ipotesi induttiva, allora, $\psi_t^x \in \Gamma^*$ per tutti i termini t di L'. Supponiamo per assurdo $\forall x.\psi \notin \Gamma^*$ allora per la massimalità di Γ^*:
    $$\neg\forall x.\psi \in \Gamma^* \Leftrightarrow \exists x.\neg\psi \in \Gamma^*$$
    Poiché Γ^* è un insieme di Henkin, esiste una costante $\bar{c}$ per cui vale l'assioma di Henkin:
    $$\exists x.\neg\psi \rightarrow \neg\psi_{\bar{c}}^x \in \Gamma^*$$
    Ma essendo $\bar{c}$ un termine ground di L'abbiamo ottenuto una contraddizione con l'ipotesi che $\psi_t^x \in \Gamma^*$ per tutti i termini t di L'. Per cui,$\forall x.\psi \in \Gamma^*$ come desiderato.

Problema dell'espressività

Abbiamo dimostrato il problema della completezza del calcolo di FOL: $\Gamma \vDash \phi \Rightarrow \Gamma \vdash \phi$

Questa espressione implica una proprietà già osservata quando abbiamo parlato della completezza della logica proposizionale, ovvero il Teorema di Compattezza che generalizzato enuncia:

Per ogni insieme Γ *di formule di FOL e formule* ϕ *vale che*
$\Gamma \vDash \phi$ *sse* $\exists\Gamma_0 \subseteq \Gamma$ *con* Γ_0 *finito tale che* $\Gamma_0 \vDash \phi$

Questo teorema è una diretta conseguenza del teorema di completezza; infatti, il teorema di completezza dice che ϕ è conseguenza logica di Γ se esiste una sequenza finita di formule che da Γ derivano ϕ.

$$\Gamma \vDash \phi \xRightarrow{\text{Teorema Completezza}} \Gamma \vdash \phi \Leftrightarrow \exists\pi \text{ deduzione di } \phi \text{ da } \Gamma$$

Ma π è albero finito $\Gamma_0 = \{\psi | \psi \text{ è formula di } \pi\}$; dunque:

$$\Gamma_0 \vdash \phi \text{ e } \Gamma_0 \text{ è finita} \xLeftrightarrow{Teorema\ Correttezza} \Gamma_0 \vdash \phi \Rightarrow \Gamma \vDash \phi$$

Il teorema di compattezza. Per come è definito il concetto di conseguenza logica esprime in pratica:

$\Gamma \cup \{\neg\phi\}$ è insoddisfacibile $\Leftrightarrow \exists \Gamma_0 \subseteq \Gamma$ finito tale che $\Gamma_0 \cup \{\neg\phi\}$ è insoddisfacibile

Che in termini equivalenti posso scrivere

- Γ è insoddisfacibile $\Leftrightarrow \exists \Gamma_0 \subseteq \Gamma$ finito Γ_0 è insoddisfacibile
- Γ è soddisfacibile $\Leftrightarrow \forall \Gamma_0 \subseteq \Gamma_0$ è soddisfacibile
    - $\Rightarrow$ dice: se riesco a dimostrare che un insieme infinito è soddisfacibile, tale soddisfazione varrà per ogni suo sottoinsieme

Dimostrare proprietà di espressività

Una proprietà P è un insieme di modelli esprimibile in FOL se esiste una formula ϕ_P tale che per ogni struttura M, $M \in P \Leftrightarrow M \vDash \phi_P$

Il concetto di espressività dichiara che se una formula esprime una proprietà (un concetto) p allora p è vera in tutti i modelli che hanno tale proprietà e falsa in tutti i modelli che non possiedono tale proprietà.

Se ad esempio prendo $\phi \triangleq \exists x. \exists y. x \neq y$ tale formula è vera in tutte e sole le strututre che hanno almeno due elementi (discrimina le strutture con un solo elemento); quindi, $P = \{M \mid |M| > 1\}$

Così come $\phi' \triangleq \exists x, y. \big(x \neq y \; e \; \forall z(z = x \vee z = y)\big)$ (discrimina tutte le strutture che hanno esattamente due elementi). $P = \{M \mid |M| = 2\}$.

Se esiste una formula che esprime una proprietà, tale proprietà è assiomatizzata da quella formula.

Proprietà non esprimibili

La proprietà di finitezza del dominio

Avere un dominio finito è una proprietà che raccoglie tutti i modelli con dominio finito. Tale proprietà non è esprimibile in FOL.

Sto cercando in pratica una formula vera in tutti i modelli che hanno un dominio finito senza esprimerne un estremo superiore (senza esplicitamente dire ad esempio “modelli con al massimo 7 elementi”). Possiamo identificare una porzione di domini finiti (come abbiamo fatto per i modelli con due elementi), ma farlo per tutti è impossibile.

Supponiamo Γ infinito tale che ogni sottoinsieme finito è soddifscibile e ipotizziamo per assurdo che esista ϕ_F che è soddisfatta da tutte e sole le strutture con dominio finito.

Costruiamo quindi un Γ con questa proprietà: $\Gamma = \{\phi_n | n \geq 1\}$

ϕ_n vuole esprimere la proprietà che esistono almeno n elementi:

$$\phi_i \triangleq \exists x_1 x_2 \dots x_n. \bigwedge_{\substack{1 \leq i,j \leq n \\ i \neq j}} \neg(x_i = x_j)$$

A questo punto prendiamo l’insieme $\Gamma \cup \{\phi_F\}$ e analizziamone le proprietà:

- Per $\Gamma \cup \{\phi_F\}$ soddisfacibile dovrebbe soddisfare tutte le formule che sono in Γ
- Se $M \vDash \Gamma \cup \{\phi_F\}$ allora $|M| < \infty$ e $\forall n \geq 1 \;\; |M| \geq n$
- Ma $\forall n \geq 1 \; |M| \geq n \Rightarrow |M| = \infty$ se non ho un limite superiore.
- Dunque, se non esiste un tale insieme abbiamo $M \vDash \Gamma \cup \{\phi_F\}$ insoddisfacibile.

Sia $\Gamma_0 \subseteq \Gamma$ finito. Allora Γ_0 è sat poiché Γ_0 è finito e quindi $\exists k$ tale che $\forall z > k. \phi_z \notin \Gamma_0$ (quindi k esprime l'estremo supreiorie). Se $\phi_F \in \Gamma_0$ abbiamo che M è dominio finito con almeno k elementi.

Tale discorso vale per qualsiasi modello finito (c'è sempre un massimo); ma questo evince un problema. Infatti, $\Gamma \cup \{\phi_F\}$ è insoddisfacibile ma non esiste nessun sottoinsieme insoddisfacibile di Γ; che per il teorema di completezza implicherebbe $\Gamma \cup \{\phi_F\}$ sat, il che è assurdo.

Per questo la proprietà non è esprimibile. Lo stesso vale per la proprietà di infinitezza.

Raggiungibilità di un grafo

Analogamente, una altra proprietà non esprimibile in FOL è la raggiungibilità dii un grafo.

Sia $G = (V, E)$ con $M_G = (D, \{R^2\})$ (con $V = D$ e E espresso in $\{R^2\}$).

Nel linguaggio $\mathcal{L}_G = (\{s, t\}, \{R^2\})$ posso solo dire che non esiste una formula di FOL che esprime la proprietà di essere connesso per un grafo (o la struttura associata M_G).

L'approccio è simile all'esempio precedente. Supponiamo ϕ_{CON} che è vera in tutte e sole le strutture in cui s e t sono connesse. $\Gamma = \{\phi_n(s,t) | n \geq 1\} \cup \underbrace{\{\forall x. \neg R(x,), \forall x, y. \big(R(x,y) \rightarrow R(y,x)\big)\}}_{\text{per essere più generico possibile}}$

$\phi_n(s,t)$ dice che non esiste un percorso da s a t di lunghezza n (può esserci un percorso, ma non di lunghezza n).

$$\phi_n(s,t) \triangleq \neg \exists x_1 \dots x_n. \left(x_1 = s \wedge x_n = t \wedge \bigwedge_{i=1}^{n-1} R(x_i, x_{i+1}) \right)$$

(si noti che tale concetto è la versione semplice di quello di percorso definito in Esempi di applicazioni nel capitolo 4).

Prendiamo $\Gamma \cup \{\phi_{CON}\}$, se una struttura soddisfa Γ allora non può esistere un modello che soddisfi $\Gamma \cup \{\phi_{CON}\}$ (qualsiasi lungehzza scelga non c'è un percorso tra s e t).

Quindi se un modello soddisfa Γ non c'è un percorso tra s e t ma se prendiamo un arbitrario sottoinsieme finito, che avrà una massima formula parametrizzata con un massimo k tale che il sottoinsieme $\Gamma_0 \cup \{\phi_{CON}\}$ venga soddisfatto.

Problema della decisione

Prendiamo una macchina di Turing $TM = (Q, \Sigma, \delta, 0, n)$. con $\delta: Q \times \Sigma \rightarrow Q \times \Sigma \times \{L, R\}$
Tale macchina termina se partendo da q_0 alliro vallo stato finale n.

La computazione è $\pi : \mathbb{N} \rightarrow Q \times \mathbb{N} \times tape$ con $\pi(i) = (q \in Q, p \in \mathbb{N}, tape)$ e $tape: \mathbb{N} \rightarrow \Sigma$ funzione.

Scegliamo un linguaggio $\mathcal{L} = \left(\{0\}, \{p', s'\}, \left\{ q_i^2 \middle| i \in \{0, \dots, n\} \right\} \cup \left\{ a_i^2 \middle| i \in \{0, \dots, n\} \right\} \right)$

L'idea è che p deve catturare la funzione predecessore e s la funzione successore.

Tali predicati sono definiti da $q_i(x,y)$ che vuole valutare il fatto che la TM si trova nello stato i al passo k ed è posizionata nella cella y. (x è il passo di computazione e y la posizione del nastro).

$a_i(x,y)$ sarà vero se al passo x nella cella y è presente il simbolo i.

Esprimiamo le seguenti proprietà

1) Coerenza delle posizioni sul nastro

$$\phi_1 \triangleq \forall x \left(p\big(s(k)\big) = x \wedge \big(x > 0 \rightarrow s\big(p(x)\big) = x\big) \right)$$

2) Coerenza di una computazione (in ogni passo, ogni cella può contenere solo un simbolo):

$$\phi_2 \triangleq \forall x, y. \bigwedge_{\substack{i \neq j \\ i,j \in \Sigma}} \neg \left(a_i(x,y) \wedge a_j(k,y) \right)$$

3) In ogni istante la TM sta in un unico stato:

$$\phi_3 \triangleq \forall x, y, z. \bigwedge_{i,j \in Q} \neg \left(q_i(x,y) \wedge q_j(x,z) \right)$$

4) In ogni istante la testina sta in una sola posizione:

$$\phi_4 \triangleq \forall x, y, z. \left(y \neq z \rightarrow \bigwedge_{i \in Q} \neg \left(q_i(x,y) \wedge q_i(x,z) \right) \right)$$

5) Se un simbolo in posizione y cambia al passo x allora uno stato era attivo in quella posizione in quel momento

$$\phi_5 \triangleq \forall x, y. \bigwedge_{i \in \Sigma} \left(\left(a_i(x,y) \wedge \neg a_i(s(x),y) \right) \rightarrow \bigvee_{j \in Q} q_j(x,y) \right)$$

6) Descrive una transizione:

$$\phi_6 \triangleq \forall x, y. \bigwedge_{\substack{i \in Q \\ j \in \Sigma}} \phi_{ij}$$

7) Descrizione di tutti i possibili passi in δ

$$\phi_{ij} \triangleq \left(q_i(x,y) \wedge a_j(x,y) \right) \rightarrow \left(q_k \left(s(x), \begin{matrix} p(y) \\ s(y) \end{matrix} \right) \wedge a_z(s(x),y) \right)$$

   - Guardando la transizione $\delta(i,j) = \left(k, z, \begin{matrix} L \\ R \end{matrix} \right)$ (ho tutte e sole le possibili mosse della MT)

Sia ϕ_z lo stato iniziale: $\phi_z \triangleq q_0(0,0) \wedge \forall y. a_0(0,y)$ (il nastro è vuoto)

A questo punto posso definire la computazione terminante:

$$\phi_\pi \triangleq \bigwedge_{r=1}^{z} \phi_r \rightarrow \exists x, y. q_n(x,y)$$

Se TM non termina esiste un modello che falsifica ϕ_π (se la macchina non termina ϕ_π non può essere vera). Il modello $M = \langle \mathbb{N}, \mathcal{I} \rangle$ con

- $\mathcal{I}(0) = 0$,
- $\mathcal{I}(s) =$ funzione successore,
- $\mathcal{I}(p) =$ funzione predecessore,
- $\mathcal{I}(q_i) = \{(t,p) | (\pi)_t = (i,p,tape)\}$ ($t =$ tempo, $p =$ posizione)
- $\mathcal{I}(a_i) = \{(t,p) | (\pi)_t = (j,p',tape_t)\ e\ tape(p) = i\}$

$M_\pi \vDash \bigwedge_{i=1}^{\pi} \phi_i$ per costruzioen am $M_\pi \nvDash \exists x, y. q_n(x,y)$

Regole di inferenza

$$\frac{A \land B}{A}\land\text{el} \qquad \frac{A \land B}{B}\land\text{er} \qquad \frac{A \quad B}{A \land B}\land\text{i} \qquad \frac{A}{A \lor B}\lor\text{il} \qquad \frac{B}{A \lor B}\lor\text{ir} \qquad \frac{\begin{matrix}[B] & [A] & \\ \vdots & \vdots & \\ C & C & A \lor B\end{matrix}}{C}\lor\text{e}$$

$$\frac{\begin{matrix}[A]\\ \vdots \\ B\end{matrix}}{A \to B}\to\text{i} \qquad \frac{A \qquad A \to B}{B}\to\text{e} \qquad \frac{A \quad \neg A}{\bot}\neg\text{e} \qquad \frac{\bot}{A}\bot\text{e} \qquad \frac{\begin{matrix}[\neg A]\\ \vdots \\ \bot\end{matrix}}{A}\text{RAA} \qquad \frac{\begin{matrix}[A]\\ \vdots \\ \bot\end{matrix}}{\neg A}\neg\text{i}$$

$$\frac{\forall x.\phi}{\phi_t^x}\forall\text{e} \qquad \frac{\phi_y^x}{\forall x.\phi}\forall\text{i} \qquad \frac{\phi_t^x}{\exists x.\phi}\exists\text{i} \qquad \frac{\begin{matrix}[\phi_y^x] & \\ \vdots & \\ \psi & \exists x.\phi\end{matrix}}{\psi}\exists\text{e}$$

$$\frac{}{t = t}\text{RFL} \qquad \frac{t_1 = s_1 \dots t_n = s_n}{f(t_1, \dots, t_n) = f(s_1, \dots, s_n)} \qquad \frac{t_1 = s_1 \dots t_n = s_n}{\phi_{t_1,\dots,t_n}^{x_1,\dots,x_n} \to \phi_{s_1,\dots,s_n}^{x_1,\dots,x_n}}\text{sub}\phi$$

Un esempio di traduzione

- "La persona più alta del mondo vive in Marocco"
  - $\mathcal{L} = \left(\{Marocco\}, \left\{\underbrace{P^1}_{\text{persona}}, \underbrace{\leq^2}_{\text{minore}}, \underbrace{V^2}_{\text{vive}}\right\}, \left\{\underbrace{h^2}_{\text{altezza}}\right\}\right)$
  - $\exists x.\left(P(x) \land V(x, Marocco) \land \underbrace{\forall y.\big(\big(P(y) \land \neg(x = y)\big) \to h(y) < h(x)\big)}_{\text{tutte le altre sono più basse}}\right)$

Esercizi sulle regole di inferenza

- $\vdash \neg\Big(\exists x.\forall y.\Big(\big(\neg P(x,x) \to P(x,y)\big) \to \big(P(x,x) \to \neg P(x,y)\big)\Big)\Big)$
- $\forall x.\exists y.(x < y \to x \in y), \exists x.\forall y.(x \in y \to y \subseteq x) \vdash \exists x.\exists y.(x < y \to y \subseteq x)$
- $(A \land B) \lor (\neg C \lor B) \vdash C \to B$
- $p \lor q, p \to q, q \to p, \neg(p \land q) \vdash R$
- $\neg\exists x.\neg(x = m) \vdash \forall x.\forall y.\big(P(x) \to P(y)\big)$

π_1 e π_2

- $\pi_1: \big(\forall x.P(x) \to \exists x.Q(x)\big), \forall x.P(x) \vdash \exists x.\big(P(x) \to Q(x)\big)$

$$\cfrac{\cfrac{\cfrac{[Q(y)]_1}{P(y) \to Q(y)}\to\text{i}}{\exists x.\big(P(x) \to Q(x)\big)}\exists\text{i} \qquad \cfrac{[\forall x.P(x)]_{hp} \quad [\forall x.P(x) \to \exists x.Q(x)]_{hp}}{\exists x.Q(x)}\to\text{e}}{\exists x.\big(P(x) \to Q(x)\big)}\exists\text{e}_1$$

- $\pi_2: \neg\forall x.P(x) \vdash \exists x.\big(P(x) \to Q(x)\big)$

$$\cfrac{\cfrac{\cfrac{\cfrac{\cfrac{[P(y)]_2 \quad [\neg P(y)]_1}{\bot}}{Q(y)}\bot\text{e}}{P(y) \to Q(y)}\to\text{i}_2}{\exists x.\big(P(x) \to Q(x)\big)}\exists\text{i} \qquad \cfrac{[\neg\forall x.P(x)]_{hp} \quad \cfrac{\cfrac{\cfrac{\cfrac{\cfrac{[\neg P(y)]_4}{\exists x.\neg P(x)}\exists\text{i} \quad [\neg\exists x.\neg P(x)]_3}{\bot}\neg\text{e}}{P(y)}\text{RAA}_4}{\forall x.P(x)}\forall\text{i}}{\cfrac{\bot}{\exists x.\neg P(x)}\text{RAA}_3}\neg\text{e}}{\exists x.\big(P(x) \to Q(x)\big)}\exists\text{e}_1$$