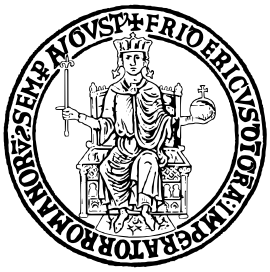
Università degli Studi di Napoli Federico II



Scuola Politecnica e delle Scienze di Base

Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione

Corso di Laurea Magistrale in Informatica

Appunti: Logica per Informatica

Autori

Balzano Giuseppe
Converso Riccardo
Di Geronimo Antonio
Mangiacapra Luigi
Vitale Fabrizio

Anno Accademico 2022–2023

Indice

Formulario

`https://leanprover.github.io/logic_and_proof/nd_quickref.html`

- **Regole per $\wedge$**

$$\frac{A \qquad B}{A \wedge B} \wedge i \qquad \frac{A \wedge B}{A} \wedge e_l \qquad \frac{A \wedge B}{B} \wedge e_r$$

- **Regole per $\vee$**

$$\frac{A}{A \vee B} \vee i_l \qquad \frac{B}{A \vee B} \vee i_r \qquad \frac{A \vee B \quad \overset{[A]}{\vdots}\ C \quad \overset{[B]}{\vdots}\ C}{C} \vee e$$

- **Regole per $\rightarrow$**

$$\frac{A \qquad A \rightarrow B}{B} \rightarrow e \qquad \frac{\overset{[A]}{\vdots}\ B}{A \rightarrow B} \rightarrow i$$

- **Regole per $\neg$**

$$\frac{A \qquad \neg A}{\bot} \neg e \qquad \frac{\overset{[A]}{\vdots}\ \bot}{\neg A} \neg i$$

- **Regole per $\bot$**

$$\frac{\bot}{\vdots\ A} \bot e \qquad \frac{\overset{[\neg A]}{\vdots}\ \bot}{A} RAA$$

- **Regole per $\forall$**

$$\frac{A(x)}{\forall z.\ A(z)} \forall i$$

La variabile x deve essere non libera in tutte le assunzioni non ancora scartate e non libera in $\forall z.\ A(z)$

$$\frac{\forall z.\ A(z)}{A(t)} \forall e$$

La variabile t può essere una qualsiasi variabile che non deve andare in clash con nessuna variabile vincolata in A e deve essere libera.

- **Regole per $\exists$**

$$\frac{A(t)}{\exists x.\ A(x)} \exists_i$$

La variabile t può essere una qualsiasi variabile che non deve andare in clash con nessuna variabile vincolata in A e deve essere libera.

$$\frac{\overset{[\overline{A(y)}]^1}{\vdots}\ \psi \qquad \exists x.\ A(x)}{\psi} \exists_e$$

La variabile y non deve essere libera in ψ e non deve essere libera in qualche assunzione non ancora scartata (ma può esserlo in 1).

Capitolo 1

Lezione 1 - Introduzione

1.1 Asserzione

Un'asserzione è banalmente una frase composta da diverse **premesse** e **conclusioni**.

1.2 Argomentazione

Più asserzioni invece compongono un'**argomentazione** (o ragionamento). In generale, un argomentazione (cioè se tutte le premesse sono vere) è valida se nelle qualsiasi situazioni del mondo è valida e la conclusione non può essere falsa.

Esempio 1.2.1

1. Tutti gli uomini sono immortali
2. Socrate è un uomo
   QUINDI
3. E' immortale

Nell'esempio precedente quindi i punti 1 e 2 rappresentano le premesse, mentre il punto 3 è la conclusione.

1.3 Validità

Dobbiamo tenere presente che il concetto di verità in ambito logico non ci interessa più di tanto, il concetto fondamentale è la **validità**, ovvero la coerenza tra la risposta data e le premesse stabilite nell'asserzione.

1.4 Tipi Di Ragionamento

Una prima classificazione empirica che possiamo fare è la suddivisione in:

- Ragionamento Deduttivo
- Ragionamento Induttivo
- Ragionamento Abduttivo

1.4.1 Ragionamento Deduttivo

Per ragionamento **deduttivo** intendiamo quando la conclusione di un'asserzione è una diretta conseguenza esplicita delle premesse e la conclusione non aggiunge ne toglie ulteriori informazioni al senso dell'asserzione.

Esempio 1.4.1

1. Carlo o non dorme o sogna
2. Carlo dorme
3. Se Carlo sogna allora è felice
   QUINDI
4. Carlo sogna
   QUINDI
5. Carlo è felice

Possiamo notare come le conclusioni (4 e 5) sono una diretta conseguenza delle premesse che non aggiungono nulla all'asserzione.

1.4.2 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.

Esempio 1.4.2

1. Gran parte degli studenti ama la logica
2. Ciro odia la logica
   QUINDI
3. Ciro non è uno studente

Possiamo notare che la conclusione non è implicita nei fatti descritti dalle premesse.

1.4.3 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. Differisce dal

principio induttivo in quanto non c'è il passo induttivo. Proprio per questo, questo prende il nome di **induzione malformata**.

Esempio 1.4.3

1. Tutti i bar di Napoli in cui sono andato servono un caffè decente
2. Questo è un nuovo bar di Napoli appena aperto
   QUINDI
3. Servirà un caffè decente

1.5 Asserzione Corretta

Se tutte le premesse sono vere e le conclusioni vere, allora parleremo di asserzione **corretta**.

1.6 Consistenza

Per consistenza intendiamo che nell'insieme di affermazioni 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.

1.7 Inconsistenza

Per inconsistenza invece quando esiste almeno una contraddizione interna tra le affermazioni. In altre parole, se in tutte le situazioni del mondo, c'è almeno un'affermazione che è **falsa**.

1.8 Equivalenza

Parleremo invece di **equivalenza** tra due affermazioni se e solo se sono entrambe vere (o false) nelle stesse situazioni. In altre parole, devono significare la stessa cosa.

Esempio 1.8.1

1. Carlo non dorme o è felice
2. Se Carlo dorme è felice

Queste due affermazioni sono equivalenti.

Capitolo 2

Lezione 2

2.1 Introduzione alla Conseguenza Logica

Diremo che una certa asserzione Φ è conseguenza logica di Γ **se e solo se** $(\Gamma \cup \{\neg\Phi\})$ è inconsistente.

In altre parole, l'insieme $(\Gamma \cup \{\neg\Phi\})$ è **inconsistente** se non esiste un'interpretazione che renda vere tutte le asserzioni in Γ e allo stesso tempo renda falsa Φ. Da questo possiamo dire che la consistenza non è legata al concetto di conseguenza logica.

Esempio 2.1.1

1. Se Carlo è italiano e John non è spagnolo, allora Rachel è tedesca
2. Se Rachel è tedesca allora Lucy è francese o John è spagnolo
3. Se Lucy non è francese allora Carlo è italiano
4. John non è spagnolo
   QUINDI
5. Lucy è francese

Dimostrazione 2.1.1

Per poter dimostrare quindi, per il principio precedente, neghiamo la 5 (Φ) ottenendo 6 ($\neg\Phi$).

6. Lucy non è francese

Per la 3 Carlo è italiano, quindi la 1 è vera (considerando e sapendo che la 4 è vera), quindi Rachel è tedesca. Ma per la 2 o Lucy è francese o John è spagnolo, ma avevamo supposto all'inizio con la 6 che Lucy non è francese $\longrightarrow$ John è spagnolo ma questo non è possibile per la 4.

Quindi possiamo affermare che 5 (Φ) è vera e 6 ($\neg\Phi$) è falsa.
Questo significa che la negata della 5 è falsa (cioè la 6) e che quindi l'insieme delle affermazioni $\Gamma = \{1, 2, 3, 4\} \cup \{\neg\Phi\}$ è inconsistente, perchè appunto c'è un'affermazione che è falsa che è proprio $\neg\Phi$.

Allora possiamo concludere dicendo che 5 (Φ) è conseguenza logica di Γ.

2.2 Strutture

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

Struttura 1:

1. Tutti i P sono Q
2. X è P
   QUINDI
3. X è Q

Struttura 2:

1. Se non A allora B
2. Non A
   QUINDI
3. B

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

2.3 Logica

Una **logica** è costituita da 3 elementi:

1. **Linguaggio** (Sintassi): l'insieme di simboli e regole per la loro combinazione, che definiscono le formule ben formate all'interno della logica.
2. **Semantica**: fornisce un'interpretazione del linguaggio, cioè assegna significato alle formule ben formate.
3. **Calcolo Deduttivo** (o Meccanismo Inferenziale, cioè un insieme di regole o anche dette inferenze): 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).

2.4 Programma Del Corso

Adesso facciamo un'introduzione al programma, poi nella prossima lezione entreremo nel vivo del corso. In genere, in base a come variano i punti 1, 2 e 3 precedenti è possibile ottenere logiche diverse. Nel nostro caso, studieremo nello specifico due tipi di logiche:

- Logica Proposizionale (PL)

- Logica di primo ordine (FOL)

La PL è inclusa nella FOL ed entrando nello specifico la logica **proposizionale** è composta dagli operatori booleani, mentre la logica di **primo ordine** è composta da variabili, quantificatori, $\forall$, $\exists$...

2.5 PL: Logica Proposizionale

Gli elementi cardine trattati sono chiamati **AP** (proposizioni atomiche, cioè proposizioni non ulteriormente decomponibili in altre più semplici).

Per esempio:

$$4 \leq 6$$

è una proposizione atomica, e potremmo dire che è facile immaginare che l'insieme di AP sia potenzialmente infinito (ovvero perchè 4 e 6 $\in N$).

Gli **operatori** utilizzati sono $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\iff$. E ovviamente possiamo dire che più AP possono essere combinate attraverso gli operatori.

Capitolo 3

Lezione 3

Nella lezione precedente quindi abbiamo accennato a come una logica è terna composta da 3 elementi fondamentali: *linguaggio*, *semantica* e *calcolo* (L, M, C).

Nella lezione di oggi tratteremo di L e M.

3.1 Linguaggio L

L 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).

L'**alfabeto** sarà costituito dai seguenti simboli:

$$\{(\ ,)\ \neg\ ,\vee\ ,\wedge\ ,\Leftarrow\ ,\iff\ ,\dots\ ,[\top\ ,\bot\ ,\oplus\ ]\}$$

L'insieme dei **simbolo logici** è un insieme finito ed è composto dagli elementi come: $\{\neg, \wedge, \vee, \rightarrow, \iff\}$, comunemente anche detti **connettivi logici**. Tali operatori logici sono dotati di un **arietà**, cioè il numero di "parametri" su cui lavorano.

Fatta eccezione per le parentesi tonde $(\ ,)$ che non sono connettivi logici, ma permettono sia di definire un'univoca interpretazione sintattica sia di definire le precedenze. $(a \cdot b) + c \neq a \cdot (b + c)$

Per **variabili proposizionali** (o proporzioni atomiche) intenderemo i simboli non logici arbitrari, per esempio: $A_1, A_2, ..., A_n$. Sono chiamate così in quanto il loro significato non è determinato logicamente, cioè che non è fissato dalla logica.

3.2 Espressioni

Le espressioni sono sequenze **finite** di simboli dell'alfabeto. Per discriminare tra espressioni sensate e espressioni non sensate useremo il concetto di **ben formatezza** (WFF: Well Formed Formula) o anche dette **regole** di ben formatezza.

Esempio 3.2.1

$$(A_1 \wedge A_2) \vee A_3 \text{(ben formata)}$$

$$\neg A_1 \vee \wedge A_2 \text{(mal formata)}$$

Le espressioni **ben formate** sono definite come **formule** ben formate della PL (Propositional Logic).

Denoteremo con **W** l'insieme delle **formule ben formate** di PL.
Mentre denoteremo con **AP** l'insieme delle **proposizioni atomiche** $(A_1, A_2, ..., A_n)$.

3.2.1 L'insieme W: Definizione Induttiva

Denoteremo con **W** l'insieme delle **formule ben formate** (WFF) di PL.

Definizione 3.2.1

1. $W \supseteq AP$ (caso base)
2. Se ϕ_1 e $\phi_2 \in W$, allora

   $[(\phi_1 \wedge \phi_2), (\phi_1 \vee \phi_2), (\phi_1 \rightarrow \phi_2), (\phi_1 \iff \phi_2), (\neg\phi_1)] \in W$ (caso induttivo)

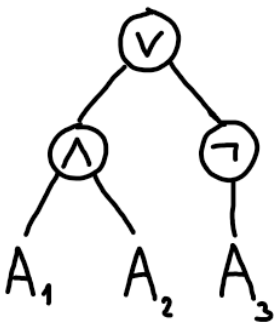
3. Nient'altro appartiene a W (regola di chiusura)

Spesso indicheremo l'insieme W direttamente con PL o con L (linguaggio).

3.3 Rappresentazione ad Albero Binario

Una qualsiasi espressione ben formata può essere rappresentata con come un albero binario in cui le foglie rappresentano le proposizioni atomiche (AP), mentre i nodi interni rappresentano i connettivi logici. Potremo dire che un albero con queste caratteristiche rappresenterà sempre una formula ben formata.

Per esempio data la seguente formula ben formata: $(A_1 \wedge A_2) \vee (\neg A_3)$



3.4 Dimostrazioni per induzione strutturale

In questa sezione tenteremo di effettuare una dimostrazione per induzione strutturale. In generale una dimostrazione per induzione strutturale segue sia le regole di costruzione sia la struttura degli oggetti.

3.5 Funzione Sub(p)

Presa una formula $\phi \in W$, una **sottoformula** è una funzione definita come:

$$Sub : W \to 2^W (\mathbb{P}(W) : \text{insieme delle parti di W})$$

Definizione 3.5.1

1. $Sub(p) = \{p\}$ se $p \in AP$
2. - $Sub(\phi_1 \circ \phi_2) = \{\phi_1 \circ \phi_2\} \cup Sub(\phi_1) \cup Sub(\phi_2)$
   - $Sub(\neg\phi) = \{\phi\} \cup Sub(\phi)$

   con $\circ \in \{\neg, \wedge, \vee, \ldots\}$

3.6 Funzione Len(p)

Presa una formula $\phi \in W$, la **lunghezza** di una formula è una funzione come:

$$Len : W \to N$$

Definizione 3.6.1

1. $Len(p) = 1 \qquad \forall p \in AP$
2. - $Len(\phi_1 \circ \phi_2) = 1 + Len(\phi_1) + Len(\phi_2)$
   - $Len(\neg\phi_1) = 1 + Len(\phi_1)$

3.7 Proposizione: $|Sub(\phi)| = Len(\phi)$

Vogliamo dimostrare che $\forall \phi \in W, \quad |Sub(\phi)| = Len(\phi)$

Dimostrazione 3.7.1

Usiamo l'induzione strutturale

- **Caso base:**
  $\forall p \in AP, \quad |Sub(p)| = Len(p)$

- **Caso induttivo:**
  Presa $\phi \in W \setminus AP$ (cioè che ϕ non è una proposizione atomica)

  Sia $\phi = \phi_1 \circ \phi_2$

  Allora:

  $Sub(\phi) \stackrel{def}{=} \{\phi\} \cup Sub(\phi_1) \cup Sub(\phi_2)$
  $|Sub(\phi)| = 1 + |Sub(\phi_1)| + |Sub(\phi_2)|$
  $\rightarrow$ Per l'ipotesi induttiva (caso base)
  $|Sub(\phi)| = 1 + Len(\phi_1) + Len(\phi_2)$
  $|Sub(\phi)| = Len(\phi) = Len(\phi_1 \circ \phi_2)$

3.8 Modello m del Linguaggio L

Il concetto di **modello** è definito per discriminare tra le AP vere da quelle false. Denoteremo con M la **classe dei modelli**.

Denoteremo m un **modello** (che può essere finito o infinito):

- $m \in M$ (classe dei modelli)
- $m \subseteq AP$ (un sottoinsieme delle proposizioni atomiche)

Quindi concettualmente $\forall p \in m \iff p$ è vera in m.

3.8.1 Concetto di soddisfacibilità e di insoddisfacibilità

Per specificare che un modello $m \in M$ **soddisfa** (rende vero) $\phi \in W$ (o anche denotato con PL) utilizzeremo il simbolo $\mathbf{m \models \phi}$.

Tale simbolo specificherà la relazione di **soddisfazione**. Al contrario $\mathbf{m \not\models \phi}$ specificherà la relazione di **insoddisfazione**.

E' importante osservare che tale relazione di soddisfacibilità in questo caso è definita come un sottoinsieme $\models \subseteq \boldsymbol{M}$ x $\boldsymbol{W}$.

3.8.2 Soddisfacibilità di un modello m: $m \models \phi$

Diremo che m **soddisfa** ϕ (con $\phi \in W$) con la seguente notazione $m \models \phi$.

Definizione 3.8.1

Preso $\phi \in W$, diremo che $\boldsymbol{m \models \phi}$ (m soddisfa ϕ) se e solo se:

1. $\phi \in m$ se $\phi \in AP$
2. - $m \not\models \psi$ se $\phi = \neg\psi$

- $m \models \phi_1$ e $m \models \phi_2$ se $\phi = \phi_1 \wedge \phi_2$
- $m \models \phi_1$ o $m \models \phi_2$ se $\phi = \phi_1 \vee \phi_2$
- se $m \models \phi_1$ allora $m \models \phi_2$ se $\phi = \phi_1 \rightarrow \phi_2$

3.8.3 Concetto di Soddisfacibile

Diremo che ϕ (con $\phi \in W$) è **soddisfacibile** se e solo se $\exists m \in M$ tale che $m \models \phi$

3.8.4 Concetto di Validità

Diremo che ϕ (con $\phi \in W$) è **valido** se e solo se $\forall m \in M$ $m \models \phi$

3.8.5 Concetto di Insoddisfacibile

Diremo che ϕ (con $\phi \in W$) è **insoddisfacibile** (o contraddittorio) se e solo se non è soddisfacibile. In altre parole, cioè se **non esiste** un modello $m \in M$ tale che $m \models \phi$ (m soddisfi ϕ).

3.8.6 Concetto di Conseguenza Logica

Dato un insieme di premesse Γ diremo che ϕ è **conseguenza logica** di Γ con la seguente notazione: $\boldsymbol{\Gamma \models \phi}$.

E' importante osservare che il simbolo $\models$ in questo caso è utilizzato per specificare la relazione di conseguenza logica che è definita come un sottoinsieme $\models \subseteq \boldsymbol{2^W}$ x $\boldsymbol{W}$.

Definizione 3.8.2

Sia $\Gamma \subseteq W$ e $\phi \in W$, diremo che $\boldsymbol{\Gamma \models \phi}$ (ϕ è c**onseguenza logica** di Γ) se e solo se:

$\forall m \in M$, $m \models \Gamma$ allora $m \models \phi$

Dire che $m \models \Gamma$ significa che $\forall \psi \in \Gamma$, $m \models \psi$

Questo perchè ci basiamo sul principio visto nelle lezioni precedenti in cui se abbiamo tutte premesse vere allora avremo un'affermazione vera, quindi se m soddisfa tutte le premesse allora soddisferà sicuramente anche la conclusione ϕ.

Capitolo 4

Lezione 4

Nell'ultima lezione abbiamo visto che

$$\Gamma \models \phi \iff \forall m \in M,\ m \models \Gamma \Rightarrow m \models \phi$$

Inoltre, se prendessimo $m \models \Gamma$ allora sicuro non può succedere che $m \not\models \phi$.
In altre parole, se $\Gamma \models \phi$ e $m \models \Gamma$ e allora $m \models \phi$ e che $m \not\models \neg\phi$ (se m soddisfa ϕ allora non soddisfa $\neg\phi$).

Di conseguenza l'insieme $(\Gamma \cup \{\neg\phi\})$ è insoddisfacibile (o non ha modello). In altre parole, non può esistere un modello che renda vero Γ e non renda vero ϕ.

Definizione 4.0.1

$$\Gamma \cup \{\neg\phi\} \textbf{ è insoddisfacibile} \iff \Gamma \models \phi$$

Questo vale perchè sappiamo che: $\Gamma \models \phi$ e $m \models \Gamma$ allora $m \models \phi$ e che $m \not\models \neg\phi$

4.1 Equivalenza Logica

Definizione 4.1.1

Due formule ϕ e ψ sono **logicamente equivalenti** se e solo se

$$\forall m \in 2^{AP},\ m \models \phi \iff m \models \psi$$

Che è **equivalente** a dire

$$\{\phi\} \models \psi \text{ e } \{\psi\} \models \phi$$

Dobbiamo avere entrambi i versi delle conseguenze logiche, quindi se tutti i modelli rendono entrambe le formule entrambe vere o entrambe false.

Abbiamo visto che quindi tutte le relazioni che abbiamo visto in queste lezioni, come relazione di inconsistenza, relazione di conseguenza logica, validità o relazione di equivalenza, sono tutte riconducibili alla relazione di soddisfazione, quindi saper determinare una **relazione di soddisfazione** è un problema importante.

4.2 Da logica dei modelli a logica booleana

Un modello $m \in 2^{AP}$ del linguaggio L può avere infinite proposizioni atomiche (PA), però una qualsiasi formula $\phi \in W$ (anche denotato con PL) in realtà contiene solo un insieme **finito** di simboli. Supponendo che $m \models \phi$, ci poniamo il seguente problema:

> "*Abbiamo bisogno di tutta l'informazione contenuta nel modello m per decidere se la formula ϕ è vera o falsa?*"

Ovviamente in linea di principio non siamo in grado di dirlo, dovremmo vedere la semantica e determinare se riesco a togliere qualcosa da m lasciando invariata la relazione di soddisfazione con ϕ. In altre parole, supponendo che m **soddisfi** ϕ ($m \models \phi$), se buttassimo via cose (informazione) da m, vorremmo provare che m' (ovvero il modello ridotto) soddisfi ancora o meno ϕ.

Quello che possiamo fare però è:

- Partire dalla formula ϕ e vedere qual è l'informazione minima che serve per poter decidere se la formula è vera o falsa
- Infine, creare un modello proprio per questa formula

Introduciamo quindi il concetto di "**modello di una formula**":

Definizione 4.2.1

Allora data una formula ϕ, definiremo con:

$$AP(\phi) = AP \ \cap Sub(\phi)$$

L'insieme finito delle proposizioni atomiche AP contenute in ϕ

Inoltre vale che $|AP(\phi)| \leq \text{len}(\phi)$

Questa funzione $AP(\phi)$ colleziona tutte le **proposizioni atomiche** contenute nella formula ϕ. É importante notare che $AP(\phi)$ è un insieme finito di simboli.

Quindi il punto a cui vorremo arrivare è che se una formula non contiene una determinata proposizione atomica, i valori di quella proposizione atomica sono effettivamente **irrilevanti** per la validità della formula.

Esempio 4.2.1

Consideriamo che un modello con le seguenti asserzioni:

1. Il tempo è brutto
2. Il meteo porta pioggia
3. In aula ci sono 30 studenti

La frase numero 3 è del tutto irrilevante se l'affermazione ci dice se piove o meno.

4.2.1 Assegnamento α

Il concetto di modello che possiamo ritagliare a partire da una formula è il concetto di **assegnamento**.

Definizione 4.2.2

Definiremo α **assegnamento** per ϕ con una funzione:

$$\alpha : AP(\phi) \to \{0, 1\}$$

Quindi è una funzione che prende una proposizione contenuta nella formula e gli assegna un valore di 0 o 1.

4.2.2 Valore di verità di α

A questo punto possiamo definire il concetto di **valore di verità** che un assegnamento α assegna ad una formula arbitraria ϕ.

Definizione 4.2.3

Data la formula ϕ, definiremo con:

$$val^{\alpha}(\phi) : W \to \{0, 1\}$$

Dove $val^{\alpha}(\phi)$ specifica il valore di verità che α assegna ad una formula ϕ

Dove dato un modello ti restituisce la risposta che sia 0 o 1.

Definiremo il comportamento di $val^{\alpha}(\phi)$ come segue:

1. **Se** $\phi = A$ con $A \in AP(\phi)$ **allora** $val^{\alpha}(\phi) = \alpha(A)$
2. **Se** $\phi = \neg\psi$ **allora** $val^{\alpha}(\phi) = 1 - val^{\alpha}(\psi)$
3. **Se** $\phi = \psi_1 \wedge \psi_2$ **allora** $val^{\alpha}(\phi) = min\{val^{\alpha}(\psi_1), val^{\alpha}(\psi_2)\}$
4. **Se** $\phi = \psi_1 \vee \psi_2$ **allora** $val^{\alpha}(\phi) = max\{val^{\alpha}(\psi_1), val^{\alpha}(\psi_2)\}$
5. **Se** $\phi = \psi_1 \to \psi_2$ **allora** $\mathrm{val}^{\alpha}(\phi) = \begin{cases} 1 & \text{se } \mathrm{val}^{\alpha}(\psi_1) \leq \mathrm{val}^{\alpha}(\psi_2) \\ 0 & \text{altrimenti} \end{cases}$

Proposizione 4.2.1

Sia $\phi \in W$ (una qualsiasi formula: $\forall \phi \in W$)

$$\text{Se } \exists\, m \in M : m \models \phi \iff \exists\, \alpha : AP(\phi) \to \{0,1\} : val^{\alpha}(\phi) = 1$$

Questa formula si traduce in, se esiste almeno un modello m che soddisfa la formula ϕ se e solo se esiste almeno un assegnamento α che ci farà ottenere 1 come risultato della formula ϕ.

Dimostrazione 4.2.1

Dim ⇒

Per ipotesi prendiamo un modello $m \models \phi$, dovremmo costruire un

$$\alpha_m : AP(\phi) \to \{0,1\}$$

Definiremo il nostro α_m nel seguente modo:

$$\forall A \in AP(\phi),\ \alpha_m : \begin{cases} 1, \text{ Se } A \in m \\ 0 \text{ altrimenti} \end{cases}$$

Dobbiamo dimostrare ora che $val^{\alpha_m}(\phi) = 1$.
Faremo la dimostrazione per **induzione strutturale**.

1. Se $\phi = A$ con $A \in AP$ (caso base)

$$m \models \phi \iff \phi \in m \iff \alpha(\phi) = 1 \iff val^{\alpha}(\phi) = 1$$

2. – $\phi = \psi_1 \wedge \psi_2$
$m \models \phi \iff m \models \psi_1$ e $m \models \psi_2 \iff \psi_1 \in m$ e $\psi_2 \in m \iff \alpha(\psi_1) = 1$ e $\alpha(\psi_2) = 1 \overset{\text{ipotesi induttiva}}{\iff} val^{\alpha}(\psi_1) = 1$ e $val^{\alpha}(\psi_2) = 1 \iff min\{val^{\alpha}(\psi_1),\ val^{\alpha}(\psi_2)\} = 1 \iff val^{\alpha_m}(\phi) = 1$

– $\phi = \neg\psi$
$m \models \phi \iff m \not\models \psi \iff \psi \notin m \iff \alpha_m(\psi) = 0 \overset{\text{ipotesi induttiva}}{\iff} val^{\alpha_m}(\psi) = 0 \iff 1 - val^{\alpha_m}(\psi) = 1 \iff val^{\alpha_m}(\neg\psi) = 1$

Dim ⇐

Questa dimostrazione è lasciata allo studente

4.3 Tabelle di verità

Dalle assegnazioni dei vari α e dal risultato ottenuto con *val* essenzialmente otteniamo le **tabelle di verità**, possiamo vederle come un modo alternativo di rappresentare la semantica che abbiamo descritto nella sezione precedente. In altre parole, le tabelle di verità sono un altro modo di vedere gli assegnamenti α.

Di seguito gli esempi più rilevanti:

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

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

A	B	$A \rightarrow B$
0	0	1
0	1	1
1	0	0
1	1	1

Possiamo osservare che tutte le tabelle avranno $2^{|AP(\phi)|}$ righe, dove $|AP(\phi)|$ è la cardinalità dell'insieme delle proposizione atomiche della formula ϕ.

Per decidere la soddisfacibilità di una formula basta costruire la tabella di verità e osservare che:

- Se c'è almeno un 1 diremo che la formula è **soddisfacibile**
- Se vi fossero tutti 1 diremo che la formula è **valida** (anche detta **tautologia**)
- Se invece vi fossero tutti 0 la formula non è **soddisfacibile**

Quindi possiamo concludere che se compare almeno un 1 nella tabella di verità allora la formula è **soddisfacibile** e quindi per definizione $\exists$ un m che soddisfa la formula ϕ.

Capitolo 5

Lezione 5

Definiamo alcune **tautologie notevoli**:

- $(A \wedge (B \vee C)) \iff ((A \wedge B) \vee (A \wedge C))$
- $(A \vee (B \wedge C)) \iff ((A \vee B) \wedge (A \vee C))$
- $\neg\neg A \iff A$
- $\neg(A \to B) \iff (A \wedge \neg B)$
- $\neg(A \iff B) \iff (A \wedge \neg B) \vee (\neg A \wedge B)$
- $\neg(A \wedge B) \iff (\neg A \vee \neg B)$ [*De Morgan*]
- $\neg(A \vee B) \iff (\neg A \wedge \neg B)$ [*De Morgan*]
- $(A \to B) \iff (\neg A \vee B)$
- $(A \vee \neg A) \equiv \top$ [*Legge del terzo escluso*]
- $(A \wedge \neg A) \equiv \bot$ [*Contraddizione*]
- $(A \to B) \iff (\neg B \to \neg A)$ [*Contrapposizione*]
- $(A \to (B \to C)) \iff ((A \wedge B) \to C)$
- $(A \wedge (A \to B)) \to B$ [*Modus Ponens*]
- $(\neg B \wedge (A \to B)) \to \neg A$ [*Modus Tollens*] (basato sulla Contrapposizione)
- $(A \to \bot) \iff \neg A$
- $((A \to B) \wedge (A \wedge \neg B)) \to \bot$

Le leggi di De Morgan permettono di "spingere" dentro la negazione, cioè la negazione verrà "propagata" all'interno e verrà applicata alle AP (proposizioni atomiche) della formula (**NNF**: Forma Normale della Negazione).

5.1 Dalla logica ai circuiti

Data una funzione $f(x) = x + 1$ definiremo $f(x)$ la **definizione intenzionale** della funzione. Definiremo le coppie $(input, output)$ come la **definizione estensionale** della funzione $f(x)$.

$$(0, 1)$$
$$(1, 2)$$
$$(2, 3)$$
$$...$$
$$(x, x + 1)$$

Ogni circuito combinatorio calcola funzioni booleane:

$$f : \{0, 1\}^k \to \{0, 1\}$$

C'è una correlazione tra circuiti e espressioni booleane (e viceversa).

Il dominio della funzione f, cioè $\{0, 1\}^k$ è finito (supponendo k finito) e la sua cardinalità è $|\{0, 1\}^k| = 2^k$.

Da un dominio finito possiamo creare una funzione f che definisce un output. Quindi dati k elementi in input possiamo generare la seguente tabella:

x_1 x_2 ... x_k	f
0 0 ... 0	0
0 0 ... 1	1
0 0 ... 0	0
1 0 ... 1	1

La rappresentazione tabellare generata è la definizione estensionale della funzione, ma tale rappresentazione è una vera e propria tabella di verità. Quindi f è una funzione booleana e può essere espressa con una **espressione booleana**.

Per rendere vera f per esempio avremo la seguente espressione booleana:

$$(\neg x_1 \wedge \neg x_2 \wedge ... \wedge x_k) \vee (x_1 \wedge \neg x_2 \wedge ... \wedge x_k)$$

Prenderemo tutte le espressioni che rendono vera f (cioè le righe della tabella contenente degli 1) e le concateneremo con gli $\vee$ (OR).

5.2 Forme Normali

5.2.1 NNF: Forma Normale Negativa

La NNF (Normal Negative Formula) è una forma di una formula logica dove $\neg$ (negazione) compare solo di fronte a **proposizioni atomiche** ($\forall p \in AP$).

Definiremo un dato $A \in AP$ come **letterale positivo**. Di conseguenza, definiremo $\neg A \in AP$ come **letterale negativo**.

5.2.2 DNF: Forma Normale Disgiuntiva

Una DNF (Disjunctive Normal Form) è una forma di una formula logica quando vi è una **grossa disgiunzione** (OR) di una o più congiunzioni (AND) di letterali.

In altre parole, è una "grande" disgiunzione di congiunzioni di letterali:

$$\bigvee_{i \in I} C_i$$

Dove C_i (formule) è detta **clausola congiuntiva**, cioè una grande congiunzione di letterali (come: $A \wedge \neg B \wedge C$).

Esempio 5.2.1

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

5.2.3 CNF: Forma Normale Congiuntiva

Una CNF (Conjunctive Normal Form) è una forma di una formula logica quando vi è una **grossa congiunzione** (AND) di una o più disgiunzioni (OR) di letterali.

In altre parole, è una "grande" congiunzione di disgiunzioni di letterali:

$$\bigwedge_{i \in I} C_i$$

Dove C_i (formule) è detta **clausola disguntiva**, cioè una grande disgiunzione di letterali (come: $A \vee \neg B \vee C$).

Esempio 5.2.2

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

Osservazione 5.2.1
Dobbiamo vedere quindi $\bigvee$ come una sommatoria ($\sum$) di più formule booleane (con $\wedge$), mentre $\bigwedge$ come una produttoria ($\prod$) di più formule booleane (con $\vee$).

5.3 Ogni formula è una CNF o una DNF?

Arrivati a questo punto ci verrebbe spontaneo porci la seguente domanda:

"*E' vero che ogni formula ϕ è una CNF o una DNF?*"

5.3.1 Proposizione DNF

Proposizione 5.3.1

Per quanto riguarda DNF è facilmente dimostrabile poichè ogni formula proporzionale è visibilie come una DNF attraverso le tabelle di verità (come abbiamo visto nella tabella di 5.1)

5.3.2 Proposizione CNF

Proposizione 5.3.2

Sia $\phi \in W$, ne costruiamo una tabella di verità (in modo casuale).

A	B	C	ϕ
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

In questo caso, a differenza del caso delle DNF in cui si selezionavano le espressioni (o formule) che mi generavano $\phi = 1$, selezioneremo le espressioni (o formule) che generano $\phi = 0$. In altre parole:

- Selezioneremo gli assegnamento a 0
- Faremo la congiunzione degli assegnamenti selezionati negandoli

Quindi faremo:

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

Applichiamo *De Morgan*

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

Quindi la formula è anche CNF, banalmente applicando *De Morgan* sui casi in cui ϕ risponde 0.

5.4 Insieme di connettivi completi

Possiamo osservare che l'insieme $\{\neg, \wedge, \vee\}$ è un insieme **completo**, cioè può esprimere qualsiasi formula (utilizzando solo quei 3 simboli). Sapendo che quindi questo è completo, posso costruirne un altro arbitrariamente e definirlo anch'esso completo.

Proposizione 5.4.1
Sia $X = \{\neg, \vee\}$ un insieme di connettivi arbitrario,
possiamo dire che X è **completo**.

Banalmente perchè possiamo costruire il connettivo mancante $\wedge$ in questo modo (*per De Morgan*):

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

Dimostrazione 5.4.1

Per poter effettuare una dimostrazione ci baseremo su un **algoritmo ricorsivo** sulla stuttura:

Dato $X_1 = \{\neg, \wedge, \vee\}$ e $X_2 = \{\neg, \vee\}$

Presa una formula $\phi \in \mathcal{L}_{X_1} \rightarrow T(\phi) \in \mathcal{L}_{X_2}$ (con T = traduzione)

- $\phi = A \in AP \ \rightarrow \ T(\phi) = A$
- $\phi = \neg\psi \ \rightarrow \ T(\phi) = \neg T(\psi)$
- $\phi = \psi_1 \vee \psi_2 \ \rightarrow \ T(\psi_1) \vee T(\psi_2)$
- $\phi = \psi_1 \wedge \psi_2 \ \rightarrow \ \neg(\neg T(\psi_1) \vee \neg T(\psi_2))$

Capitolo 6

Lezione 6

6.1 Connettivi Logici e Funzioni

Partiamo con il dire che ogni connettivo logico è collegato ad una funzione.

Esempio 6.1.1

$$\vee \to f_{\vee} : \{0,1\} \times \{0,1\} \longrightarrow \{0,1\}$$

Anche quelli unari sono collegati a funzioni.

Esempio 6.1.2

$$\neg \longrightarrow f_{\neg}$$

I connettivi logici **unari** possiamo vederli anche come connettivi **binari**.

Esempio 6.1.3

$\neg_1(A, B)$ restituisce il negato di A (negando A e lasciando B inalterato)

$\neg_2(A, B)$ restituisce il negato di B (negando B e lasciando A inalterato)

Arrivati a questo punto ci verrebbe spontaneo porci la seguente domanda:

"*Data una funzione $f : A \longrightarrow B$, qual è il numero di funzioni da A in B?*"

Dimostrazione 6.1.1

Per poter rispondere ci riferiamo a questo principio, data una funzione $f : A \to B$, il numero di funzioni f è $|B|^{|A|}$.

Nel nostro caso, considerando: $f_\vee : \{0,1\} \times \{0,1\} \longrightarrow \{0,1\}$

Allora $|A| = 2 \times 2 = 2^2$ e $|B| = 2$.

$$|B|^{|A|} = 2^{2^2} = 16$$

Tale risultato corrispondere proprio al numero di operatori che abbiamo in totale, cioè proprio 16: $\bot, \top, \pi_1, \pi_2, \neg_1, \neg_2, \wedge, \vee, \Rightarrow, \iff, \oplus, \Leftarrow, |, \downarrow, <, >$.

In particolare:

- $\pi_1(A, B) = A$ (PROIEZIONE): restituisce il valore di verità di A
- $\pi_2(A, B) = B$ (PROIEZIONE): restituisce il valore di verità di B
- $\oplus$ (OR ESCLUSIVO): basta che uno dei due sia vero per essere vero
- $\Leftarrow$ (IMPLICATO): è il verso opposto dell'implicazione
- $|$ (NAND): è vero quando l'AND è falso e viceversa
- $\downarrow$ (NOR): è vero quando l'OR è falso e viceversa

6.1.1 NAND e NOR completi

Diremo che il NAND e il NOR da soli sono **completi**:

- **NAND**
  - $\neg A \equiv A|A$
  - $A \vee B \equiv (A|A)|(B|B)$

    Dimostrazione 6.1.2

    $$A|B \overset{nand}{=} \neg(A \wedge B) \overset{DeMorgan}{\equiv} \neg A \vee \neg B$$

    Allora:

    $$(A|A) \mid (B|B) = \neg A \mid \neg B \overset{nand}{\equiv} \neg(\neg A \wedge \neg B) \overset{DeMorgan}{\equiv} A \vee B$$

  - $A \wedge B \equiv (A|B) \mid (A|B)$
- **NOR**
  - $\neg A \equiv A \downarrow A$
  - $A \wedge B \equiv (A \downarrow A) \downarrow (B \downarrow B)$
  - $A \vee B \equiv (A \downarrow B) \downarrow (A \downarrow B)$

6.2 Tautologie finitamente generabili

Arrivati a questo punto ci potremmo chiedere come generare nuove formule che rispettino specifiche proprietà (per esempio essere tautologia) a partire da altre formule.

Potremmo quindi porci la seguente domanda:

> "*L'insieme delle tautologie è finitamente generabile?*"

Per finitamente generabile intendiamo un insieme che può essere generato da un numero finito di elementi (per esempio l'insieme dei numeri naturali che può essere generato a partire da 1 e -1 attraverso l'operazione somma).

La **risposta** alla domanda è **SI** e lo dimostreremo successivamente.

6.2.1 Sostituzione

E' una funzione $f : AP \to W$(o PL) di riscrittura, anche detta di mappatura, utilizzata per mappare una proposizione atomica in una formula.

Esempio 6.2.1

$$\text{Sia } \sigma = \{A \leftarrow (A \wedge B), B \leftarrow (\neg A), C \leftarrow (C),\ \dots\}$$

$$\text{e sia } \phi = A \wedge \neg B \wedge C$$

$$\text{Allora } \phi_\sigma = (A \wedge B) \wedge \neg\neg A \wedge C$$

Dove ϕ_σ rappresenta la formula tradotta.

Andremo letteralmente a sostituire (in un unico step) le proposizioni atomiche con le formule descritte in σ. Attraverso questa funzione di sostituzione possiamo **generare nuove formule**.

Definizione 6.2.1

Preso $\sigma : AP \to W$ e presa una formula ϕ

Possiamo definire $\boldsymbol{\phi_\sigma}$ (la formula tradotta) in questo modo:

- se $\phi = A \in AP$, allora $\sigma(A)$
- se $\phi = \neg\psi$, allora $\neg(\psi_\sigma)$
- se $\phi = \psi_1 \wedge \psi_2$, allora $(\psi_{1\sigma}) \wedge (\psi_{2\sigma})$
- se $\phi = \psi_1 \vee \psi_2$, allora $(\psi_{1\sigma}) \vee (\psi_{2\sigma})$

In generale vale che:

$$(\psi_{1\sigma}) \circ (\psi_{2\sigma}) \quad \text{con } \circ \in \{\wedge, \vee, \neg, \to, \leftarrow, \dots\}$$

Avremo che:

$$(\psi_1 \circ \psi_2) = (\psi_{1\sigma}) \circ (\psi_{2\sigma})$$

Esempio 6.2.2

Possiamo verificare che $\phi \vee \neg\phi$ è una **tautologia**.

Presa $\sigma = \{A \leftarrow \phi\}$ la funzione di sostituzione,
allora $(A \vee \neg A)\sigma = (A_\sigma \vee \neg A_\sigma) = \sigma(A) \vee \neg\sigma(A) = \phi \vee \neg\phi$

Ora che abbiamo definito il concetto di sostituzione, è inevitabile porci la seguente domanda:

> "*Possiamo dire che partendo da una **tautologia** e facendo delle sostituzioni per ottenere una nuova formula, essa è ancora una tautologia?*"

Proposizione 6.2.1
Se ϕ è soddisfacibile **non** è assicurato che ϕ_σ è soddisfacibile

Dimostrazione 6.2.1

Per dimostrare questa proposizione basta trovare un esempio:

Sia $\phi = (A \wedge B)$ e sia $\sigma = \{A \leftarrow A, B \leftarrow \neg A\}$ la funzione di sostituzione,

Supponendo $\phi = (A \wedge B)$ soddisfacibile, se si preservasse la proprietà allora anche ϕ_σ dovrebbe esserlo. Ma possiamo osservare che $\phi_\sigma = (A \wedge \neg A)$, cioè la formula tradotta (risulta essere proprio la contraddizione per eccellenza), non è soddisfacibile.

Quindi in generale possiamo dire che **la sostituzione non preserva la proprietà di soddisfacibilità**.

Abbiamo osservato che per la proprietà di soddisfazione la sostituzione non preserva tale proprietà, ma allora:

> "*Per le tautologie? La sostituzione preserva la proprietà?*"

6.2.2 Teorema di Sostituzione

Teorema 6.2.1

Sia α un assegnamento di $\alpha : AP \to \{0,1\}$ e sia σ una funzione di sostituzione,

Presa una formula $A \in AP$, definiremo un nuovo assegnamento con:

$$\alpha^\sigma : \boldsymbol{\alpha^\sigma(A) = val^\alpha(\sigma(A))}$$

Enunciato Teorema:
Date ϕ, σ e α (assegnamento per ϕ_σ), allora vale che:

$$\boldsymbol{val^{\alpha^\sigma}(\phi) = val^\alpha(\phi_\sigma)}$$

In altre parole, se un assegnamento α soddisfa la **trasformata** (ϕ_σ), allora trovo un assegnamento α^σ che soddisfa la **non trasformata** (ϕ).

Conseguenza Teorema

Se ϕ_σ è soddisfacibile, allora ϕ è soddisfacibile.

Preso ϕ una **tautologia** e supponiamo **per assurdo** che ϕ_σ **non è una tautologia**.

Allora $\exists\alpha$ tale che $val^\alpha(\phi_\sigma) = 0$, ma questo vorrebbe dire che anche $val^{\alpha^\sigma}(\phi) = 0$ (per il teorema di sostituzione), ma questo è assurdo poiché abbiamo supposto che ϕ era una tautologia. Quindi ϕ_σ deve essere una tautologia.

Dimostrazione 6.2.2

- **Caso Base:** $\phi = A$
  $\boldsymbol{val^{\alpha^\sigma}(\phi)} = val^{\alpha^\sigma}(A) = \alpha^\sigma(A) \overset{def}{=} val^\alpha(\sigma(A)) = \boldsymbol{val^\alpha(\phi_\sigma)}$
- **Caso Induttivo:**
  - $\phi = \neg\psi$
    $\boldsymbol{val^{\alpha^\sigma}(\phi)} = val^{\alpha^\sigma}(\neg\psi) = 1 - val^{\alpha^\sigma}(\psi) \overset{ipotesi\ induttiva}{=} 1 - val^\alpha(\psi_\sigma) = val^\alpha(\neg(\psi_\sigma)) = val^\alpha((\neg\psi)\sigma) = \boldsymbol{val^\alpha(\phi_\sigma)}$
  - $\phi = \psi_1 \wedge \psi_2$
    $\boldsymbol{val^{\alpha^\sigma}(\phi)} = min(val^{\alpha^\sigma}(\psi_1), val^{\alpha^\sigma}(\psi_2)) \overset{ipotesi\ induttiva}{=} min(val^\alpha(\psi_{1\sigma}), val^\alpha(\psi_{2\sigma})) = val^\alpha(\psi_{1\sigma} \wedge \psi_{2\sigma}) = val^\alpha((\psi_1 \wedge \psi_2)\sigma) = \boldsymbol{val^\alpha(\phi_\sigma)}$

6.3 Teorema di Deduzione Semantica

C'è una relazione tra $\models$ (simbolo del meta-linguaggio che permette di esprime le proprietà del linguaggio) e " $\Longrightarrow$ " (simbolo del linguaggio).

Teorema 6.3.1

Dato $\Gamma \in 2^W$ (insieme di formule) e due formule ψ e $\phi \in W$ se:

$$\Gamma, \psi \models \phi \text{ allora } \Gamma \models (\psi \to \phi)$$

Osserviamo che $\Gamma, \psi = \Gamma \cup \{\psi\}$. Di conseguenza allora $\Gamma \cup \{\psi\} \models \phi$ significa che ϕ è **conseguenza logica**.

Dimostrazione 6.3.1

$\forall m \in M$ (classe dei modelli)

- se $m \models \Gamma$ (cioè che $\forall \xi \in \Gamma, m \models \xi$)
- e se $m \models \psi$

Allora $m \models \phi$ (m soddisfa/rende vera la formula ϕ) proprio per la definizione di conseguenza logica (perché abbiamo supposto da teorema che ϕ fosse conseguenza logica di $\Gamma \cup \{\psi\}$).

Quindi se associamo in questo modo:

- $A = m \models \xi$ $(\forall \xi \in \Gamma)$
- $\wedge$ ="e"
- $B = m \models \psi$
- $\to$="allora"
- $C = m \models \phi$

Possiamo scrivere la frase precedente come $(A \wedge B) \to C$, che è tautologia con: $A \to (B \to C)$

Quindi possiamo riscrivere la frase come:

Se $m \models \xi$ (per ogni $\xi \in \Gamma$) allora, se $m \models \psi$ allora $m \models \phi$

Noi sappiamo dalla teoria che $m \models \psi \to \phi \iff m \models \psi$ allora $m \models \phi$.
In altre parole, se prendessimo la tabella di verità dell'implicazione e sapendo che $m \models \psi$ (cioè B) è vero (m soddisfa la formula ψ significa proprio che m rende vera la formula), allora l'unico modo per ottenere vero come risultato dell'implicazione è che anche $m \models \phi$ sia vero (cioè che anche C è vero).

Quindi andando a sostituire nella nostra frase scriveremo:

Se $m \models \xi$ (per ogni $\xi \in \Gamma$) allora, $m \models \psi \to \phi$

Possiamo quindi concludere che:

$$\Gamma \models (\psi \to \phi)$$

Capitolo 7

Lezione 7

7.1 Da soddisfacibilità a validità, insoddisfacibilità e conseguenza logica

Nelle lezioni precedenti abbiamo analizzato il concetto di **soddisfacibilità**, in questa lezione vedremo come tale concetto sia forte per poter effettuare stime di:

1. Insoddisfacibilità
2. Validità
3. Conseguenza Logica

In altre parole, se ϕ è soddisfacibile, sapremo anche dire se è valida, insoddisfacibile e se è conseguenza logica. Quello che vorremmo fare sarà trasformare questi tre problemi (validità, insoddisfacibilià e conseguenza logica) in un problema di soddisfacibilità.

7.1.1 Insoddisfacibilità

Quello che vorremmo fare è partire dalla definizione di soddisfacibile, cioè esiste almeno un modello che soddisfi la formula $(\exists m : m \models \phi)$ e arrivare alla definizione di insoddisfacibilità, cioè non esiste nessun modello che soddisfi la formula $(\nexists m : m \models \phi)$ o anche $(\forall m : m \not\models \phi)$. Un modo per procedere è utilizzare l'algoritmo di NotSoddisfacibilità, cioè un algoritmo che verifichi che data una formula ϕ non esisti un modello che la soddisfi, cioè la renda vera.

$$\text{Insoddisfacibilità} \rightarrow \text{NotSoddisfacibile}(\phi)$$

7.1.2 Validità

Quello che vorremmo fare è partire dalla definizione di soddisfacibile, cioè esiste almeno un modello che soddisfi la formula $(\exists m : m \models \phi)$ e arrivare alla definizione di validità, cioè che ogni modello soddisfi la formula $(\forall m : m \models \phi)$.

Negando la definizione di soddisfacibile $\neg(\exists m : m \models \phi)$ otterremo $\forall m : m \not\models \phi$ che è quasi simile alla definizione di validità (ma è la definizione di insoddisfacibilità).

Utilizzeremo anche in questo caso l'algoritmo di NotSoddisfacibilità, che già sappiamo essere un algoritmo che data una formula ϕ verifica che non esiste un modello che la soddisfi, cioè che la renda vera ($\forall m : m \not\models \phi$). Ma passeremo la formula negata ($\neg\phi$). In altre parole, verificheremo che $\forall m : m \not\models (\neg\phi)$ e di conseguenza, se ogni modello non verificherà $\neg\phi$, allora verificherà ϕ. In termini matematici $\forall m : m \models \phi$ che è proprio la definizione di validità.

$$\text{Validità} \rightarrow \text{NotSoddisfacibile}(\neg\phi)$$

7.1.3 Conseguenza logica

Dato un insieme Γ diremo che ϕ è **conseguenza logica** di Γ con la seguente notazione: $\boldsymbol{\Gamma \models \phi}$.

7.1.3.1 Proprietà di Compattezza

Proprietà 7.1.1 [Compattezza]
Se una formula ϕ è una conseguenza logica di un insieme di formule Γ, allora esiste un sottoinsieme finito di formule di Γ di cui ϕ ne è conseguenza logica.

Questa proprietà permette di ridurre il problema della conseguenza logica alla soddisfacibilità:

$$\Gamma \models \phi \textbf{ se e solo se } \text{ esiste un modello } m \text{ tale che } m \models \phi$$

7.2 Teorema di Cook-Levin

Teorema 7.2.1

Decidere se $\phi \in PL$ è soddisfacibile, risulta essere un problema **NP-completo**.

Osservazione 7.2.1
I problemi NP-completi sono considerati difficili dal punto di vista della complessità, il che significa che sono difficili da risolvere in tempo polinomiale.

7.3 DNF e CNF per soddisfacibilità e validità

Ci sono formule la cui **forma** può rendere facile (in tempo polinomiale) la decisione della soddisfacibilità e validità.

Definizione 7.3.1 [DNF: Soddisfacibilità]

Le formule DNF hanno la seguente forma:

$$\phi = C_1 \vee C_2 \vee ... \vee C_n$$

$$\text{Dove } C_i = l_1^i \wedge l_2^i \wedge ... \wedge l_{ki}^i \text{ (cioè letterali della forma A o } \neg A)$$

Per essere **soddisfacibile** basta trovare almeno un C_i vero.

Per essere **non soddisfacibile** deve valere che $\forall C_i,\ C_i$ non è soddisfacibile.

Una formula che sicuramente non è soddisfacibile è $A \wedge \neg A$ (complementare), quindi possiamo limitarci a cercare se tutte le C_i presentano tale formula.

Diremo quindi che ϕ è soddisfacibile se nei C_i non ci sono $A \wedge \neg A$.
La ricerca di un complementare avviene in tempo polinomiale.

Definizione 7.3.2 [CNF: Validità]

Le formule CNF hanno la seguente forma:

$$\phi = C_1 \wedge C_2 \wedge ... \wedge C_n$$

$$\text{Dove } C_i = l_1^i \vee l_2^i \vee ... \vee l_{ki}^i \text{ (cioè letterali della forma A o } \neg A)$$

Per assicurare la **validità** quindi devo verificare che $\forall C_i$ siano tutti validi, quindi per poter rendere vera una C_i abbiamo bisogno che almeno un l_i^i sia vero.
Questo è garantito se in ogni C_i ci sia almeno un l_i^i che sia uguale ad A e un l_j^i che sia uguale a $\neg A$, così da garantire sicuramente che uno dei due sia vero e di conseguenza (essendo C_i una disgiunzione) C_i è vero.

Diremo che ϕ è **valida** se nei C_i ci sono A e $\neg A$.
Questa ricerca degli A e dei $\neg A$ è polinomiale.

7.4 Trasformazione di una formula

> "*Ma quindi se a partire da una qualsiasi formula, riuscissimo a trasformarla in una DNF e poi verificare la soddisfacibilità, questo non sarebbe un problema risolvibile in tempo polinomiale?*"

La risposta è **NO**, perchè andremmo contro il teorema di Cook-Levin, questo perchè la **trasformazione di una formula in una DNF non è un problema polinomiale**.

Proposizione 7.4.1

Prendere una formula CNF e trasformarla in DNF impiegherà tempo esponenziale.

Dimostrazione 7.4.1

Prendiamo per esempio una CNF:

$$(A \vee B) \wedge (D \vee E)$$

Iniziamo la trasformazione in DNF sfruttando:

$$\phi \wedge (\psi_1 \vee \psi_2) \equiv (\phi \wedge \psi_1) \vee (\phi \wedge \psi_2) \text{ [Regola della Distributività]}$$

Allora avremo che:

$$\overset{Distributivita'}{\Downarrow}$$

$$((A \vee B) \wedge D) \vee ((A \vee B) \wedge E)$$

$$\overset{Distributivita'}{\Downarrow}$$

$$((D \wedge A) \vee (D \wedge B)) \vee ((E \wedge A) \vee (E \wedge B))$$

Adesso prendiamo tutte le congiunzioni ($\wedge$) che dipendono in numero dal numero di proposizioni atomiche (k) che abbiamo nella formula.

Per ogni congiunzione C_i dobbiamo scegliere le coppie a partire da un insieme $1, 2, ...k$. Vediamo ogni congiunzione come una funzione $C_i : \{1, ..., k\} \to \{1, 2\}$ che fa una scelta. Il numero delle clausole $C_1, .., C_k$ sarà il numero delle funzioni di scelta e il numero di scelte possibili saranno 2^k.

Quindi la trasformazione per passare da CNF a DNF comporterà un'esplosione **esponenziale**, perchè la trasformazione sarà esponenziale.

7.5 Perchè utilizzare i Sistemi Assiomatici?

Dati $L, (M, \models)$ dove:

- L rappresenta la sintassi (o linguaggio)
- $(M, \models)$ rappresenta la semantica

Utilizzare l'approccio deduttivo di sostituzione non è sufficiente per generare, a partire da un insieme di tautologie, tutte le altre tautologie. In altre parole, questo approccio non riesce a garantire la proprietà di completezza.

7.6 Sistemi Assiomatici

Definizione 7.6.1

E' un insieme di regole e assiomi utilizzati per derivare proposizioni e teoremi attraverso un processo di **deduzione logica**, è composto da:

- **Insieme di assiomi** (finito) che devono essere delle tautologie
- **Regole di inferenza**: regole di generalizzazione che preservano delle proprietà

Un esempio di regola di inferenza è rappresentato dalla **sostituzione**, mediante la sostituzione possiamo prendere degli assiomi e instanziarli in altri modi. Parleremo proprio per questo motivo non di assiomi ma di **schemi** e inoltre considereremo che l'insieme degli assiomi sia **finito**.

Da un sistema assiomatico vogliamo:

- **Correttezza**: Generare solo tautologie a partire dal nostro sistema
- **Completezza**: Poter generare qualsiasi tautologia a partire dal nostro sistema

I sistemi di assiomi **naive** hanno un'unica regola d'inferenza chiamata "**Modus Ponens**".

7.6.1 Modus Ponens

$$\frac{\psi \qquad \psi \to \phi}{\phi} \quad MP$$

In poche parole significa che se ψ è vera e $\psi \to \phi$ è anch'essa vera, allora anche ϕ sarà vera (per la verità di $\to$).

7.7 Introduzione alla Deduzione

Definiremo con il simbolo $\vdash$ la **deducibilità**, mentre come già sappiamo con $\models$ indicheremo la conseguenza logica poiché avremo a che fare con Γ insieme di formule.

Arriveremo al seguente corollario:

$$\Gamma \models \phi \iff \Gamma \vdash \phi$$

In altre parole ci dice che una formula è una conseguenza logica di un insieme di formule se e solo se può essere dedotta da quell'insieme di formule attraverso un procedimento formale.

Osservazione 7.7.1
Possiamo osservare che:

1. L'implicazione $\Rightarrow$ ci garantisce la **completezza** del calcolo
2. L'implicazione $\Leftarrow$ ci garantisce la **correttezza** del calcolo

Capitolo 8

Lezione 8

Nella lezione precedente abbiamo espresso il concetto di **sistema assiomatico**, composto da un insieme di schemi di assiomi e da regole di inferenza. Tra le regole di inferenza abbiamo introdotto il **Modus Ponens** e la **Sostituzione**.

8.1 La sostituzione non è come sembra

La **sostituzione** non viene considerata una vera e propria regola di inferenza (nonostante garantisce correttezza e completezza) poiché non garantisce che se la formula originale ϕ è soddisfacibile, allora anche la trasformata ϕ_σ della formula è soddisfacibile.

Ma abbiamo visto che se ϕ è tautologia, allora anche ϕ_σ è tautologia. In altre parole, la sostituzione verifica la validità (tautologia) ma non la verità (soddisfacibilità).

$$Subst : \frac{\phi}{\phi_\sigma}$$

Esempio 8.1.1

Sia $\phi : AP \to W$, se ϕ è tautologia, allora ϕ_σ è tautologia

Ma se prendessimo $\phi = A \wedge B$ e usando $\{\phi : A \to A, B \to \neg A\}$

Allora $\phi_\sigma = A \wedge \neg A$, ottenendo allora che $m \models \phi$ e che $m \not\models \phi_\sigma$.

Cioè otterremo un modello che soddisfarà ϕ e che non soddisfarà ϕ_σ.

8.2 Logica Sintattica

Utilizziamo il Modus Ponens:

$$\frac{\psi \qquad \psi \to \phi}{\phi}$$

Il Modus Ponens soddisfa validità e verità.

In termini di Logica Semantica
Per assurdo, prendiamo un modello m che:

- $m \models \psi$
- $m \models (\psi \to \phi)$
  ma che
- $m \not\models \phi$

Possiamo osservare che è impossibile che $m \not\models \phi$ (m non soddisfa ϕ, cioè non rende vero) per i valori di verità dell'implicazione:

P	Q	$P \to Q$
V	V	V
V	F	F
F	V	V
F	F	V

Nella logica sintattica quindi ci baseremo sul concetto di **assiomi** e di come le dimostrazioni avvengono in maniera deduttiva.

8.3 Gli Assiomi Fondamentali

Di seguito gli schemi di assiomi fondamentali.

Definizione 8.3.1

- **A1:** $A \to (B \to A)$
- **A2:** $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$
- **A3:** $(A \wedge B) \to A$
- **A4:** $(A \wedge B) \to B$
- **A5:** $A \to (B \to (A \wedge B))$
- **A6:** $A \to (A \vee B)$
- **A7:** $B \to (A \vee B)$
- **A8:** $((A \to C) \to (B \to C) \to (A \vee B) \to C)$
- **A9:** $(A \to \bot) \to \neg A$
- **A10:** $(\neg A \to \bot) \to A$
- **A11:** $\bot \to A$
- **A12:** $A \to (\neg A \to \bot)$

Ricordiamo che tutti questi schemi sono tautologie (se ne facessimo le tabelle di verità).

8.4 Deduzione

Definizione 8.4.1

La deduzione è una relazione definita come:

$$\vdash \subseteq 2^W \times W$$

Possiamo osservare che la relazione è dello stesso tipo di quello di $\models$ conseguenza logica, cioè sempre una relazione $\subseteq 2^W \times W$.

La **deduzione** è una sequenza di formule

$$\Pi = \psi_1, \psi_2, ..., \psi_n$$

tale che $\forall i \in [1, n]$

1. $\psi_i \in \Gamma$ (cioè è una premessa)
   oppure
2. ψ_i è un assioma (anche istanziando un assioma tramite una sostituzione)
   oppure
3. ψ_i è ottenibile via *Modus Ponens* a partire da formule con indice $< i$

La cosa importante da ricordare è che ψ_i non può essere contemporaneamente 1 (premessa), 2 (assioma) e 3 (ottenibile via Modus Ponens).

Con la notazione $\Gamma \vdash \phi (= \psi_i)$ significa dire che ϕ è una deduzione da Γ. Inoltre, se $\Gamma = \emptyset$, allora avremo che $\vdash \phi$ è un teorema.

Esempio 8.4.1 Proviamo a dimostrare $\vdash \phi \to \phi$
Per dimostrarlo ci serviremo di **A1:** $A \to (B \to A)$ istanziandolo con:

$$\{\sigma : A \leftarrow \phi,\ B \leftarrow (\phi \to \phi)\}$$
$$\textbf{A1: } \phi \to ((\phi \to \phi) \to \phi)$$

E ci serviremo anche di **A2:** $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ istanziandolo con:

$$\{\sigma : A \leftarrow \phi,\ B \leftarrow (\phi \to \phi),\ C \leftarrow \phi\}$$
$$\textbf{A2: } \phi \to ((\phi \to \phi) \to \phi) \to ((\phi \to (\phi \to \phi)) \to (\phi \to \phi))$$

Possiamo notare che presi **A1** e **A2** questi due sono uguali:

$$\textbf{A1: } \phi \to ((\phi \to \phi) \to \phi)$$
$$\textbf{A2: } \phi \to ((\phi \to \phi) \to \phi) \to ((\phi \to (\phi \to \phi)) \to (\phi \to \phi))$$

Quindi possiamo applicare il *Modus Ponens* tra A1 e A2:

$$\frac{A1 : \phi \to ((\phi \to \phi) \to \phi \qquad A2 : \phi \to ((\phi \to \phi) \to \phi \to ((\phi \to (\phi \to \phi)) \to (\phi \to \phi))}{((\phi \to (\phi \to \phi)) \to (\phi \to \phi))}\ MP$$

Da $\bigstar : ((\phi \to (\phi \to \phi)) \to (\phi \to \phi))$ vogliamo adesso dimostrare $\phi \to \phi$.

Applichiamo nuovamente la sostituzione a **A1** nel seguente modo:

$$\{\sigma : A \leftarrow \phi,\ B \leftarrow \phi\}$$
$$\textbf{A1: } \phi \to (\phi \to \phi)$$

Costruito questo "nuovo" **A1** possiamo notare:

$$\textbf{A1: } \phi \to (\phi \to \phi)$$
$$\bigstar : ((\phi \to (\phi \to \phi)) \to (\phi \to \phi))$$

Quindi possiamo riapplicare il *Modus Ponens*:

$$\frac{\phi \to (\phi \to \phi) \qquad (\phi \to (\phi \to \phi)) \to (\phi \to \phi)}{(\phi \to \phi)}$$

Così abbiamo generato il nuovo assioma:

$$\textbf{A13: } (\phi \to \phi)$$

8.5 Proprietà di Monotonicità

Proprietà 8.5.1 [Monotonicità]

$$\text{Se } \Gamma \vdash \phi \text{ allora } \Gamma, \psi \vdash \phi \quad \forall \psi \in W$$

Questa proprietà esplicita che se deduco ϕ da Γ e aggiungo altre informazioni in Γ, riesco ancora a dedurre ϕ da Γ.

Dimostrazione 8.5.1

$$\text{Se } \Gamma \vdash \phi \text{ deve esistere una deduzione } \Pi \text{ di } \phi \text{ da } \Gamma$$

In altre parole, se deduco ϕ da Γ, allora esiste una deduzione (che è una sequenza di formule). Se aggiungessi in Γ altra informazione, comunque avrei ancora la sequenza di formule necessarie per dedurre ϕ.

8.6 Teorema di Monotonicità

Il seguente teorema generalizza la proprietà precedente.

Teorema 8.6.1 [Monotonicità]

Se $\Gamma \vdash \phi$ allora $\Gamma \cup \Gamma' \vdash \phi,\ \forall\ \Gamma' \subseteq W$

8.7 Teorema di Deduzione Sintattica

Teorema 8.7.1

Per ogni $\Gamma \subseteq W$ e prese $\psi, \phi \in W$
Se $\Gamma, \psi \vdash \phi \iff \Gamma \vdash (\psi \to \phi)$

Dimostrazione 8.7.1

Per induzione sulla lunghezza della deduzione Π di ϕ

Dim ⇒
Partiamo dalla premessa, cioè che esiste una deduzione di ϕ da Γ, ψ $(= \Gamma\ \cup\ \psi)$.

- **Caso Base** $(n = 1)$ [cioè la deduzione più semplice fatta da una sola formula]
  1. Con $\boldsymbol{\phi \in \Gamma \cup \{\psi\}}$
     - Se $\phi = \psi$ (cioè se $\phi \in \{\psi\}$)
       Per **A13** sappiamo che $\vdash \psi \to \psi$, allora

$$\begin{array}{c} \textbf{Monotonicità} \\ \Downarrow \\ \Gamma \vdash \psi \to \psi \\ \Downarrow \\ \Gamma \vdash \psi \to \phi \end{array}$$

     - Se $\phi \in \Gamma$ (è premessa)
       Sappiamo che $\Gamma \vdash \phi$ e che per **A1:** $A \to (B \to A)$ abbiamo che $\vdash \phi \to (\psi \to \phi)$

$$\begin{array}{c} \textbf{Monotonicità} \\ \Downarrow \\ \Gamma \vdash \phi \to (\psi \to \phi) \\ \textbf{Modus Ponens} \\ \Downarrow \\ \dfrac{\Gamma \vdash \phi \qquad \Gamma \vdash \phi \to (\psi \to \phi)}{(\psi \to \phi)} \\ \textbf{Monotonicità} \\ \Downarrow \\ \Gamma \vdash \psi \to \phi \end{array}$$

2. Con $\boldsymbol{\phi}$ **assioma**
   Se è un assioma allora vale $\vdash \phi$ e per **A1** avremo $\vdash \phi \to (\psi \to \phi)$
   Allora proprio come nel caso precedente otteniamo

$$\Gamma \vdash \psi \to \phi$$

- **Caso Induttivo** $(n > 1)$
  Quindi abbiamo la sequenza di formule come $\Pi = (\psi_1, \psi_2, ..., \psi_n)$

  Se ψ_n $(=\phi)$ è ottenuta per Modus Ponens da ψ_i e ψ_j con $i, j < n$

  $\psi_1, ..., \psi_i$ è una deduzione di ψ_i da $\Gamma \cup \{\psi\}$
  $\psi_1, ..., \psi_j$ è una deduzione di ψ_j da $\Gamma \cup \{\psi\}$

  Per ipotesi induttiva $\boldsymbol{\Gamma \vdash \psi \to \phi}$, otteniamo:

  $\bigstar : \Gamma \vdash \psi \to \psi_i$ (usando $\phi = \psi_i$)
  $\blacktriangle : \Gamma \vdash \psi \to \psi_j$ (usando $\phi = \psi_j$)

  Sappiamo che ψ_n è ottenuto tramite Modus Ponens da ψ_i e ψ_j, cioè:

$$\frac{\psi_i \qquad \psi_j}{\psi_n}$$

  ma questo significa che $\psi_j = \psi_i \to \psi_n$, allora avremo che:

$$\frac{\psi_i \qquad \psi_i \to \psi_n}{\psi_n}$$

  Andando a sostituire alla formula $\blacktriangle$ con $\psi_j = \psi_i \to \psi_n$ otteniamo:

$$\blacktriangle : \Gamma \vdash \psi \to (\psi_i \to \psi_n)$$

  Arrivati a questo prendiamo **A2:** $(A \to (B \to C)) \to ((A \to B) \to (A \to C))$ istanziandolo con:

$$\{\sigma : (A \leftarrow \psi, B \leftarrow \psi_i, C \leftarrow \psi_n)\}$$

  Ottenendo:

  $\boldsymbol{A2} : \Gamma \vdash (\psi \to (\psi_i \to \psi_n)) \to ((\psi \to \psi_i) \to (\psi \to \psi_n))$

  Con il **Modus Ponens** tra $\blacktriangle$ e $\boldsymbol{A2}$ otteniamo:

  $\nabla : ((\psi \to \psi_i) \to (\psi \to \psi_n))$

  E infine sempre con il **Modus Ponens** tra ∇ e $\bigstar$ otteniamo:

  $(\psi \to \psi_n)$

  Dove sostituendo ψ_n con ϕ otteniamo: $\psi \to \phi$

  Alla fine ottenendo: $\Gamma \vdash (\psi \to \phi)$

Dim $\Leftarrow$

Sfruttiamo la premessa che $\Gamma \vdash \psi \to \phi$

Cioè che esiste una deduzione $\pi = \psi_1, \psi_2, ..., (\psi \to \phi)$
Dove $\pi = \psi_1, \psi_2, ..., (\psi \to \phi)$ è una deduzione di $\psi \to \phi$ da Γ e scriveremo:

$$\Gamma \vdash \psi \to \phi$$

Ma per la proprietà di monotonicità, possiamo riscriverlo come:

$$\Gamma, \psi \vdash \psi \to \phi$$

Allora, arrivati a questo punto:

1. Inoltre se $\Gamma, \psi \vdash \psi$
2. e sapendo quindi che $\Gamma, \psi \vdash \psi \to \phi$
3. Applicando Modus Ponens tra 1 e 2

$$\frac{\Gamma, \psi \vdash \psi \quad \Gamma, \psi \vdash \psi \to \phi}{\phi} \; MP$$

Allora $\Gamma, \psi \vdash \phi$

Questo teorema è molto importante perchè ci permette di spostare una formula o proposizione atomica a nostro piacimento a sinistra o a destra del simbolo di deduzione:

$$\vdash A \to (B \to C) \equiv A \vdash B \to C$$

Capitolo 9

Lezione 9

In questa lezione introduciamo due regole fondamentali, **Regola del taglio** e **Reductio ad Absurdum**. Inoltre dimostreremo altri assiomi fondamentali.

9.1 Regola del Taglio (CR)

Definizione 9.1.1

Se $\Gamma, \psi \vdash \phi$ e $\Gamma' \vdash \psi$ allora $\Gamma \cup \Gamma' \vdash \phi$

Dimostrazione 9.1.1

1. $\Gamma, \psi \vdash \phi \overset{T.Deduzione}{\Rightarrow} \Gamma \vdash (\psi \to \phi) \overset{Monotonicita'}{\Rightarrow} \Gamma \cup \Gamma' \vdash (\psi \to \phi)$
2. $\Gamma' \vdash \psi \overset{Monotonicita'}{\Rightarrow} \Gamma \cup \Gamma' \vdash \psi$
3. Applicando Modus Ponens tra 1 e 2

$$\frac{\Gamma \cup \Gamma' \vdash \psi \qquad \Gamma \cup \Gamma' \vdash (\psi \to \phi)}{\phi} \; MP$$

Quindi abbiamo ottenuto $\Gamma \cup \Gamma' \vdash \phi$

9.2 Reductio ad Absurdum (RAA)

Definizione 9.2.1

Se $\Gamma, \psi \vdash \bot \Rightarrow \Gamma \vdash \neg\psi$

In altre parole, se assumendo qualcosa ottengo una contraddizione, allora una delle assunzioni è sbagliata.

Dimostrazione 9.2.1

Partendo da:

$$\Gamma, \psi \vdash \bot$$

Grazie al Teorema di Deduzione, ottengo:

$$\bigstar : \Gamma \vdash (\psi \to \bot)$$

Utilizzando **A9:** $\vdash (A \to \bot) \to \neg A$
e istanziandolo con $\sigma : (A \leftarrow \psi)$, otterremo:

$$\boldsymbol{A9\sigma} : \vdash (\psi \to \bot) \to \neg\psi$$

Applicando Modus Ponens tra $\bigstar$ e $\boldsymbol{A9\sigma}$

$$\frac{\Gamma \vdash (\psi \to \bot) \qquad (\psi \to \bot) \to \neg\psi)}{\neg\psi}$$

Quindi abbiamo dimostrato $\Gamma \vdash \neg\psi$

9.3 Dimostrazione di diversi assiomi

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

Dimostrazione 9.3.1
Come dimostrare $(\neg A \to \neg B) \to ((\neg A \to B) \to A)$?

Sappiamo che per ***A13*** : $\vdash A \to A$ e per Teorema di Deduzione otteniamo $A \vdash A$

Di conseguenza, sfruttando **A13** e applicando il Teorema di Deduzione possiamo ottenere le seguenti tre **assunzioni**:

1. $\neg A \to \neg B \ \vdash \ \neg A \to \neg B$
2. $\neg A \to B \ \vdash \ \neg A \to B$
3. $\neg A \ \vdash \ \neg A$

Applicando Modus Ponens tra 3 e 1 otteniamo:

$$\frac{\neg A \ \vdash \ \neg A \qquad \neg A \to \neg B \ \vdash \ \neg A \to \neg B}{\bigstar \ : \ \neg A, \neg A \to \neg B \ \vdash \ \neg B} \ MP$$

Applicando Modus Ponens tra 3 e 2 otteniamo:

$$\frac{\neg A \ \vdash \ \neg A \qquad \neg A \to B \ \vdash \ \neg A \to B}{\blacktriangle \ : \ \neg A, \neg A \to B \ \vdash \ B} \ MP$$

Sfruttando ***A12*** $: \vdash A \to (\neg A \to \bot)$ e istanziandolo con $\sigma = \{A \leftarrow B\}$,
otterremo: ***A12σ*** $: \vdash \ B \to (\neg B \to \bot)$

Applicando Modus Ponens tra ▲ e ***A12σ*** otterremo:

$$\frac{\neg A, \neg A \to B \ \vdash \ B \qquad \vdash \ B \to (\neg B \to \bot)}{\neg A, \neg A \to B \ \vdash \ (\neg B \to \bot)} \ MP$$

Applicando Modus Ponens tra ★ e **quanto ottenuto**, avremo:

$$\frac{\neg A, \neg A \to \neg B \qquad \neg A, \neg A \to B \ \vdash \ (\neg B \to \bot)}{\neg A, \neg A \to \neg B, \neg A \to B \ \vdash \ \bot}$$

Applicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

$$\neg A \to \neg B, \neg A \to B \ \vdash \ \neg A \to \bot$$

Sfruttando ***A10*** $: \vdash \ (\neg A \to \bot) \to A$ e applicando il Modus Ponens, otterremo:

$$\frac{\neg A \to \neg B, \neg A \to B \ \vdash \ \neg A \to \bot \qquad \vdash \ (\neg A \to \bot) \to A}{\neg A \to \neg B, \neg A \to B \ \vdash \ A} \ MP$$

Applicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

$$\neg A \to \neg B \ \vdash \ ((\neg A \to B) \to A)$$

Riapplicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

$$\vdash \ (\neg A \to \neg B) \ \to ((\neg A \to B) \to A)$$

□

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

Questa formula ci dice essenzialmente che l'ordine delle assunzioni (solo in questo caso specifico) è irrilevante.

Dimostrazione 9.3.2 [Dimostrazione]
Come dimostrare $A \to (B \to C) \iff ((A \wedge B) \to C)$?

Dim ⇒

Sfrutteremo:

- ***A3*** $: \vdash (A \wedge B) \to A \overset{T.Deduzione}{\Rightarrow} A \wedge B \vdash A$
- ***A4*** $: \vdash (A \wedge B) \to B \overset{T.Deduzione}{\Rightarrow} A \wedge B \vdash B$

Sappiamo che per ***A13*** $: \vdash A \to A$ e per Teorema di Deduzione otteniamo $A \vdash A$, quindi avremo che la nostra assunzione:

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

Applicando il Modus Ponens tra la nostra **assunzione** e **A3** dedotto, otteniamo:

$$\frac{A \to (B \to C) \vdash \ A \to (B \to C) \qquad A \wedge B \vdash A}{A \to (B \to C), A \wedge B \ \vdash (B \to C)} \ MP$$

Applicando il Modus Ponens tra **quanto ottenuto** e **A4** dedotto, otteniamo:

$$\frac{A \to (B \to C), A \wedge B \ \vdash (B \to C) \qquad \vdash (A \wedge B) \to B}{A \to (B \to C), A \wedge B \ \vdash C} \ MP$$

Applicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

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

Riapplicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

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

□

Dim ⇐

La nostra premessa è: $(A \wedge B) \to C$, ma per **A13** e **Teorema Di Deduzione** avremo la nostra **assunzione**:

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

Sfrutteremo ***A5*** $: \vdash A \to (B \to (A \wedge B))$ e assumeremo $A \ \vdash \ A$ e $B \ \vdash B$

Applicando Modus Ponens tra $\boldsymbol{A \vdash A}$ e $\boldsymbol{A5}$, otterremo:

$$\frac{A \vdash A \qquad \vdash A \to (B \to (A \wedge B))}{A \vdash (B \to (A \wedge B))} \; MP$$

Applicando Modus Ponens tra **quanto ottenuto** e $\boldsymbol{B \vdash B}$, otterremo:

$$\frac{A \vdash (B \to (A \wedge B)) \qquad B \vdash B}{A, B \vdash A \wedge B} \; MP$$

Applicando Modus Ponens tra la **nostra assunzione** e **quanto ottenuto**, avremo:

$$\frac{(A \wedge B) \to C \vdash (A \wedge B) \to C \qquad A, B \vdash A \wedge B}{A, B, (A \wedge B) \to C \vdash C} \; MP$$

Applicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

$$A, (A \wedge B) \to C \vdash B \to C$$

Riapplicando il Teorema di Deduzione su **quanto ottenuto**, avremo:

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

Riapplicando l'ultima volta il Teorema di Deduzione su **quanto ottenuto**, avremo:

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

□

$\vdash \neg\neg A \iff A$

- $\rightarrow$

  Prendiamo **A14**: $\vdash (\neg A \rightarrow \neg B) \rightarrow ((\neg A \rightarrow B) \rightarrow A)$

  $$\sigma : (A \leftarrow A, B \leftarrow \neg A)$$

  A14σ: $(\neg A \rightarrow \neg\neg A) \rightarrow ((\neg A \rightarrow \neg A) \rightarrow A)$

  Applicando la deduzione otteniamo:

  $$(\neg A \rightarrow \neg\neg A) \vdash ((\neg A \rightarrow \neg A) \rightarrow A)$$

  Ci costruiamo **ASS. 1**: $\vdash \neg A \rightarrow \neg A$

  Applichiamo il *Modus Ponens* tra **ASS. 1** e **A14** e otteniamo:

  $$(\neg A \rightarrow \neg\neg A) \vdash A$$

  Applichiamo la deduzione:

  ASS. 2: $\vdash (\neg A \rightarrow \neg\neg A) \rightarrow A$

  Adesso prendiamo **A1** e sostituiamo con $A \leftarrow \neg\neg A, B \leftarrow \neg A$:

  A1σ: $\vdash \neg\neg A \rightarrow (\neg A \rightarrow \neg\neg A)$

  Applichiamo la deduzione:

  $$\neg\neg A \vdash (\neg A \rightarrow \neg\neg A)$$

  Applichiamo il *Modus Ponens* tra **A1** e **ASS. 2** e otteniamo:

  $$\neg\neg A \vdash A$$

  Applicando una deduzione otteniamo:

  $$\neg\neg A \rightarrow A$$

- $\leftarrow$

  Dobbiamo dimostrare: $A \rightarrow \neg\neg A$

  Prendiamo **A14**: $(\neg A \rightarrow \neg B) \rightarrow ((\neg A \rightarrow B) \rightarrow A)$

  $\sigma : (A \leftarrow \neg\neg A, B \leftarrow A)$

  A14σ: $(\neg\neg\neg A \rightarrow \neg A) \rightarrow ((\neg\neg\neg A \rightarrow A) \rightarrow \neg\neg A)$

  Adesso prendiamo **ASS. 1**: $\vdash \neg\neg A \rightarrow A$ dimostrato prima

  $\sigma : (A \leftarrow \neg A)$

  ASS. 1 σ: $\vdash \neg\neg\neg A \rightarrow \neg A$

Applichiamo il *Modus Ponens* tra **ASS. 1** e **A14** ottenendo:

$$\textbf{ASS. 2}\text{: } \vdash (\neg\neg\neg A \to A) \to \neg\neg A)$$

Adesso prendiamo **A1** con questa sostituzione $\sigma : (A \leftarrow A, B \leftarrow \neg\neg\neg A)$

A1σ: $\vdash A \to (\neg\neg\neg A \to A)$

Applichiamo il teorema di deduzione:

$$A \vdash (\neg\neg\neg A \to A)$$

Adesso possiamo applicare il *Modus Ponens* tra **A1** e **ASS. 2** ottenendo:

$$A \vdash \neg\neg A$$

Applicando il teorema di deduzione per una volta otteniamo:

$$\vdash A \to \neg\neg\neg A$$

$\neg A \to (A \to \bot)$

Prendiamo **A12**: $A \to (\neg A \to \bot)$

$\sigma : (A \leftarrow \neg A)$

A12σ: $\neg A \to (\neg\neg A \to \bot) \Rightarrow \neg A \vdash (\neg\neg A \to \bot)$

Applichiamo *Modus Ponens* tra $A \vdash \neg\neg A$ e **A12**, ottenendo:

$$A, \neg A \vdash \bot$$

Applicando due volte il teorema di deduzione otteniamo:

$$\vdash \neg A \to (A \to \bot)$$

$(A \to B) \iff (\neg B \to \neg A)$

- $\leftarrow$

  Quello che dovremo dimostrare sarà:

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

  Prendiamo **A14** e sostituiamo $(A \leftarrow B, B \leftarrow A)$ ottenendo:

  $$\mathbf{A14}\sigma\text{: } \vdash (\neg B \to \neg A) \to ((\neg B \to A) \to B)$$

  Applicando il teorema di deduzione:

  $$(\neg B \to \neg A) \vdash ((\neg B \to A) \to B)$$

  Prendiamo **A1** e sostituiamo $(A \leftarrow A, B \leftarrow \neg B)$ ottenendo:

  $$\mathbf{A1}\sigma\text{: } \vdash A \to (\neg B \to A)$$

  Applicando il teorema di deduzione:

  $$A \vdash \neg B \to A$$

  Adesso applichiamo il *Modus Ponens* tra **A1** e **A14** ottenendo:

  $$(\neg B \to \neg A), A \vdash B$$

  Applicando due volte il teorema di deduzione otteniamo:

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

- $\rightarrow$

  Quello che dovremo dimostrare sarà:

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

  Prendiamo la formula dimostrata prima, $(\neg B \to \neg A) \to (A \to B)$ e applichiamo la sostituzione:

  σ: $(A \leftarrow \neg B, B \leftarrow \neg A)$

  Ottenendo:

  $$\mathbf{ASS\ 1.}\text{: } (\neg\neg A \to \neg\neg B) \to (\neg B \to \neg A)$$

  Prendiamo **A13** e sostituiamo $(A \leftarrow A \to B)$ ottenendo:

  $$\mathbf{A13}\sigma\text{: } A \to B \vdash A \to B$$

  Prendiamo anche **ASS. 2** che sarebbe $\neg\neg A \vdash A$ dimostrato in precedenza.

  Applichiamo il *Modus Ponens* tra **ASS. 2** e **A13** e otteniamo:

ASS. 3: $(\neg\neg A), (A \to B) \vdash B$

Adesso prendiamo $A \to \neg\neg A$ e sostituiamo $A \leftarrow B$ ottenendo:

ASS. 4: $B \to \neg\neg B$

Applichiamo il *Modus Ponens* tra **ASS. 3** e **ASS. 4** ottenendo:

ASS. 5: $(\neg\neg A), (A \to B) \vdash \neg\neg B$

Applichiamo il teorema di deduzione e otteniamo:

ASS. 5: $(A \to B) \vdash (\neg\neg A \to \neg\neg B)$

Infine applichiamo il *Modus Ponens* tra **ASS. 5** e **ASS. 1** ottenendo:

$$(A \to B) \vdash (\neg B \to \neg A)$$

Applicando una volta il teorema di deduzione otteniamo:

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

Capitolo 10

Lezione 10

10.1 PL: Completezza e Correttezza di $\Gamma \vdash \phi \iff \Gamma \models \phi$

Definizione 10.1.1

Per **correttezza** del calcolo intendiamo che se abbiamo un insieme Γ che deduce ϕ allora ϕ è conseguenza logica di Γ , mentre per **completezza** intendiamo che se ϕ è conseguenza logica di Γ allora Γ deduce ϕ.

In altre parole, quanto detto viene espresso come:

1. **Correttezza del sistema di calcolo** (o del calcolo):
$$\Gamma \vdash \phi \Rightarrow \Gamma \models \phi$$
2. **Completezza del calcolo**:
$$\Gamma \models \phi \Rightarrow \Gamma \vdash \phi$$

Dimostrazione 10.1.1
Dim ⇒ (Correttezza)
Come dimostrare che $\Gamma \vdash \phi \Rightarrow \Gamma \models \phi$?

In questo caso, quello che sarà sufficiente è che tutti gli assiomi precedenti siano validi e che il Modus Ponens sia valido e che generi ancora formule valide.
La dimostrazione si effettuerà per induzione (strutturale) sulla lunghezza della deduzione.

- **Caso Base (n = 1)** [cioè la deduzione più semplice, cioè fatta da una sola formula]
  La nostra premessa è che $\Gamma \vdash \phi$, allora esiste una deduzione di ϕ che indicheremo con $\Pi = \psi_0$ (con lunghezza 1). Ovviamente $\phi = \psi_0$.

  Inoltre, per definizione di deduzione, sappiamo che:
  1. ψ_0 o è premessa ($\psi_i \in \Gamma$)
  2. ψ_0 o è assioma (anche istanziato tramite una sostituzione)
  3. ψ_0 o è ottenuto tramite *Modus Ponens* da formule con indice $< i$

- Possiamo banalmente osservare che 3 è impossibile, perché c'è un un'unica formula, quindi ψ_0 non è ottenibile tramite Modus Ponens.
- Se ψ_0 è premessa, cioè che $\psi_0 \in \Gamma$.
  Di conseguenza, preso un modello $m \in M$, dire che $\Gamma \models \psi_0$ (cioè ψ_0 è conseguenza logica di Γ), significa che $m \models \psi_i \ \forall \psi_i \in \Gamma$. Ma visto che $\psi_0 \in \Gamma$, allora $m \models \psi_0$.
  Potremmo quindi scrivere che $\Gamma \models \psi_0$.
- Se ψ_0 è assioma, allora banalmente otteniamo $\models \psi_0$ e per monotonicità otteniamo $\Gamma \models \psi_0$.
  In altre parole, se $\forall m \in M,\ m \models \phi$ (in tutti i modelli possibili e immaginabili), allora anche per i modelli $m' : m' \models \Gamma(\forall\ \psi_i \in \Gamma,\ m' \models \psi_i)$, allora $m' \models \phi$ (cioè i modelli m' sono un sottoinsieme degli m).

- **Ipotesi Induttiva**
  Se $m \models \psi$ e $m \models \psi \to \phi$ ma $m \not\models \phi$, di conseguenza avremo che:
  - o $m \not\models \psi$
  - o $m \models \phi$

  Ma per la semantica dell'implicazione otterremo che per forza di cose $m \models \phi$. Applicato su tutti i modelli, ricaviamo proprio la definizione di conseguenza logica e quindi possiamo scrivere che: $\Gamma \models \phi$.

- **Caso Induttivo (n > 1)**
  La nostra premessa è che $\Gamma \vdash \phi$, allora esiste una deduzione di ϕ che indicheremo con $\Pi = \psi_0, ..., \psi_n$. Ovviamente $\phi = \psi_n$.

  Inoltre, per definizione di deduzione, sappiamo che:
  1. ψ_n o è premessa ($\psi_i \in \Gamma$)
  2. ψ_n o è assioma (anche istanziato tramite una sostituzione)
  3. ψ_n è ottenuto tramite *Modus Ponens* da formule con indice $< i$

  - Possiamo osservare che 1 e 2 si dimostrano come per il Caso Base
  - Se ϕ è ottenuto tramite Modus Ponens, allora significa che è ottenuto da una certa ψ_i e $\psi_j = \psi_i \to \phi$ con $i, j < n$.

  Allora sappiamo che esistono due deduzioni Π_i per ψ_i e Π_j per ψ_j, cioè tale per cui $\Gamma \vdash \psi_i$ e $\Gamma \vdash \psi_j$, così descritte $\Pi_i = \psi_1, ..., \psi_i$ e $\Pi_j = \psi_1, ..., \psi_j$.
  Possiamo osservare che queste due deduzioni hanno lunghezza minore di n e quindi applichiamo l'Ipotesi Induttiva. Ma per l'Ipotesi Induttiva che $\Gamma \models \psi_i$ e $\Gamma \models \psi_i \to \phi$. Se queste due cose sono vere, per la semantica dell'implicazione per forza di cose anche ϕ è vero, cioè $\Gamma \models \phi$.

Dim ⇐ (Completezza)
Come dimostrare che $\Gamma \models \phi \Rightarrow \Gamma \vdash \phi$?

Per effettuare questa dimostrazione utilizzeremo i seguenti concetti:

1. $\Gamma \models \phi \iff \Gamma \cup \{\neg\phi\}$ *insoddisfacibile* (per un teorema precedente Lezione 4)
2. Γ *è inconsistente* $\iff \Gamma \vdash \bot$

Dimostrazione sintetica per 2.
Sappiamo che:

$$\Gamma \text{ è inconsistente} \iff \Gamma \vdash \bot$$

Riscrivibile come:

$$\Gamma \vdash \bot \iff \Gamma \vdash \phi \quad \forall \phi \in W$$

Dim ⇒
Usiamo l'assioma $A : \vdash \bot \to A$ istanziandolo con ϕ, ottenendo:

$$\bot \vdash \phi \quad \forall \phi \in W$$

Allora applicando Modus Ponens, ottenendo:

$$\frac{\Gamma \vdash \bot \qquad \vdash \bot \to \phi}{\Gamma \vdash \phi} \; MP$$

Dim ⇐
Banale, perchè per la premessa so che da Γ posso dedurre qualsiasi formula $\phi \in W$, ma sapendo che $\bot$ è una formula, posso dedurre anch'esso. Quindi da $\Gamma \vdash \bot$.

Torniamo alla nostra dimostrazione principale:
Sappiamo per la DIM ⇒ (Correttezza) che $\Gamma \vdash \phi \Rightarrow \Gamma \models \phi$.

Allora per la **legge di contrapposizione** vale scrivere che:

$$\Gamma \models \phi \Rightarrow \Gamma \vdash \phi \equiv \Gamma \nvdash \phi \Rightarrow \Gamma \not\models \phi$$

1. Ma se $\Gamma \nvdash \phi$ allora significa sicuramente che $\Gamma \cup \{\neg\phi\}$ è **soddisfacibile**.

2. Mentre potremmo dire che $\Gamma \cup \{\neg\phi\}$ è **consistente**, questo perchè:

   Per assurdo diciamo che sia inconsistente, quindi $(\Gamma, \neg\phi) \vdash \bot$ per definizione.
   Quindi possiamo applicare la **RAA** e ottenere $\Gamma \vdash \phi$ ma questo è **impossibile** perchè abbiamo detto in precedenza proprio che $\Gamma \nvdash \phi$.

Quindi abbiamo che $\Gamma \cup \{\neg\phi\}$ è consistente e che $\Gamma \cup \{\neg\phi\}$ è soddisfacibile. Allora se riusciamo a dimostrare che se prendiamo un insieme Σ tale che Σ è consistente $\Rightarrow$ Σ è soddisfacibile, allora abbiamo dimostrato che $\Gamma \nvdash \phi \Rightarrow \Gamma \not\models \phi$.

In poche parole se dimostriamo che:

$$\Gamma \cup \{\neg\phi\} \textit{ è consistente} \Rightarrow \Gamma \cup \{\neg\phi\} \textit{ è soddisfacibile}$$

allora

$$\Gamma \nvdash \phi \Rightarrow \Gamma \not\models \phi$$

allora per contrapposizione

$$\Gamma \models \phi \Rightarrow \Gamma \vdash \phi$$

10.2 Introduzione al Teorema Completezza: $(consistenza \Rightarrow soddisfacibilita')$

Teorema 10.2.1 [Completezza]

Per ogni insieme $\Sigma \subset W$:

$$\Sigma \textit{ è consistente} \iff \Sigma \textit{è soddisfacibile}$$

Per dimostrare questo teorema ci serviremo prima del **Lemma di Lindenbaum**.

10.2.1 Lemma di Lindenbaum

Teorema 10.2.2 [Lemma di Lindenbaum]

Ogni insieme consistente può essere esteso ad un insieme **massimale consistente**.

Per massimale consistente intendiamo un insieme consistente che se venisse esteso ulteriormente perderebbe la proprietà di consistenza.

Dimostrazione 10.2.1
Questo teorema si dimostra per induzione, consideriamo una sequenza infinita di insiemi consistenti dove l'insieme successivo è un'estensione del precedente:

$$\Gamma_0 \subseteq \Gamma_1 \subseteq ... \subseteq \Gamma_n \subseteq ...$$

- **Caso Base**: Abbiamo che $\Gamma_0 = \Sigma$ che sarà ovviamente consistente perchè partiamo da un insieme base Γ_0 consistente
- **Caso Induttivo:** Sapendo che W é un insieme finitamente enumerabile, prendiamo una alla volta le formule, seguendo l'ordine dato dalla funzione di enumerazione scelta. In altre parole, prendiamo un'arbitraria enumerazione di formule di W come $\phi_1, \phi_2, ..., \phi_n$ e ad ogni passo $i+1$ considereremo la formula ϕ_i.

  Tutti gli insiemi Γ_i saranno costruiti con la seguente regola di costruzione.
  Al passo $i+1$ avremo che:

$$\Gamma_{i+1} = \begin{cases} \Gamma_i \cup \{\phi_i\} & \text{se } \Gamma_i \cup \{\phi_i\} \textit{ è consistente} \\ \Gamma_i & \textit{altrimenti} \end{cases}$$

  Dove ϕ_i è una semplice formula generica che decidiamo di aggiungere (o di scartare).

  Allora per Induzione possiamo dire che $\forall i\ \Gamma_i$ sarà consistente, cioè tutti gli insiemi così costruiti sono consistenti.

Alla fine costruiremo un unico Γ che sarà l'unione di tutti i Γ_i:

$$\Gamma = \bigcup_{i \geq 0} \Gamma_i$$

Adesso non ci resta che dimostrare se questo insieme è **consistente** (non è detto che se tutti i Γ_i sono consistenti, allora anche Γ lo è) e se è **massimale**.

1. Γ *è consistente?*

   Per assurdo diciamo che Γ *è inconsistente*, quindi per definizione:

   $$\Gamma \vdash \bot \Rightarrow \exists \textit{ deduzione } \Pi = \psi_1, \psi_2, ..., \psi_k \text{ dove } \psi_k = \bot$$

   Adesso prendiamo un enumerazione di formule di W e facciamo caso che esiste un ψ_j che non appartiene alla deduzione di Γ ed ha indice maggiore uguale del massimo indice delle formule di Π:

   $$\dots \quad \overset{\psi_4}{\bullet} \quad \overset{\psi_2}{\bullet} \quad \overset{\psi_k}{\bullet} \quad \overset{\psi_2}{\bullet} \quad \overset{\psi_j}{\bullet} \quad \dots$$

   Adesso considereremo per costruzione un insieme Γ_j consistente:

   $$\Gamma_j = (\Gamma_k \cup \psi_j) \textit{consistente}$$

   Avremo allora che $\Gamma_j \supseteq \Gamma$, cioè Γ_j includerà l'unione di tutti i Γ_i (e quindi tutte le formule contenute in essi). Inoltre, avevamo supposto che $\Gamma \vdash \bot$, quindi di conseguenza anche $\Gamma_j \vdash \bot$, ma questo è **assurdo**, perché avevamo supposto che Γ_j è consistente.
   L'osservazione importante da fare è che se aggiungo delle formule nella costruzione dei Γ_i con $i <= j$, tutte queste saranno presenti in Γ_j. Quindi se da $\Gamma \vdash \bot$ e $\Gamma \subseteq \Gamma_j$, allora Γ_j avrà tutte le formule aggiunte da tutti i $\Gamma_i \in \Gamma$ e di conseguenza avrò anche le formule che gli permetteranno di dedurre il bottom, cioè $\Gamma_j \vdash \bot$.

   □

2. Γ *è massimale?*

   Supponiamo che esista **per assurdo** un insieme Δ consistente tale che $\Gamma \subset \Delta$.

   Quindi vuol dire che esisterà una formula $\phi_m \in \Delta \setminus \Gamma$ (cioè una formula $\phi_m \in \Delta$ e $\phi_m \notin \Gamma$).

   Nella costruzione di :

   $$\Gamma_{m+1} = \begin{cases} \Gamma_m \cup \{\phi_m\} & \text{se } \Gamma_m \cup \{\phi_m\} \textit{ è consistente} \\ \Gamma_m & \textit{altrimenti} \end{cases}$$

   Nella costruzione dei Γ_i di Γ, al passo $m+1$ verrà considerata la formula ϕ_m che però sappiamo $\phi_m \notin \Gamma$ quindi non è aggiunta (ma è stata considerata), cioè:

   $$\Gamma_m \cup \{\phi_m\} \textit{ inconsistente } \Rightarrow \Gamma_m \cup \{\phi_m\} \vdash \bot$$

   Ma sappiamo che:
   - $\Gamma_m \subseteq \Gamma$
   - $\Gamma \subset \Delta$ consistente
   - $\phi_m \notin \Gamma$ ma che $\phi_m \in \Delta$

   Allora per forza di cose $\Gamma_m \cup \ \phi_m$ dovrà essere consistente e che $\phi_m \in \Gamma$.

Se ciò non fosse, Δ sarebbe inconsistente, perché includerebbe tutte le formule di Γ e quindi anche quelle che portano assieme a ϕ_m ($\in \Delta$) alla deduzione del bottom, cioè un qualcosa di inconsistente e quindi ottenendo un assurdo perchè Δ per ipotesi era consistente.

□

10.3 Proprietà insiemi massimali consistenti

Definizione 10.3.1
Sia Γ insieme massimale consistente, avremo che:

1. Esattamente uno tra ϕ e $\neg\phi$ sta in Γ
2. $\phi_1 \wedge \phi_2 \in \Gamma \iff \phi_1 \in \Gamma$ e $\phi_2 \in \Gamma$
3. $\phi_1 \vee \phi_2 \in \Gamma \iff \phi_1 \in \Gamma$ o $\phi_2 \in \Gamma$
4. $\phi_1 \to \phi_2 \in \Gamma \iff \phi_1 \notin \Gamma$ o $\phi_2 \in \Gamma$

Dimostrazione 10.3.1

1. Esattamente uno tra ϕ e $\neg\phi$ sta in Γ

   Per **Assurdo**, assumiamo che $\phi \notin \Gamma$ e che $\neg\phi \notin \Gamma$ e supponiamo che i è l'indice di ϕ e j è indice di $\neg\phi$ con $j > i$ nell'enumerazione delle formule.

   Quindi se $\phi \notin \Gamma$ allora questo significa che la formula non è stata aggiunta nella costruzione di Γ_i perchè portava inconsistenza, cioè: $\Gamma_i \cup \{\phi\} \vdash \bot$. Possiamo osservare che per monotonicità deriverò *bottom* per sempre da indice i in poi quando aggiungerò questa formula, quindi anche per $\Gamma_j \cup \{\phi\} \vdash \bot$, cioè sicuramente se non l'ha aggiunta un insieme con indice minore, sicuramente non sarà presente in un insime con indice maggiore ($j > i$). Ma stesso e identico ragionamento vale anche per $\Gamma_j \cup \{\neg\phi\} \vdash \bot$

   Possiamo riscrivere:

   $$\Gamma_j \cup \{\phi\} \vdash \bot = \Gamma_j, \phi \vdash \bot$$
   $$\Gamma_j \cup \{\neg\phi\} \vdash \bot = \Gamma_j, \neg\phi \vdash \bot$$

   Applicando il Teorema Di Deduzione, otterremo:

   $$\Gamma_j \vdash \phi \to \bot$$

   $$\Gamma_j \vdash \neg\phi \to \bot$$

   Utilizzando i seguenti assiomi:

   - $\vdash (\phi \to \bot) \to \neg\phi$
   - $\vdash (\neg\phi \to \bot) \to \phi$

   Applicando Modus Ponens, otterremo:

   $$\frac{\Gamma_j \vdash \phi \to \bot \qquad \vdash (\phi \to \bot) \to \neg\phi}{\Gamma_j \vdash \neg\phi} \; MP$$

   Ma ottenendo anche:

   $$\frac{\Gamma_j \vdash \neg\phi \to \bot \qquad \vdash (\neg\phi \to \bot) \to \phi}{\Gamma_j \vdash \phi} \; MP$$

Per **Modus Ponens** con l'assioma $\vdash A \to (B \to (A \wedge B))$ istanziato con $\{\phi \leftarrow A, \neg\phi \leftarrow B\}$, otterremo:

$$\frac{\Gamma_j \vdash \phi \qquad \vdash \phi \to ((\neg\phi) \to (\phi \wedge (\neg\phi)))}{\Gamma_j \vdash (\neg\phi) \to (\phi \wedge (\neg\phi))} \; MP$$

Applicando ancora Modus Ponens, otterremo:

$$\frac{\Gamma_j \vdash \neg\phi \qquad \vdash (\neg\phi) \to (\phi \wedge (\neg\phi))}{\Gamma_j \vdash \phi \wedge \neg\phi}$$

Infine, per Modus Ponens con l'assioma $\vdash (A \wedge \neg A) \to \bot$ istanziato con $\{\phi\}$, otterremo:

$$\frac{\Gamma_j \vdash \phi \wedge \neg\phi \qquad \phi \wedge \neg\phi \to \bot}{\Gamma_j \vdash \bot} \; MP$$

Ma questo è **Assurdo** perchè sappiamo che Γ_j è consistente, perché la nostra premessa era che se aggiungervamo una delle due formule diventava inconsistente.

2. • $\phi_1 \wedge \phi_2 \in \Gamma \iff \phi_1 \in \Gamma$ e $\phi_2 \in \Gamma$

Dim $\Rightarrow$
Supponiamo che $\phi_1 \wedge \phi_2 \in \Gamma$ ma $\phi_1 \notin \Gamma$ e preso un indice $i = \text{MAX}(index(\phi_1 \wedge \phi_2),\ index(\phi_1))$ tale per cui $\phi_1 \wedge \phi_2 \in \Gamma_{i+1}$. Questo ci dice che tutto quello che possiamo dedurre dall'insieme più piccolo, posso continuarlo a dedurlo anche in quello più grande.

Fin ora sappiamo che $\phi_1 \notin \Gamma$ allora di conseguenza non apparterrà neanche a un insieme più grande, cioè che $\phi_1 \notin \Gamma_{i+1}$

Di conseguenza, questo significa che nella costruzione di Γ_{i+1} avevamo che $\Gamma_i \cup \{\phi_1\} \vdash \bot$. Cioè se non appartiene è perchè se l'avessi aggiunta mi avrebbe portato un *bottom*.

Possiamo riscrivere queste quantità come:

$$\Gamma_i, \phi_1 \vdash \bot \overset{T.Deduzione}{\Rightarrow} \Gamma_i \vdash \phi_1 \to \bot$$

Per monotonicità avremo sicuramente che:

$$\Gamma_{i+1} \vdash \phi_1 \to \bot$$

Ma la nostra premessa era che $\phi_1 \wedge \phi_2 \in \Gamma_{i+1}$ e quindi questo significa che:

$$\Gamma_{i+1} \vdash (\phi_1 \wedge \phi_2)$$

Cioè se una formula è premessa, è banalmente deducibile.

Applicando **Modus Ponens** e sfruttando l'assioma $\vdash (A \wedge B) \to A$ istanziandolo con $\{A \leftarrow \phi_1, B \leftarrow \phi_2\}$, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_1 \wedge \phi_2 \qquad \vdash (\phi_1 \wedge \phi_2) \to \phi}{\Gamma_{i+1} \vdash \phi_1} \; MP$$

Riapplicando **Modus Ponens**, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_1 \qquad \Gamma_{i+1} \vdash \phi_1 \to \bot}{\Gamma_{i+1} \vdash \bot}$$

Ottenendo che $\Gamma_{i+1} \vdash \bot$, cioè che è inconsistente, ma questo è **assurdo**. É assurdo in quanto avevamo supposto che $\phi_1 \notin \Gamma$ poiché rendeva Γ inconsistente, di conseguenza se non veniva aggiunta in un insieme più piccolo, anche in un insieme più grande come Γ_{i+1} non è presente, di conseguenza non è possibile che Γ_{i+1} sia inconsistente perché l'avevamo supposto consistente.

Dim $\Leftarrow$
Sappiamo che $\phi_1 \in \Gamma$ e $\phi_2 \in \Gamma$ ma **per assurdo** supponiamo che $\phi_1 \wedge \phi_2 \notin \Gamma$ e prendiamo un indice $i = \text{MAX}(index(\phi_1 \wedge \phi_2), index(\phi_1), index(\phi_2))$

Osserviamo che:

- $\phi_1 \in \Gamma_{i+1}$
- $\phi_2 \in \Gamma_{i+1}$
- $\phi_1 \wedge \phi_2 \notin \Gamma_{i+1}$

Possiamo allora dire che:

- $\Gamma_{i+1} \vdash \phi_1$
- $\Gamma_{i+1} \vdash \phi_2$
- $\Gamma_{i+1} \vdash (\phi_1 \wedge \phi_2) \to \bot$

Applicando **Modus Ponens** con $\vdash A \to (B \to (A \wedge B))$ istanziandolo con $\{A \leftarrow \phi_1, B \leftarrow \phi_2\}$, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_1 \qquad \phi_1 \to (\phi_2 \to (\phi_1 \wedge \phi_2))}{\Gamma_{i+1} \vdash (\phi_2 \to (\phi_1 \wedge \phi_2))} \; MP$$

Applicando **Modus Ponens**, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_2 \qquad \Gamma_{i+1} \vdash (\phi_2 \to (\phi_1 \wedge \phi_2))}{\Gamma_{i+1} \vdash \phi_1 \wedge \phi_2} \; MP$$

Applicando **Modus Ponens**, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_1 \wedge \phi_2 \qquad \Gamma_{i+1} \vdash \; (\phi_1 \wedge \phi_2) \to \bot)}{\Gamma_{i+1} \vdash \bot} \; MP$$

Ma questo è **assurdo** perchè va contro la definizione di insieme massimale consistente. In altre parole, noi sapevamo che Γ_{i+1} era consistente, ma diventava inconistente quando aggiungevamo la formula $(\phi_1 \wedge \phi_2)$.

3. $\phi_1 \vee \phi_2 \in \Gamma \iff \phi_1 \in \Gamma$ o $\phi_2 \in \Gamma$

Sappiamo che $\phi_1 \vee \phi_2 \in \Gamma$ ma **per assurdo** supponiamo che $\phi_1 \notin \Gamma$ e $\phi_2 \notin \Gamma$. Allora preso un indice $i = \text{MAX}(index(\phi_1 \vee \phi_2),\ index(\phi_1))$ tale per cui $\phi_1 \vee \phi_2 \in \Gamma_{i+1}$.

Ma la nostra premessa era che $\phi_1 \vee \phi_2 \in \Gamma_{i+1}$ e quindi questo significa che:

- $\Gamma_{i+1} \vdash (\phi_1 \vee \phi_2)$

Cioè se una formula è premessa, è banalmente deducibile.

Ma sappiamo anche che $\phi_1 \notin \Gamma_{i+1}$ e $\phi_2 \notin \Gamma_{i+1}$ e questo quindi vuol dire che:

- $\Gamma_{i+1}, \phi_1 \vdash \bot \overset{T.Deduzione}{\Rightarrow} \Gamma_{i+1} \vdash \phi_1 \to \bot$
- $\Gamma_{i+1}, \phi_2 \vdash \bot \overset{T.Deduzione}{\Rightarrow} \Gamma_{i+1} \vdash \phi_2 \to \bot$

Applicando il **Modus Ponens** con l'assioma $\vdash (A \to C) \to ((B \to C) \to (A \vee B) \to C)$ istanziandolo con $\{A \leftarrow \phi_1, B \leftarrow \phi_2, C \leftarrow \bot\}$, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_1 \to \bot \qquad \vdash (\phi_1 \to \bot) \vdash ((\phi_2 \to \bot) \vdash (\phi_1 \vee \phi_2) \to \bot)}{\Gamma_{i+1} \vdash ((\phi_2 \to \bot) \vdash (\phi_1 \vee \phi_2) \to \bot)} \; MP$$

Applicando il **Modus Ponens**, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_2 \to \bot \qquad \Gamma_{i+1} \vdash ((\phi_2 \to \bot) \vdash (\phi_1 \vee \phi_2) \to \bot)}{\Gamma_{i+1} \vdash (\phi_1 \vee \phi_2) \to \bot} \; MP$$

Applicando il **Modus Ponens**, otterremo:

$$\frac{\Gamma_{i+1} \vdash \phi_1 \vee \phi_2 \qquad (\phi_1 \vee \phi_2) \to \bot}{\Gamma_{i+1} \vdash \bot} \; MP$$

Ma questo è **assurdo** perchè tutti i Γ_i sono consistenti.

10.4 Dimostrazione Teorema Completezza ($consistenza \Rightarrow soddisfacibilita'$)

Teorema 10.4.1 [Completezza]

Per ogni insieme $\Sigma \subset W$:

$$\Sigma \text{ è } consistente \iff \Sigma \text{è } soddisfacibile$$

Per dimostrare questo teorema ci serviremo del **Lemma di Lindenbaum**.

Dimostrazione 10.4.1

Dim $\Leftarrow$
Supponiamo che Σ è soddisfacibile $\overset{def}{\Rightarrow} \exists m \subseteq AP : m \models \phi \ \forall \phi \in \Sigma$

Quello che dimostreremo sarà un qualcosa di più **forte**:

$$\text{Se } \Sigma \vdash \psi \Leftarrow m \models \psi$$

Cioè dimostreremo che il modello m che soddisfa tutte le formule in Σ, soddisferà tutte le formule dedotte da Σ.

Questa dimostrazione la faremo sulla lunghezza della deduzione.

- **Caso Base:** $(l = 1)$
  Questo significa che $\phi \in \Sigma$ (premessa) o che ϕ è un assioma. Il primo caso è ovvio perchè se prima ho detto che m soddisfa tutte le formule di Σ allora soddisferà anche ϕ poichè $\phi \in \Sigma$. Il secondo caso ϕ è un assioma, quindi una **tautologia** e quindi sicuramente è soddisfatto da m. Per essere più specifici, se ϕ è assioma, allora banalmente otteniamo $m \models \phi$ perché tutti i modelli soddisfano gli assiomi.

- **Ipotesi Induttiva**
  Se $m \models \psi$ e $m \models \psi \to \phi$, per la semantica dell'implicazione otterremo che per forza di cose $m \models \phi$.

- **Caso Induttivo:** $(l > 1)$
  - Se $\phi \in \Sigma$ (premessa) o che ϕ è un assioma, otteniamo la stessa dimostrazione identica e precisa del Caso Base.
  - Se ϕ è ottenuta per Modus Ponens:

    $$\frac{\alpha \qquad \alpha \to \phi}{\phi} \quad MP(\text{preso un qualche } \alpha)$$

    Esistono quindi due deduzioni Π_i per α e Π_j per $\psi_j = \alpha \to \phi$, con lunghezza minore di l e quindi applichiamo l'Ipotesi Induttiva. Ma per l'Ipotesi Induttiva abbiamo che $m \models \alpha$ e $m \models \alpha \to \phi$, cioè m soddisfa ogni cosa dedotta con lunghezza $< n$. Ma se queste due cose sono vere, per la semantica dell'implicazione per forza di cose anche ϕ è vero e di conseguenza vale anche che $m \models \phi$

Possiamo concludere dicendo che Σ è **consistente**. Perché?
Perché se non lo fosse, allora per definizione di inconsistenza significherebbe che $\Sigma \vdash \bot$, ma questo è

impossibile poiché non esiste nessun modello $m \models \bot$.
Quindi abbiamo appena dimostrato che se Σ è consistente $\Leftarrow$ Σ soddisfacibile.

Dim $\Rightarrow$
Se Σ è un insieme consistente per il **Lemma di Lindembaum** $\exists\ \Gamma : \Sigma \subseteq \Gamma$ con Γ **massimale consistente**.

Prendiamo un modello definito come $m = \{A \in AP : A \in \Gamma\}$, quello che vogliamo dimostrare è:

$$\forall \phi \in W \ \ m \models \phi \iff \phi \in \Gamma$$

Se riuscissi a dimostrare questo, quello che otterrei è un modello m per Γ che soddisfa ogni sua formula. Ma possiamo osservare che $\Sigma \subseteq \Gamma$ in quanto tutte le formule di $\Sigma \in \Gamma$, ma allora m soddisfa anche tutte le formule di Σ, ottenendo quindi un modello m anche per Σ.

Per fare questa dimostrazione useremo un'induzione strutturale su ϕ.

- **Caso Base:** $\phi \in AP$
  Vuol dire che ϕ è una proposizione atomica, ma un modello m soddisfa un atomo quando quell'atomo è contenuto in m, ma m (per come è stato definito) contiene tutti e soli gli atomi contenuti in Γ quindi vale che $m \models \phi$ se e solo se $\phi \in \Gamma$.

- **Ipotesi Induttiva:** $\forall \phi \in W \ m \models \phi \iff \phi \in \Gamma$

- **Caso Induttivo:**
  Si dimostrerà per le proprietà di insieme massimale e consistente viste in precedenza.
  - Se $\phi = \neg\psi$ (per una qualche ψ più piccola/semplice di Γ, cioè di grado inferiore)
    Allora per **Ipotesi Induttiva** vale che $m \models \psi \iff \psi \in \Gamma$
    Questo possiamo riscriverlo come:

    $$m \not\models \psi \iff \psi \notin \Gamma$$

    Che è equivalente a:

    $$m \models \neg\psi \iff \neg\psi \in \Gamma$$

    Ma $\neg\psi = \phi$, ottenendo:

    $$m \models \phi \iff \phi \in \Gamma$$

  - Se $\phi = \phi_1 \wedge \phi_2$
    Allora per **Ipotesi Induttiva** vale che:

    $$\begin{cases} m \models \phi_1 \iff \phi_1 \in \Gamma \\ m \models \phi_2 \iff \phi_2 \in \Gamma \end{cases}$$

    Questo vale se e solo se $m \models \phi_1 \wedge \phi_2 \iff \phi_1 \wedge \phi_2 \in \Gamma$.

Allora finalmente possiamo osservare che preso un Δ arbitrario insieme di formule che:

$$\Delta \models \phi \Rightarrow \Delta \vdash \phi$$

Sappiamo che possiamo riscrivere questo come:

$$\Delta \cup \{\neg\phi\} \text{ è insoddisfacibile } \Rightarrow \Delta \cup \{\neg\phi\} \text{ è inconsistente}$$

Possiamo riscrivere il tutto per contrapposizione come:

$$\Delta \cup \{\neg\phi\} \text{ è consistente } \Rightarrow \Delta \cup \{\neg\phi\} \text{ è soddisfacibile}$$

Dove $\Sigma = \Delta \cup \{\neg\phi\}$ e ottenendo allora:

$$\Sigma \text{ consistente } \Rightarrow \Sigma \text{ soddisfacibile}$$

□

Capitolo 11

Lezione 11

11.1 Deduzione Naturale

Con la **deduzione naturale** abbiamo solo regole di inferenza senza assiomi. Inoltre non gode del problema della sostituzione, visto per i sistemi assiomatici, cioè che $\Gamma \vdash \phi \not\Rightarrow \Gamma \vdash \phi\sigma$.

Definizione 11.1.1
Le regole di inferenza sono :

- **Regole per $\wedge$**

$$\frac{A \qquad B}{A \wedge B} \wedge i \qquad \text{AND INTRODUCTION}$$

$$\frac{A \wedge B}{A} \wedge e_l \qquad \frac{A \wedge B}{B} \wedge e_r \qquad \text{AND ELIMINATION (LEFT/RIGHT)}$$

- **Regole per $\vee$**

$$\frac{A}{A \vee B} \vee i_l \qquad \frac{B}{A \vee B} \vee i_r \qquad \text{OR INTRODUCTION (LEFT/RIGHT)}$$

$$\frac{A \vee B \quad \begin{array}{c}[A] \\ \vdots \\ C\end{array} \quad \begin{array}{c}[B] \\ \vdots \\ C\end{array}}{C} \vee e \qquad \text{OR ELIMINATION}$$

- **Regole per $\rightarrow$**

$$\frac{A \qquad A \rightarrow B}{B} \rightarrow e \qquad \text{IMPL ELIMINATION}$$

$$\frac{\begin{array}{c}[A]\\ \vdots\\ B\end{array}}{A \to B} \to i \qquad \text{IMPL INTRODUCTION}$$

- **Regole per ¬**

$$\frac{A \qquad \neg A}{\bot} \neg e \qquad \text{NEG ELIMINATION}$$

$$\frac{\begin{array}{c}[A]\\ \vdots\\ \bot\end{array}}{\neg A} \neg i \qquad \text{NEG INTRODUCTION}$$

- **Regole per ⊥**

$$\frac{\bot}{\begin{array}{c}\vdots\\ A\end{array}} \bot e \qquad \text{BOT ELIMINATION}$$

$$\frac{\begin{array}{c}[\neg A]\\ \vdots\\ \bot\end{array}}{A} RAA \qquad \text{REDUCTION AD ABSURDER}$$

Capitolo 12

Lezione 12

Esercizio 12.0.1
Dimostrare in Deduzione Naturale.

$$\vdash \neg\neg A \to A$$

Assumiamo la premessa $\neg\neg A$ vera.

Quindi, **ragionando per assurdo**, prendendo ad esempio ciò con cui non può essere abbinato con $\neg\neg A$, come per esempio $\neg A$ e considerando le regole di inferenza precedenti avremo:

$$\frac{[\neg\neg A]_2 \quad [\neg A]_1}{\bot}\neg e$$

Il nostro obiettivo è scartare tutte le premesse rappresentanti le foglie di un albero rovesciato (albero di prova).

In questo caso dobbiamo scartare $[\neg\neg A]$ e $[\neg A]$.
Con RAA_1 scartiamo $[\neg A]_1$ in quanto dà come conclusione A dalla premessa $\bot$:

$$\frac{\bot}{A}RAA_1$$

Con $\to i_2$ scartiamo $[\neg\neg A]_2$ in quanto vale come conclusione $\neg\neg A \to A$ avendo così un assurdo rispetto alle premesse assunte per assurdo

$$\frac{A}{\neg\neg A \to A} \to i_2$$

La notazione di deduzione utilizzata fino ad ora è la seguente $\vdash$ dando per scontato la seguente notazione di Hilbert $\vdash_H$. Adesso per la deduzione naturale utilizziamo la seguente notazione $\vdash_{ND}$.

Esercizio 12.0.2
Dimostrare in Deduzione Naturale.

$$\vdash_{ND} (\neg A \to \neg B) \to ((\neg A \to B) \to A)$$

Partiamo dall'assunzione

$$\neg A \to \neg B$$

e poiché la premessa della conclusione è la seguente

$$\neg A \to B$$

Ragionando per assurdo, prendiamo come assunzione $\neg A$ da affiancare a $\neg A \to \neg B$ poiché se assumiamo vero anche $\neg A \to B$ allora A deve essere vero in quanto vale: $(\neg A \to B) \to A)$

Il nostro obiettivo è scartare, innanzitutto, $\neg A$

$$\frac{[\neg A \to \neg B]_3 \quad [\neg A]_1}{\neg B} \to e \quad \frac{[\neg A \to B]_2 \quad [\neg A]_1}{B} \to e$$

$$\frac{\neg B \quad B}{\bot} \neg e$$

$$\frac{\bot}{A} RAA_1$$

Con questo abbiamo scartato $\neg A$ poiché la conclusione è A.
Ma ci mancano ancora $[\neg A \to \neg B]_3$ e $[\neg A \to B]_2$. Iniziamo con lo scartare la 2 assunzione:

$$\frac{A}{(\neg A \to B) \to A} \to i_2$$

Adesso scartiamo la 3 assunzione:

$$\frac{(\neg A \to B) \to A}{(\neg A \to \neg B) \to (\neg A \to B) \to A} \to i_3$$

Con questo arriviamo ad un assurdo poiché abbiamo assunto $\neg A \to B$ vero e quindi poiché abbiamo concluso che vale anche $(\neg A \to B) \to A$ allora $\neg A$ è falso.

Esercizio 12.0.3
Dimostrare in Deduzione Naturale.

$$\vdash_{ND} (A \to B) \iff (\neg B \to \neg A) \quad \textbf{(Regola Di Contrapposizione)}$$

Dim $\Rightarrow$
Assumiamo $A \to B$ e *assumiamo* A

$$\frac{[A \to B]_3 \quad [A]_1}{B} \to e$$

Assumiamo $\neg B$

$$\frac{B \quad [\neg B]_2}{\bot} \neg e$$

Scartiamo $[A]_1$

$$\frac{\bot}{\neg A} \neg i_1$$

Scartiamo $[\neg B]_2$

$$\frac{\neg A}{\neg B \to \neg A} \to i_2$$

Scartiamo $[A \to B]_3$

$$\frac{\neg B \to \neg A}{(A \to B) \to (\neg B \to \neg A)} \to i_3$$

Dim $\Leftarrow$
Assumiamo $\neg B$ e *assumiamo* $\neg B \to \neg A$

$$\frac{[\neg B \to \neg A]_2 \quad [\neg B]_1}{\neg A} \to e$$

Assumiamo A

$$\frac{\neg A \quad [A]_3}{\bot} \neg e$$

Scartiamo $[\neg B]_1$

$$\frac{\bot}{B} \neg i_1$$

Scartiamo $[A]_3$

$$\frac{B}{A \to B} \to i_3$$

Scartiamo $[\neg B \to \neg A]_2$

$$\frac{A \to B}{(\neg B \to \neg A) \to (A \to B)} \to i_2$$

Esercizio 12.0.4
Dimostrare in Deduzione Naturale.

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

Assumiamo $A \to B$ (già ci viene detto) e *assumiamo* A

$$\frac{A \to B \quad [A]_1}{B} \to e$$

$$\frac{B}{C \vee B} \vee i_r$$

Assumiamo C

$$\frac{[C]_2}{C \vee B} \vee i_l$$

Per *scartare* 1 e 2, *assumiamo* $C \vee A$

$$\frac{C \vee B \quad C \vee B \quad [C \vee A]_3}{C \vee B} \vee e_{1,2}$$

Scartiamo $[C \vee A]_3$

$$\frac{C \vee B}{(C \vee A) \to (C \vee B)} \to i_3$$

Esercizio 12.0.5
Dimostrare in Deduzione Naturale.

$$\vdash \neg(A \wedge \neg A)$$

Essendo una tautologia **ragioniamo per assurdo**, quindi *assumiamo* come vero$A \wedge \neg A$

$$\frac{[A \wedge \neg A]_1}{A} \wedge e_l \quad \frac{[A \wedge \neg A]_1}{\neg A} \wedge e_r$$

$$\frac{A \quad \neg A}{\bot} \neg e$$

Scartiamo entrambe le assunzioni 1

$$\frac{\bot}{\neg(A \wedge \neg A)} \neg i_1$$

Con l'ultima conclusione abbiamo una contraddizione rispetto alle assunzioni $[A \wedge \neg A]_1$ fatte.
Quindi concludiamo che vale $\neg(A \wedge \neg A)$.

Quando non ci sono le implicazioni, si può dimostrare per assurdo.

Inoltre, tutte le foglie dell'albero (rovesciato) sono assunzioni. Se introduciamo nuove assunzioni, oltre a quelle già presenti, possiamo se scartarle o meno.

Esercizio 12.0.6
Dimostrare in Deduzione Naturale.

$$\vdash A \vee \neg A$$

Dimostriamo anche **questa per assurdo**

Assumiamo $\neg A$

$$\frac{[\neg A]_1}{A \vee \neg A} \vee i_r$$

Assumiamo $\neg(A \vee \neg A)$ per avere una contraddizione

$$\frac{A \vee \neg A \quad [\neg(A \vee \neg A)]_2}{\bot} \neg e$$

Scartiamo l'assunzione 1

$$\frac{\bot}{A} RAA_1$$

Possiamo osservare che fino a qua abbiamo 2 assuzioni

Riapplichiamo il ragionamento ripetendo gli stessi passaggi

$$\frac{A}{A \vee \neg A} \vee i_l$$

Riutilizziamo l'assunzione 2

$$\frac{A \vee \neg A \quad [\neg(A \vee \neg A)]_2}{\bot} \neg e$$

Scartiamo l'assunzione 2

$$\frac{\bot}{A \vee \neg A} RAA_2$$

Abbiamo, quindi, raggiunto una contraddizione rispetto $\neg(A \vee \neg A)$

La suddivisione fatta precedentemente, invece, è utile per mostrare come questa dimostrazione può essere ricondotta alla regola del taglio che consiste in breve di rappresentare la formula iniziale come insieme di sottoformule, ecco la rappresentazione applicando tale regola:

- $\neg A, \neg(A \vee \neg A) \vdash \bot$
- $\neg(A \vee \neg A) \vdash A$
- $A, \neg(A \vee \neg A) \vdash \bot$
- $\neg(A \vee \neg A) \vdash \bot$

Esercizio 12.0.7
Formalizzare in Deduzione Naturale.

1. Se Carlo è italiano e John non è spagnolo allora Rechel è tedesca
2. Se Lucy non è francese allora Carlo è italiano
3. Se Rechel è tedesca allora o Lucy è francese o John è spagnolo
4. John non è spagnolo

Noi vogliamo dimostrare che $\vdash$ Lucy è francese

C : Carlo è italiano
J : John è spagnolo
R : Rechel è tedesca
L : Lucy è francese

Quindi riconsideriamo le frasi precedenti nel seguente modo:

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

Per assurdo, *assumiamo* $\neg L$ e *assumiamo* L e J

$$\frac{[\neg L]_3 \quad [L]_1}{\bot} \to e \quad \frac{[J]_2 \quad \neg J}{\bot} \to e$$

Dobbiamo trovare un modo per scartare le assunzioni 1 e 2, per farlo effettuiamo i seguenti passi:

$$\frac{[\neg L]_3 \quad \neg L \to C}{C} \to e$$

$$\frac{C \quad \neg J}{C \wedge \neg J} \wedge i$$

$$\frac{C \wedge \neg J \quad (C \wedge \neg J) \to R}{R} \to e$$

$$\frac{R \quad R \to (L \vee J)}{L \vee J} \to e$$

Possiamo *scartare* le assunzioni 1 e 2

$$\frac{\bot \quad \bot \quad L \vee J}{\bot} \vee e_{1,2}$$

Scartiamo anche l'assunzione 3

$$\frac{\bot}{L} RAA_3$$

Esercizio 12.0.8

1. $p \to q, r \to s \vdash (p \vee r) \to (q \vee s)$
2. $p \iff q, q \iff r \vdash p \iff r$
3. $p \iff q \vdash \neg p \iff \neg q$
4. $(p \vee q) \iff p \vdash q \to p$
5. $p \iff \neg q, q \iff \neg r \vdash p \iff r$ $\star$
6. $\neg(p \to q) \vdash q \to r$
7. $\neg p \wedge \neg r \vdash \neg(p \vee r)$
8. $p \vdash (p \wedge q) \vee (p \wedge \neg q)$
9. $p \to q \vdash (p \wedge q) \iff p$
10. $(p \wedge q) \iff p \vdash p \to q$

Capitolo 13

Lezione 13

13.1 Teorema $\Gamma \vdash_H \phi \iff \Gamma \vdash_{ND} \phi$

Teorema 13.1.1

$$\Gamma \vdash_H \phi \iff \Gamma \vdash_{ND} \phi$$

Dove $\Gamma \vdash_H \phi$ è la deduzione con la notazione di Hilbert e si applica ad una sequenza di passi partendo dagli assiomi e applicando il Modus Ponens, mentre in $\Gamma \vdash_{ND} \phi$ si cerca di trovare un albero di prova che abbia come radice ϕ.

Dimostrazione 13.1.1
Dim ⇒ L'obiettivo è quindi quello di tradurre Hilbert in un albero di prova/deduzione naturale passo per passo.

La dimostrazione avviene per induzione sulla lunghezza della deduzione $\Pi = \phi_0, \phi_1, ..., \phi_n$.

- **Caso Base** $(n = 0)$
  Se $n = 0$ allora questo significa che c'è un'unica formula, cioè $\Pi = \phi_0$ con $\phi = \phi_0$.

  Di conseguenza avremo che:

  - o $\phi_0 \in \Gamma$ **premessa** (o assunzione)
  - o ϕ_0 è **assioma**

  Se $\phi_0 \in \Gamma$ **premessa** è banale, in quanto in ND avremo che $\phi_0 \in \Gamma$ (cioè è assunzione). Γ avrà già la prova perché banalmente possiamo dedurre $\Gamma \vdash_{ND} \phi_0$. In altre parole, avremo un albero di prova in ND formato solo dalla radice, cioè proprio Γ.

  Se ϕ_0 è **assioma**, dovremmo dimostrare che esiste un albero di prova in ND per tutti i 12 assiomi, ma ci limiteremo a farlo solo per alcuni.

A1) $\phi_0 = A \to (B \to A)$

$$\frac{\frac{[A]_1}{B \to A} \to_i}{A \to (B \to A)} \to_{i_1}$$

A2) $\phi_0 = (A \to (B \to C)) \to ((A \to B) \to (A \to C))$
Assumiamo $A, \quad A \to (B \to C), \quad A \to B$

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

Applichiamo nuovamente $\to_e$

$$\frac{B \to C \quad B}{C} \to_e$$

Siamo arrivati all'obiettivo che era C, che ora diventa la nostra radice e costruiamo l'albero di prova.

Scartiamo 1

$$\frac{C}{A \to C} \to_{i_1}$$

Scartiamo 2

$$\frac{A \to C}{(A \to B) \to (A \to C)} \to_{i_2}$$

Scartiamo 3

$$\frac{(A \to B) \to (A \to C)}{(A \to (B \to C)) \to ((A \to B) \to (A \to C))} \to_{i_3}$$

A8) $\phi_0 = ((A \to C) \to (B \to C) \to (A \vee B) \to C))$
Assumiamo sono $[A]_1 \quad [B]_2 \quad [A \to C]_5 \quad [B \to C]_4 \quad [A \vee B]_3$ mentre l'obiettivo è C.
Utilizziamo anche qui $\to_e$

$$\frac{[A] \quad [A \to C]}{C} \qquad \frac{[B] \quad [B \to C]}{C}$$

Scartiamo le assunzioni 1 e 2

$$\frac{C \quad C \quad A \vee B}{C} \vee_{e_{1,2}}$$

Abbiamo trovato il nostro obbiettivo C e quindi la radice ora dobbiamo costruire l'albero di prova, scartando $3, 4, 5$.

$$\frac{C}{(A \vee B) \to C} \to_{i_3}$$

$$\frac{(A \vee B) \to C}{(B \to C) \to ((A \vee B) \to C)} \to_{i_4}$$

$$\frac{(B \to C) \to ((A \vee B) \to C)}{(A \to C) \to ((B \to C) \to ((A \vee B) \to C))} \to_{i_5}$$

- **Caso Induttivo** $(n > 0)$
  Se $n > 0$ allora avremo che $\Pi = \phi_0, ..., \phi_n$

Se ϕ_n è ottenuta tramite Modus Ponens da due formule precedenti cioè:

$$\frac{\phi_i \quad \phi_j}{\phi_n}\text{MP} \qquad \text{con} \quad \phi_j = \phi_i \to \phi_n$$

Possiamo osservare che ϕ_i e ϕ_j compaiono prima di ϕ_n, allora avremo una prova in ND con lunghezza $< n$ per entrambe le deduzioni. Di conseguenza, sarà possibile applicare l'**Ipotesi Induttiva**, cioè per tutte le deduzioni con lunghezza $< n$ esiste una prova in ND.

Ottenendo così in Natural Deduction:

$$\frac{\begin{array}{cc}\Gamma & \Gamma \\ \vdots & \vdots \\ \Pi_1^{ND} = \phi_i & \Pi_2^{ND} = \phi_i \to \phi_n\end{array}}{\phi_n} \to_e$$

Dim ⇐

- **Caso base** ($h = 0$):
  Se $\phi \in \Gamma$ allora $\Gamma \vdash_H \phi$.
  Un albero di prova di altezza 0 è un'assuzione, cioè $\phi \in \Gamma$ e quindi banalmente per Hilbert avremo che $\Gamma \vdash_H \phi$

- **Caso induttivo** ($h > 0$):
  Per **Ipotesi Induttiva** sappiamo che è tutto deducibile fino alla $(n-1)$-esima regola in Hilbert. Dobbiamo quindi verificare che se applicando l'n-esima regola se è ancora deducibile in Hilbert.

  Presi due alberi di prova in ND con altezza $h < n$

  $$\frac{\begin{array}{cc}\Gamma & \Gamma \\ \vdots & \vdots \\ \phi_1 & \phi_2\end{array}}{\phi_n} \text{ per l'ultima regola di inferenza}$$

  Di conseguenza, sappiamo che per **Ipotesi Induttiva** esiste una prova in Hilbert, cioè:

  $$\Gamma \vdash_H \phi_1 \quad \text{e} \quad \Gamma \vdash_H \phi_2$$

  Dovremmo effettuare la dimostrazione per tutte le regole di ND, ma noi lo faremo solo con alcune.

$\neg_e$)

$$\dfrac{\begin{array}{cc}\Gamma & \Gamma \\ \vdots & \vdots \\ \phi_1 & \phi_2\end{array}}{\bot}\neg_e \quad \text{con } \phi_2 = \neg\phi_1$$

Abbiamo quindi che $\Gamma \vdash_H \phi_1$ e $\Gamma \vdash_H \neg\phi_1$

Utilizzando il seguente assioma: $\vdash_H A \to (\neg A \to \bot)$ sostituendo A con ϕ_1
abbiamo $\vdash_H \phi_1 \to (\neg\phi_1 \to \bot)$

Applichiamo MP

$$\dfrac{\Gamma \vdash_H \phi_1 \quad \vdash_H \phi_1 \to (\neg\phi_1 \to \bot)}{\Gamma \vdash_H \neg\phi_1 \to \bot} MP$$

$$\dfrac{\Gamma \vdash_H \neg\phi_1 \quad \Gamma \vdash_H \neg\phi_1 \to \bot}{\Gamma \vdash_H \bot} MP$$

$\vee_e$)

$$\dfrac{\begin{array}{ccc}\Gamma,\,[A] & \Gamma,\,[B] & \Gamma \\ \vdots & \vdots & \vdots \\ C & C & A \vee B\end{array}}{C}$$

Avremo quindi che per **Ipotesi Induttiva**: $\Gamma, A \vdash_H C \quad \Gamma, B \vdash_H C \quad \Gamma \vdash_H A \vee B$

Utilizzando il seguente assioma: $\vdash_H ((A \to C) \to (B \to C) \to (A \vee B) \to C))$
Utilizzando il Terorema Di Deduzione otteniamo

$$\dfrac{\Gamma, A \vdash_H C}{\Gamma \vdash_H A \to C} \qquad \dfrac{\Gamma, B \vdash_H C}{\Gamma \vdash_H B \to C}$$

Applichiamo MP

$$\dfrac{\vdash_H (A \to C) \to ((B \to C) \to ((A \vee B) \to C)) \qquad \Gamma \vdash_H A \to C}{\Gamma \vdash_H (B \to C) \to ((A \vee B) \to C)} MP$$

$$\dfrac{\Gamma \vdash_H (B \to C) \to ((A \vee B) \to C) \qquad \Gamma \vdash_H B \to C}{\Gamma \vdash_H (A \vee B) \to C} MP$$

$$\dfrac{\Gamma \vdash_H A \vee B \qquad \Gamma \vdash_H (A \vee B) \to C}{\Gamma \vdash_H C}$$

13.2 Resolution

Per le cose dette nelle lezioni precedenti, sappiamo che:

$$\Gamma \vdash \phi \Leftrightarrow \Gamma \cup \{\neg\phi\} \text{ è inconsistente}$$

Resolution è un sistema deduttivo introdotto nel 1965 e verifica se un insieme di prove è consistente o meno. L'idea alla base è quella di trasformare Σ in **Forma Clausale**, cioè un particolare rappresentazione della CNF (Conjuntive Normal Form) .

Considerando $\Gamma \cup \{\neg\phi\} = \Sigma$, quello che faremo è trasformare Σ in Forma Clausale.
In altre parole, questo significa:

$$\forall\phi \in \Sigma \text{ calcoliamo la CNF di } \phi \text{ e mettiamole tutte in } \wedge$$

13.2.1 Forma Clausale

1. Ogni clausola della CNF viene trasformata in un insieme di letterali.
   Per esempio, con i seguenti letterali $(l_1 \vee l_2 \vee l_3 \vee ... \vee l_n)$ avremo che $\{l_1, l_2, ..., l_n\}$ è la **clausola**. La trasformazione verrà fatta per tutte le clausole della CNF.

2. La rappresentazione in Forma Clausale consisterà in un insieme di **clausole**, così come segue:

$$\{\{l_1^1, l_2^1, ..., l_k^1\}, \{l_1^2, l_2^2, ..., l_k^2\}, ..., \{l_1^n, l_2^n, ..., l_k^n\}\}$$

Dobbiamo interpretare il livello interno come disgiunzioni $(\vee)$, mentre quello esterno congiunzioni $(\wedge)$, cioè:

$$\{\{l_1^1 \vee l_2^1 \vee ... \vee l_k^1\} \wedge \{l_1^2 \vee l_2^2 \vee ... \vee l_k^2\} \wedge ... \wedge \{l_1^n \vee l_2^n \vee ... \vee l_k^n\}\}$$

Esempio 13.2.1
Trasformandola in CNF la seguente formula:
$\phi = (p \to (q \wedge r)) \wedge (q \vee (q \to p)) \wedge ((\neg p \to q) \vee (\neg q \to r))$

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

Otteniamo:

$$\phi'_{CNF} = ((\neg p \vee q) \wedge (\neg q \vee r)) \wedge (q \vee \neg q \vee p) \wedge (p \vee q \vee q \vee r)$$

Traformandolo in Forma Clausale:

$$\phi_{CL} = \{\{\neg p, q\}, \{\neg p, r\}, \{q, \neg q, p\}, \{p, q, r\}\}$$

Nell'ultima clausola si rimuove una q in quanto è inutile avere delle ripetizioni nelle clausole.

13.3 Proprietà delle clausole

13.3.1 Clausola banale

La **Clausola Banale** è la clausola contenente due letterali opposti:

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

All'interno della clausola abbiamo disgiunzioni ($\vee$) e per questo motivo $p \vee \neg p = \top = \text{true}$. Abbiamo quindi la clausola che diventa $\{\top, q, r\}$ ma anche $\top \vee q = \top$ e $\top \vee r = \top$ quindi la clausola sarà $\{\top\}$.

Osservazione 13.3.1
Se in **Forma Clausale** c'è una clausola banale:

$$\{\{l_1^1 \vee l_2^1 \vee ... \vee l_k^1\} \wedge \{l_1^2 \vee l_2^2 \vee ... \vee l_k^2\} \wedge ... \wedge \{l_1^n \vee l_2^n \vee ... \vee l_k^n\}\} =$$

$$\{C_1 \wedge C_2 \wedge ... \wedge C_n\}$$

Se C_2 è banale, allora $C_2 = \{\top\}$, allora può essere anche rimossa poiché $C_2 = \{\top\} \wedge C_i = C_i \quad \forall i \in 1, ..., n$. In altre parole, qualsiasi cosa in AND col True è uguale a quella qualsiasi cosa senza il True, allora $\top$ viene considerato "elemento neutro".

13.3.2 Clausola vuota

La **Clausola Vuota** $\{\}$ rappresenta una clausola non contenente nessun letterale ed è identificata anche con il simbolo $\square$.

Definizione 13.3.1
Sapendo che una clausola $\{l_1 \vee ... \vee l_n\}$ è **soddisfacibile** se almeno un $l_i = \text{True}$

Allora di conseguenza possiamo affermare che la clausola vuota $\square$ **non è soddisfacibile**.

Definizione 13.3.2
Una forma clausale $\{C_1 \wedge C_2 \wedge ... \wedge C_n\}$ **è soddisfacibile** se ogni clausola $C_i \quad \forall i \in \{1, ..., n\}$ è soddisfacibile.

Allora di conseguenza possiamo affermare che se in una **forma clausale** c'è una **clausola vuota**, allora possiamo affermare che la forma causale è **insoddisfacibile**:

$$\{\{...\}, \square, ..., \{...\}\} \text{ è insoddisfacibile}$$

13.4 Regola di risoluzione

Il metodo **Resolution** utilizza una **regola di risoluzione**.

Se prese due clausole C_1 e C_2 di Σ tale che $l \in C_1$ e $\neg l \in C_2$, cioè che C_1 e C_2 contengono il complemento di un letterale. Allora si costruisce una **nuova clausola** chiamata **Risolvente**:

$$\text{Risolvente}(C_1, C_2) = C_1 \setminus \{l\} \cup C_2 \setminus \{\neg l\}$$

La **regola d'inferenza** è:

$$\frac{C_1 \quad C_2}{\text{Risolvente}(C_1, C_2)} \quad Res$$

Spesso anche indicata con:

$$\frac{C_1 \quad C_2}{\text{R}} \quad Res$$

L'idea è partire da Σ e applicare ripetutamente la regola d'inferenza a coppie di clausole di Σ, generando quindi un risolvente (cioè una nuova clausola) che viene aggiunta in Σ. Possiamo vedere i risolventi come le formule dedotte attraverso la regola d'inferenza. Una volta che due clausole sono state risolte, cioè selezionate dalla regola, quelle due clausole non verranno riselezionate assieme ma potrebbero essere riselezionate con clausole diverse.

Esempio 13.4.1

$$\phi_{CL} = \{\{p, q\}, ..., \{\neg p, r\}\}$$

Sia $C_1 = \{p, q\}$ e $C_2 = \{\neg p, r\}$, il risolvente sarà il seguente:

$$Res(C_1, C_2) = \{q\} \cup \{r\} = \{q, r\}$$

13.4.1 Prioprietà della regola di risoluzione

Proposizione 13.4.1
Siano C_1 e C_2 clausole $\in \Sigma$,

$$C_1, C_2 \text{ sono soddisfacibili insieme } \iff R \text{ (risolvente) è soddisfacibile (e tutto ciò che genera)}$$

Dire che C_1 e C_2 sono soddisfacibili insieme significa dire che sono due clausole soddisfatte per uno stesso assegnamento α.

Dimostrazione 13.4.1
Dim ⇒
Dalla premessa sappiamo che C_1, C_2 sono soddisfacibili da $\alpha : AP(C_1 \cup C_2) \to \{0, 1\}$.

Supponendo $l \in C_1$ e $\neg l \in C_2$,
se $\alpha(l) = val^\alpha(l) = 1$ e $\alpha(\neg l) = val^\alpha(\neg l) = 0$

Ma dalla premessa sappiamo che sia C_1 che C_2 sono soddisfacibli,
allora di conseguenza $\exists l' \neq \neg l \in C_2$ t.c. $val^\alpha(l') = 1$.
Questo in altre parole significa che in C_2 deve esistere un letterale che la soddisfi.

Per *Res* abbiamo che

$$\frac{C_1 \quad C_2}{R} \quad Res$$

con $R = C_1 \setminus \{l\} \cup C_2 \setminus \{\neg l\}$.

Allora possiamo osservare che $l' \in C_2 \setminus \{\neg l\} \subseteq R$ e di conseguenza $\rightarrow val^\alpha(R) = 1$ e quindi R è soddisfacibile. In altre parole, possiamo osservare che in R risolvente ci sarà l' che lo soddisferà. (Stessa cosa varrebbe se si scegliesse $val^\alpha(l) = 0$ e $val^\alpha(\neg l) = 1$)

Dim $\Leftarrow$
Se R è soddisfacibile allora $\exists \alpha : AP(R) \rightarrow \{0,1\}$ e $Val^\alpha(R) = 1 \Rightarrow \exists l' \in R$ t.c. $val^\alpha(l') = 1$
In altre parole, sapendo che R è soddisfacibile, allora per forza di cose esisterà un letterale l' che avrà valore true.

Supponendo $l \in C_1$ e $\neg l \in C_2$ (o viceversa),
il nostro obiettivo è trovare un assegnamento α' in grado di soddisfare sia C_1 che C_2.

1. Se $l' \in C_1$ e $l' \in C_2$
   costruiremo un assegnamento α'
   $\alpha' : AP(C_1 \cup C_2) \rightarrow \{0,1\}$ arbitrario tale che $\alpha'(p) = \alpha(p), \forall p \in AP(R)$

2. Se $l' \in C_1$ e $l' \notin C_2$ costruiremo un assegnamento α'
   $$\alpha' : \begin{cases} \alpha(p) & \text{se } p \neq l \\ 0 & \text{se } p = l \end{cases}$$
   Sapendo che $l' \in C_1$ e sapendo che $val^\alpha(l') = 1$ allora C_1 è soddisfatta in quanto esiste un letterale (proprio l' che la soddisfa). Inoltre, sapendo che $l \in C_1$ e $\neg l \in C_2$, per soddisfare anche C_2, metteremo $l = 0$ cossicchè $\neg l = 1$ e di conseguenza anche C_2 è soddisfatta.

3. Se $l' \notin C_1$ e $l' \in C_2$ costruiremo un assegnamento α'
   $$\alpha' : \begin{cases} \alpha(p) & \text{se } p \neq l \\ 1 & \text{se } p = l \end{cases}$$
   Sapendo che $l' \in C_2$ e sapendo che $val^\alpha(l') = 1$ allora C_2 è soddisfatta in quanto esiste un letterale (proprio l' che la soddisfa). Inoltre, sapendo che $l \in C_1$ e $\neg l \in C_2$, per soddisfare anche C_1, metteremo $\neg l = 0$ cossicchè $l = 1$ e di conseguenza anche C_1 è soddisfatta.

Capitolo 14

Lezione 14

Con la regola di Resolution possiamo dedurre formule dedotte da altre regole d'inferenza come il Modus Ponens del sistema assiomatico.

Esempio 14.0.1
Se utilizzando Modus Ponens avremo:

$$p, p \to q \vDash q$$

Allora applicando il Teorema Di Deduzione avremo una contraddizione:

$$p, p \to q, \neg q \vDash \bot$$

Rendendola in forma CNF:

$$(p) \lor (p \to q) \lor (\neg q) \vdash \bot$$

Effettuiamo una trasformazione in forma clausale (con $p \to q = \neg p \lor q$), otterremo:

$$\{\{p\}\{\neg p, q\}\{\neg q\}\}$$

Quello che cercheremo di fare è applicare la risoluzione in modo da raggiungere l'insieme vuoto $\square$.

- clash tra $\{p\}$ e $\{\neg p, q\}$ ricavando $\{q\}$
- clash tra $\{q\}$ e $\{\neg q\}$ ricavando $\square$

Esempio 14.0.2

$$p \to q, q \to r \vDash p \to r$$

Rendendola in forma CNF:

$$(\neg p \lor q) \land (\neg q \lor r) \land \neg(\neg p \lor r) \vDash \bot$$

Scrivendola in un altro modo avremo:

$$(\neg p \lor q) \land (\neg q \lor r) \land (p \land \neg r) \vDash \bot$$

Effettuiamo una trasformazione in forma clausale ottenendo:

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

Applichiamo la risoluzione:

- clash tra $\{\neg p, q\}$ e $\{p\}$ ricavando $\{q\}$
- clash tra $\{q\}$ ricavato e $\{\neg q, r\}$ ricavando $\{r\}$
- clash tra $\{r\}$ e $\{\neg r\}$ ricavando $\square$

14.1 Algoritmo di Resolution

L'algoritmo di Resolution viene utilizzato per rispondere alla seguente domanda:

$$\Gamma \vDash \phi \iff \Sigma = \Gamma \cup \{\neg\phi\} \quad \Sigma \text{ è soddisfacibile?}$$

Prima di scrivere l'algoritmo, possiamo dettagliarne i parametri:

- $S = \Sigma$: rappresente l'insieme delle clausole iniziali
- $E = \varnothing$: rappresenta l'insieme per accantonare le **coppie** di clausole già selezionate per risoluzione. Ovviamente, una clausola può essere riutilizzata con un'altra clausola.
- Se ci sono più litterali in **clash** tra due clausole, ne selezioneremo uno.

Di seguito l'algoritmo per risolvere tale problema:

Input:
Output: True o False
Function:
    E = $\varnothing$, S = Σ
    repeat
        Scegli $C_1 \in S$ e $C_2 \in S$ con un letterale complementare (c_1 e c_2 hanno clash su l letterale)
        $E = E \cup \{(C_1, C_2)\}$
        $R = Resolution(C_1, C_2)$
        if $R \neq \square$ **then**
            $S = S \cup \{R\}$;
        end
    until $R = \square$ *or* $E = S \times S$;
    if $R = \square$ **then**
        return *False*
    end
    else
        return *True*
    end

Algorithm 1:

Possiamo osservare che l'algoritmo terminerà se:

- $R = \square$, cioè se abbiamo ottenuto la clausola vuota, quindi di conseguenza Σ non è soddisfacibile e quindi restituiremo **false**

- $E = S \times S$, cioè se abbiamo finito le coppie (e tutte le loro permutazioni) da poter selezionare e quindi restituiremo **true**

Per capire se l'algoritmo termina, possiamo osservare che:

- Σ è un insieme finito
- Nel caso peggiore: se S aumentasse ad ogni iterazione, anche E aumenterebbe (di conseguenza) aggiungendo le coppie prese in considerazione
- Ma R decresce

Questo è sufficiente per garantire che l'algoritmo **termina**.

Esempio 14.1.1
Se ci sono più litterali in **clash** tra due clausole, ne selezioneremo uno.

Consideriamo la seguente formula: $\{p, q\}\{\neg p, \neg q\}$

Rendendola in formato CNF:

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

Utilizzeremo un assegnamento α per renderla soddisfacibile:

$$val^{\alpha}(p) = True$$

$$val^{\alpha}(p) = False$$

Per Resolution, se raggiungiamo $\square$ la formula è insoddisfacibile.
Possiamo osservare che:

- clash tra $\{p, q\}$ e $\{\neg p, \neg q\}$ (togliendo p) ricavando $\{q, \neg q\}$
- clash tra $\{p, q\}$ e $\{\neg p, \neg q\}$ (togliendo q) ricavando $\{p, \neg p\}$

In entrambi i casi raggiungiamo una clausola banale (che è sempre true) e non essendoci altri clash (in quanto non ci sono altre clausole), possiamo concludere che la formula è soddisfacibile in quanto non si raggiunge $\square$.

14.2 Esempio classico di deduzione applicando Resolution

Esempio 14.2.1

- C = Carlo è italiano
- L = Lucy è francese
- J = John è spagnolo
- R = Rachel è tedesca

$$\left.\begin{array}{c}(C \wedge \neg J) \rightarrow R \\ R \rightarrow (L \vee J) \\ \neg L \rightarrow C \\ \neg J\end{array}\right\} \models L$$

Utilizziamo il teorema di deduzione per cambiare la formula e dimostrare che sia inconsistente.

$$\left.\begin{array}{c}(C \wedge \neg J) \rightarrow R \\ R \rightarrow (L \vee J) \\ \neg L \rightarrow C \\ \neg J \\ \neg L\end{array}\right\} \text{è inconsistente}$$

In Forma Clausale: $\{\{\neg L\}\{\neg J\}\{L, C\}\{\neg R, L, J\}\{\neg C, J, R\}\}$

Applichiamo Resolution:

- clash tra $\{\neg L\}$ e $\{L, C\}$ ricavando $\{C\}$
- clash tra $\{C\}$ ricavato e $\{\neg C, J, R\}$ ricavando $\{J, R\}$
- clash tra $\{J, R\}$ ricavato e $\{\neg J\}$ ricavando $\{R\}$
- clash tra $\{R\}$ ricavato e $\{\neg R, L, J\}$ ricavando $\{L, J\}$
- clash tra $\{L, J\}$ ricavato e $\{\neg J\}$ ricavando $\{L\}$
- clash tra $\{L\}$ ricavato e $\{\neg L\}$ ricavando $\square$

Quindi è inconsistente come volevamo.

Possiamo riescrivere il tutto ad albero:

$$\frac{\{\neg L\} \quad \{L, C\}}{\{C\}}$$

$$\frac{\{C\} \quad \{\neg C, J, R\}}{\{J, R\}}$$

$$\frac{\{J, R\} \quad \{\neg J\}}{\{R\}}$$

$$\frac{\{R\} \quad \{\neg R, L, J\}}{\{L, J\}}$$

$$\frac{\{L, J\} \quad \{\neg L\}}{\{J\}}$$

$$\frac{\{J\} \quad \{\neg J\}}{\square}$$

Esempio 14.2.2
Adesso consideriamo questa formula e vediamo se è inconsistente:

$$A \vee B, (A \wedge B) \rightarrow C, C \rightarrow D, A \vDash D$$

Rendiamola in forma clausale dopo aver applicato il teorema di deduzione:

$$\{\{A, B\}\{\neg A, \neg B, C\}\{\neg C, D\}\{A\}\{\neg D\}\}$$

Ora vediamo se è inconsistente:

$$\frac{\{\neg C, D\} \quad \{\neg D\}}{\{\neg C\}}$$

$$\frac{\{\neg C\} \quad \{\neg A, \neg B, C\}}{\{\neg A, \neg B\}}$$

$$\frac{\{\neg A, \neg B\} \quad \{A\}}{\{\neg B\}}(1)$$

$$\frac{\{\neg B\} \quad \{A, B\}}{\{A\}}$$

$$\frac{\{A\} \quad \{\neg A, \neg B, C\}}{\{\neg B, C\}}$$

$$\frac{\{\neg B, C\} \quad \{\neg C, D\}}{\{D, \neg B\}}(2)$$

$$\frac{\{D, \neg B\} \quad \{\neg D\}}{\{\neg B\}}$$

$$\frac{\{\neg B\} \quad \{A, B\}}{\{A\}}$$

Ma A l'abbiamo già trovato precedentemente quindi per adesso l'inconsistenza non è stata dimostrata in quanto non siamo arrivati a $\square$. Ciò però non vuol dire che non sia inconsistente in quanto mancano altri passaggi che avevamo escluso a causa di molteplici scelte come nei casi (1) e (2).

ALTERNATIVE

$$(1)\frac{\{\neg A, \neg B\} \quad \{A, B\}}{\{\neg B, \neg B\}}$$

Non ci porta ad inconsistenza in quanto $\{\neg B, \neg B\}$ è una clausola banale.

$$(2)\frac{\{\neg B, C\} \quad \{A, B\}}{\{A, C\}}$$

Poiché A, C non l'abbiamo già ottenuto proseguiamo

$$\frac{\{A, C\} \quad \{\neg A, \neg B, C\}}{\{\neg B, C\}}(2_b)$$

Poiché B, C già l'abbiamo ottenuto è inutile proseguire perché non ci porterà da nessuna parte, però c'è una seconda possibilità, scegliere $\{\neg C, D\}$.

$$(2_b)\frac{\{A, C\} \quad \{\neg C, D\}}{\{A, D\}}$$

A, D non l'abbiamo ancora ottenuto, quindi proseguiamo

$$\frac{\{A, D\} \quad \{\neg D\}}{\{A\}}(2_{b1})$$

Otteniamo, però, A già ottenuto in precedenza. Cerchiamo, quindi, un'altra clausola che possa fare clash con A, D

$$(2_{b1})\frac{\{A, D\} \quad \{\neg A, \neg B, C\}}{\{\neg B, D, C\}}$$

$\{\neg B, D, C\}$ non è stato ancora trovato, quindi proseguiamo

$$\frac{\{\neg B, D, C\} \quad \{A, B\}}{\{A, D, C\}}$$

A, D, C invece lo abbiamo già trovato.

L'idea è quella di provare tutte le coppie e in caso di ciclo, provare a cambiare le scelte fatta per la selezione del clash. Se provando tutte le scelte alternative non si riesce ad ottenere $\square$, allora ci dovrebbe essere un assegnamento.

Allora consideriamo il seguente assegnamento $\alpha : A = T, B = F, C = F, D = F$

Applicando l'assegnamento a $\{\{A, B\}\{\neg A, \neg B, C\}\{\neg C, D\}\{A\}\{\neg D\}\}$, otterremo:

$$\{\{T, T\}\{F, T, F\}\{T, F\}\{T\}\{T\}\}$$

Con questo assegnamento avremo che la forma clausale è soddisfacibile in quanto, da definizione, sappiamo che una clausola è soddisfatta se ha almeno un letterale a true e che la forma causale è soddisfacibile se tutte le clausole sono soddisfatte. In questo caso in ogni clausola c'è almeno letterale a true e di conseguenza la forma causale è soddisfacibile. 13.3.1 13.3.2

14.3 Resolution: Correttezza e Completezza

Teorema 14.3.1

Σ è insoddisfacibile (o inconsistente) $\iff$ *Resolution*(Σ) genera $\square$

Dimostrazione 14.3.1
Dim $\Leftarrow$ (Correttezza)
Per Assurdo, se Σ è soddisfacibile allora $\exists\alpha : AP(\Sigma) \implies \{0,1\}$ tale che $\alpha \vDash \bigwedge \Sigma$ (cioè α soddisfa tutti le formule o clausole $\in \Sigma$). Ma per il teorema 13.4.1 questo significa che α soddisferà anche tutti i risolventi R generati.

Quindi avremo che:

- Allora $\implies$ α soddisfa tutti i risolventi della deduzione per risoluzione
- Allora $\implies$ $\alpha \vDash \square$ (sapendo che $\square$ è un risolvente, cioè può essere generato).

Ma questo è **Assurdo!** perché $\square$ per definizione non è soddisfacibile in quanto non contiene nessun letterale con true e di conseguenza Σ non è soddisfacibile.

Dim $\Rightarrow$ (Completezza)
Partiamo dalla premessa che Σ è insoddisfacibile
Dimostriamo per induzione sulla dimensione di $AP(\Sigma)$ con $n = |AP(\Sigma)|$

- **Caso Base** (n = 0)
  Se $n = 0$, allora $\Sigma = \{\square\}$ (in Forma Clausale), questo perché tutte le clausole in Σ non possono contenere nessun letterale. Allora di conseguenza è ovvio che *Resolution*(Σ) genera $\square$.
- **Caso Induttivo** (n > 0)
  Per **Ipotesi Induttiva**: supporremo che $\forall\Sigma$ inconsistenti/insoddisfacibili con $|AP(\Sigma)| < n$, allora *Resolution*(Σ) genera $\square$

  Decomporremo Σ in:

  - $\Sigma^{p=0} = \{C \setminus \{p\} | C \in \Sigma \wedge \neg p \notin C\}$
    Scarto per questo insieme tutte le clausole che non contengono il letterale $\neg p$ e aggiungo tutte le altre clausole. Se ce n'è qualcuna che contiene p, l'aggiungo scartandogli p. Possiamo osservare che $p = 0$, di conseguenza $\neg p = 1$, quindi stiamo scartando tutte le clausole che sono soddisfatte per il letterale $\neg p$.

  - $\Sigma^{p=1} = \{C \setminus \{\neg p\} | C \in \Sigma \wedge p \notin C\}$
    Scarto per questo insieme tutte le clausole che non contengono il letterale p aggiungo tutte le altre clausole. Se tra queste selezionate ce n'è qualcuna che contiene $\neg p$, l'aggiungo scartandogli $\neg p$. Possiamo osservare che $p = 1$, di conseguenza $\neg p = 0$, quindi stiamo scartando tutte le clausole che sono soddisfatte per il letterale p.

Ricordiamo, inoltre, che questi insiemi non rappresentano una partizione dell'insieme Σ, quindi gli insiemi generati possono avere elementi in comune. Inoltre, possiamo osservare che entrambi gli insiemi non contengono p. Inoltre, se Σ è inconsistente lo sono anche $\Sigma^{p=0}$ e $\Sigma^{p=1}$

Possiamo osservare che entrambi questi insiemi hanno un numero di proposizioni atomiche $< n$ (perché abbiamo scartato qualcosa). Di conseguenza è possibile applicare l'**Ipotesi Induttiva**, ma per poterla veramente applicare dobbiamo anche verificare che questi due insiemi siano insoddisfacibili/inconsistenti.

Supponendo **per assurdo** $\Sigma^{p=0}$ sia soddisfacibile, allora:

$$\exists AP(\Sigma^{p=0}) \to \{0,1\}$$

$$\alpha \vDash \bigwedge \Sigma^{p=0}$$

Cioè α soddisfa tutte le formule o clausole $\in \Sigma^{p=0}$

Allora, questo significa che tutte le clausole hanno un certo letterale che le soddisfi:

$$\forall C \in \Sigma^{p=0} \quad \exists l \in C \text{ tale che } \alpha \vDash l$$

Ma α non è un assegnamento per il nostro originario Σ, poiché in $\Sigma^{p=0}$ manca p.

Allora quello che faremo è costruire un assegnamento α':

$$\forall x \in AP(\Sigma) \quad \alpha'(x) = \begin{cases} \alpha(x) & \text{se } x \neq p \\ 0 & \text{se } x = p \end{cases}$$

Possiamo osservare che $\alpha' \models \bigwedge \Sigma$ (cioè che α' assegnamento soddisfa tutte le clausole di Σ), ma questo è **Assurdo**.

Assurdo? Spieghiamo meglio questo concetto
Osserviamo che in $\Sigma^{p=0}$ non sono contenute le clausole che contenevamo $\neg p$ poichè proprio $\neg p$ le soddisfava. Ma queste clausole scartate sono contenute in Σ e per come abbiamo costruito l'assegnamento α', andremo a soddisfarre tutte le clausole di $\Sigma_{p=0}$ e tutte le clausole che $\notin \Sigma^{p=0}$ ovvero quelle scartate. Quest'ultime saranno appunto ancora soddisfatte perché abbiamo posto in α', $p = 0$ e di conseguenza saranno soddisfatte proprio per $\neg p = 1$.

Allora se da un insieme grande Σ creiamo un insieme più piccolo $\Sigma_{p=0}$ in cui ogni sua clausola è soddisfacibile e tutte le cose che non mettiamo in esso, cioè le clausole scartate, sono anch'esse soddisfacibili, allora siamo riusciti a costruire un assegnamento α' per Σ che soddisfi tutte le sue clausole. Ma questo è **Assurdo**, in quanto sapevamo per premessa (inizio dimostrazione) che Σ era insoddisfacibile e non possiamo ottenere che tutte le sue clausole sono soddisfatte. Allora per forza di cose $\Sigma^{p=0}$ è insoddisfacibile.

Con $p = 1$ avremo una situazione molto simile. Avremo dunque due notazioni inconsistenti.

Arrivati a questo punto sappiamo che:

$$\Sigma^{p=0} \text{ e } \Sigma^{p=1} \text{ sono inconsistenti}$$

Sappiamo anche che non sono vuoti, cioè c'è almeno una clausola che le soddisfa e che sopravvive in entrambe.

Allora rispettiamo le proprietà per poter applicare l'**Ipotesi Induttiva**, cioè sono inconsistenti e hanno $|AP(\Sigma^{p=0})| \quad < n > \quad |AP(\Sigma^{p=1})|$

Per ipotesi induttiva:

$$Resolution(\Sigma^{p=0}) \text{ genera } \square$$

$$Resolution(\Sigma^{p=1}) \text{ genera } \square$$

Aggiungiamo $p \in \Sigma^{p=0}$ (aggiungo p alle clausole da cui l'avevo tolto)

Aggiungiamo $\neg p \in \Sigma^{p=1}$ (aggiungo p alle clausole da cui l'avevo tolto)

Applicando Resolution otterremo:

$$\frac{\{p\} \quad \{\neg p\}}{\square}$$

Esempio 14.3.1
Adesso applichiamo ad una formula in forma clausale ciò che abbiamo detto precedentemente:

$$\Sigma = \{\{p, q\}\{p, \neg q, \neg r\}\{r\}\{\neg p, \neg r\}\{\neg p, q, r\}\}$$

Consideriamo solo le clausole con il letterale p e $\neg p$ per generare i due sottoinsiemi.

Per il primo insieme scarteremo tutte le clausole contenenti $\neg p$, aggiungendo tutte le altre e se queste contengono p, le aggiungeremo togliendolo:

$$\Sigma^{p=0} = \{\{q\}\{\neg q, \neg r\}\{r\}\}$$

Per il secondo insieme scarteremo tutte le clausole le clausole contenenti p, aggiungendo tutte le altre e se queste contengono $\neg p$, le aggiungeremo togliendolo:

$$\Sigma^{p=1} = \{\{r\}\{\neg r\}\{q, r\}\}$$

Dimostriamo se Σ è inconsistente allora $\Sigma^{p=0}$ e $\Sigma^{p=1}$ sono inconsistenti.

Costuiamo ora una prova della $\square$ per Σ

Consideriamo $\Sigma^{p=0}$

$$\frac{\{r\} \quad \{\neg q, \neg r\}}{\{\neg q\}}$$

$$\frac{\{\neg q\} \quad \{q\}}{\square}$$

Consideriamo $\Sigma^{p=1}$

$$\frac{\{r\} \quad \{\neg r\}}{\square}$$

Volendo ripristinare e aggiungeremo p alle clausole a cui l'avevamo tolto e utilizzeremo Resolution:

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

$$\frac{\{\neg q, p\} \quad \{q, p\}}{\{p\}}$$

Volendo ripristinare e aggiungeremo $\neg p$ alle clausole a cui l'avevamo tolto e utilizzeremo Resolution:

$$\frac{\{r\} \quad \{\neg r, \neg p\}}{\{\neg p\}}$$

Ora effettuiamo il clash tra p e ¬ p

$$\frac{\{p\} \quad \{\neg p\}}{\square}$$

Possiamo osservare quindi che da due prove separate, cioè $\Sigma^{p=0}$ e $\Sigma^{p=1}$ possiamo ottenere $\square$ per Σ.

Osservazione 14.3.1

$$p, q \vDash p \wedge q$$

Cerchiamo di ottenere ϕ da Γ ma Γ è in forma clausale $\{\{p\}, \{q\}\}$

Allora Resolution non permette di generare la conseguenza logica (che invece è generabile con i sistemi deduttivi).

Per il **principio di non contraddizione**:

$$p, q, \neg(p \wedge q) \vDash \bot$$

Riscrivendola come:

$$\{p, q, \neg(p \wedge q)\}$$

In Forma Clausale:

$$\{\{p\} \wedge \{q\} \wedge \{\neg p \vee \neg q\}\}$$

Possiamo osservare che è una clausola è soddisfatta se almeno un letterale è vero, inoltre sarà tutto soddisfacibile se tutte le clausole saranno soddisfatte (cioè conterranno almeno un letterale vero). Quindi dovremmo considerare $p = T, q = T$ ma $\neg p \vee \neg q$ non potrà mai avere un letterale a true.

Quindi Resolution non può generare tutte le conseguenze logiche.

Capitolo 15

Lezione 15

15.1 Riducibilità

SAT $\leq_p P_r$ significa che ogni istanza di SAT è riducibile in tempo polinomiale in P_r.
L'obiettivo è quello di prendere un'istanza di P_r per ottenere SAT.

$$P_r \longrightarrow SAT$$

15.2 Codificare Problemi in Logica Proposizionale

Utilizzando un'opportuna codifica della Logica Proposizionale è possibile risolvere una serie di problemi. Ovviamente è difficile che questa codifica permetta di trovare soluzioni più efficienti rispetto a quelle già esaminate durante il corso di Algoritmi e Strutture Dati. Proprio per questo, il focus non sarà affatto l'efficienza, ma ci concentreremo su come modellare problemi e trovare soluzioni attraverso la Logica Proposizionale.

15.2.1 Problema: Verificare se un Grafo è Aciclico

Descrizione: Dato il grafo $G = (V, E)$, verificare se il grafo è aciclico.

Un grafo non è aciclico se esiste un percorso ciclico, cioè: $\exists\pi$ in G ciclico. L'idea consiste nel verificare se possiamo costruire una sequenza di archi che sono adiacenti e che formano un percorso infinito. Inoltre, la codifica riguarderà il grafo e gli archi di quest'ultimo.

Quello che faremo è introdurre una variabile booleana per ogni coppia di vertici:

$$\{X_{vu} | vu \in V\}$$

Questo insieme rappresenterà un insieme di proposizioni atomiche.

Intuitivamente, presi due generici vertici v, u, vorremmo avere che X_{vu} sia true quando v u è un arco; viceversa dovrà essere false quando questa coppia di vertici non sarà un arco del grafo. In generale, ricordiamo che un problema di soddisfacibilità è un problema che presa una formula, vogliamo sapere se

è soddisfacibile, ma una formula è soddisfacibile se esiste un assegnamento per ogni proposizione atomica (con valore true o false).

Proprio per questo definiremo una formula:

$$\phi_G \triangleq \bigwedge_{(vu)\in E} X_{v,u} \ \wedge \bigwedge_{(v,u)\notin E} \neg X_{vu}$$

Possiamo osservare che questa formula sarà soddisfatta da tutti e soli gli assegnamenti che rendono vero X_{vu} dove v u che è un arco del grafo. In altre parole questa formula è la descrizione in logica proposizionale della struttura di G, ovvero la **codifica del grafo** (o codifica d'input).

Arrivati a questo punto vorremo codificare una potenziale soluzione (o codifica d'output). Intuitivamente, vorremmo collezionare un insieme di archi del grafo con cui è possibile costruire un percorso ciclico. Proprio per questo motivo definiremo un ulteriore insieme di proposizioni atomiche che ci permetta di rapprentare l'insieme degli archi che voglio collezionare (simile al precedente):

$$S = \{Y_{vu} | v, u \in V\}$$

Intuitivamente, ogni assegnamento assegnerà a queste proposizioni atomiche un valore true o false. Di nuovo, le proposizioni atomiche con valore true sono quelle che voglio collezionare nell'insieme S (un arbitrario sottoinsieme); viceversa quelle con valore false sono le proposizioni che voglio escludere.

$$S \subseteq \{(v,u)|(v,u) \in E\} \text{ un sottoinsieme arbitrario di } E \text{ (archi)}$$

Ovviamente bisogna garantire che le coppie di vertici in S siano effettivamente degli archi del grafo, cioè che qualsiasi Y_{uv} a true sia effettivamente un arco del grafo. Di conseguenza possiamo osservare che gli archi del grafo sono le X_{vu} a true definite in precedenza, quindi possiamo definire (sfruttando i valori di verità dell'implicazione):

$$\phi_S \triangleq \bigwedge_{v,u\in E} (Y_{vu} \longrightarrow X_{vu})$$

In altre parole, questa formula ci dice che per ogni coppia di vertici vu che metto in S, allora per forza di cose questa coppia di vertici deve essere un arco (per i valori di verità dell'implicazione).

In sostanza il nostro insieme S, corrisponderà:

$$S = \{(v,u) \mid \alpha(Y_{vu}) = 1\}$$

Abbiamo detto che S può essere un insieme arbitario di archi, quindi eventualemente anche un insieme vuoto. Ma non vogliamo che S sia vuoto, proprio per questo motivo definiremo:

$$\underset{S\neq\emptyset}{\phi} \triangleq \bigvee_{v,u\in V} Y_{vu}$$

In altre parole, vorremmo che esiste almeno un arco in S. In logica proposizionale, l'equivalente di "esiste almeno" viene rappresentato con $\bigvee$ (big OR). Con questo simbolo specificheremo che può almeno una Y_{uv} deve essere true e di conseguenza apparterrà ad S, garantendo che S non sia vuoto.

L'ultima cosa che ci rimane da dire è che S contenga un percorso infinito, cioè con gli archi che ho collezionato in esso è possibile costruire un percorso ciclico. Ricordiamo che un percorso ciclico è infinito, cioè per ogni arco c'è un arco adiacente (sempre contenuto in S). Definiremo quindi:

$$\phi_{cyc} \triangleq \bigwedge_{v,u\in V} (Y_{vu} \longrightarrow \bigvee_{z\in V} Y_{uz}) \quad \text{[S contiene un ciclo]}$$

Dove $\bigwedge$ (big AND) rappresenterà il "per ogni" arco che prendo, se questo è un arco (quindi se Y_{vu} è vero), allora deve "esistere" un altro arco, rappresentato con $\bigvee$ (big OR), che parte da dove finisce il primo arco (cioè parte da u) e va in un certo vertice $z \in S$ tale per cui Y_{uz} è true. Quindi questa formula ci rappresenterà la presenza di un ciclo, poiché un percorso infinito in un grafo finito implica che ci dovrà per forza di cose essere un ciclo.

Allora la formula finale sarà:

$$\phi \triangleq \phi_G \ \wedge \ \phi_S \ \wedge \ \bigwedge_{S \neq \emptyset} \phi \ \wedge \ \phi_{cyc}$$

Tale formula ϕ sarà soddisfacibile **se e solo se** il grafo non è aciclico. Inoltre se ϕ è soddisfacibile e da ogni assegnamento che la soddisfa posso estrarre una soluzione (o un testimone). Possiamo osservare che se tutte le formule della formula finale sono vere, allora deve esiste un ciclo.

Il numero di proposizioni atomiche è il seguente

- $|\phi_G| = |V^2|$
- $|\phi_S| = 4|V|^2$
- $|\bigwedge_{S \neq \emptyset} \phi| = 2|V|^2$
- $|\phi_{cyc}| = 3|V|^3$

Quindi avremo che $\phi = \Theta(|V|^3)$.

15.2.2 Problema: K-Probability

Descrizione: Dato un grafo $G = (V, E)$ e un insieme di k colori, vogliamo associare ad ogni vertice di G un colore in modo che due vertici adiacenti non abbiano lo stesso colore.

Dato un grafo $G = (V, E)$ ed un insieme di colori C con $|C| = k$, vorremmo:

- Costruire una funzione $f : V \to C$ (cioè f associa esattamente un colore)
- $\forall (u, v) \in E,\ f(v) \neq f(u)$ (cioè che per ogni coppia di vertici adiacenti, questi non abbiano lo stesso colore)

Possiamo osservare che per come abbiano definito f, questa risulta essere una relazione, cioè un sottoinsieme del prodotto cartesiano, cioè un insieme di coppie (vertice, colore):

$$f \subseteq \{(v, c) \mid v \in V \text{ e } c \in C\}$$

Di conseguenza, una soluzione al problema consisterà in un insieme di coppie.

Per codificare l'insieme di coppie utilizzeremo un insieme di varibili booleane per ogni coppia:

$$\{X_v^C \mid c \in C, v \in V\}$$

Come per il problema precedente, un certo assegnamento renderà vere alcune coppie e false alcune di queste. Ma in f avremo le coppie che sono vere per un certo assegnamento (ogni assegnamento rappresenta una potenziale soluzione).

Inanzitutto un insieme di coppie non è detto che rappresenti una funzione, quindi dovremo imporre il **vincolo di funzionalità**, cioè che ad ogni vertice è possibile associare solo un unico colore. In altri termini $\forall v \in V$, non può esistere $X_V^{c_1} = 1$ e $X_V^{c_2} = 1$. Questo significa che un vertice può avere un solo colore associato (true) e tutti gli altri devono essere false. Proprio per questo definiremo una formula:

$$\phi_{UN} \triangleq \bigwedge_{v \in V} \Big(\bigvee_{c \in C} \big(X_v^c \ \wedge \bigwedge_{c' \in C \ \mid \ c' \neq c} \neg X_v^{c'} \big) \Big)$$

In altre parole, vorremmo che per "ogni vertice" v (big AND), "esista almeno" (big OR) un colore associato dove X_v^c è true e per tutti gli altri colori c' (big AND) vale che $X_v^{c'}$ è false.

La cardinalità della seguente formula é $\Theta(|V||C|^2) = |\phi_{UN}|$, dove il primo big AND corrisponde alla cardinalità di V, il primo big OR corrisponde alla cardinalità dei colori(C) ed il secondo big AND corrisponde ancora alla dimensione di C.

Ora per garantire che due vertici adiacenti non possono avere lo stesso colore definiremo:

$$\phi_{\text{color}} \triangleq \bigwedge_{(v,u) \in E} \Big(\bigwedge_{c \in C} (\neg X_v^c \vee \neg X_u^c) \Big)$$

In altre parole, per "ogni arco" (v, u) (big AND) e "per ogni colore" (big AND) vale che uno dei due X^c è true e l'altro è false (uno dei due deve essere false).
La cardinalità della seguente formula è $\Theta(|V||C|) = |\phi_{\text{color}}|$.

Allora la formula finale sarà:

$$\phi \triangleq \phi_{UN} \wedge \phi_{\text{color}}$$

Possiamo osservare che questa formula sarà vera **se e solo se** esiste una k-colorazione rappresentata proprio dall'insieme di coppie definito in precedenza e ne costruisco la funzione. In altri termini, dato

un assegnamento α costruisco f che soddisfa i vincoli; oppure presa una funzione f che rispetti i vincoli, costruisco un α che mette a vero esattamente le coppie in f:

$$f(v) = c \iff \alpha(X_v^c) = 1$$

Osservazione:

C'è un modo per ottimizzare la formula rimuovendo ϕ_{UN} e consiste nel forzare ogni vertice ad avere **esattamente** un colore, semplicemente con la scelta delle proposizioni atomiche.

L'idea consiste nel codificare il colore con un numero che in logica proposizionale equivale a una codifica binaria, dove ogni proposizione atomica corrisponde ad un bit. Ma quindi associare un numero ad ogni vertice richiederà $|V|\ log_2 k$ variabili booleane (o proposizioni atomiche), così definite:

$$\{X_v^0\ ...\ X_v^{log_2 k-1} \mid v \in V\}$$

Ogni assegnamento assegnerà qualche valore di verità alle variabili, cioè assocerà esattamente un numero a ciascun vertice.

$$f(v) = c \iff X_v^0\ ...\ X_v^{log_2 k-1} \quad \text{[codifica del colore in binario]}$$

Questo già ci garantisce che un assegnamento individua una funzione poiché lo stesso vertice non può avere due sequenze (o codifiche) e quindi due colori.

Ma dobbiamo ancora garantire che se presi due vertici di un arco, questi due vertici devono avere due colori diversi. Ma i colori sono rappresentati attraverso una stringa di bit, quindi vedere che i colori sono diversi equivale a vedere se due stringhe di bit sono diverse, cioè se esiste una posizione in cui un bit è diverso. Di conseguenza possiamo sfruttare l'XOR, cioè si effettua il confronto bitwise. Proprio per questo definiremo:

$$\phi_{\text{color}} \triangleq \bigwedge_{(v,u)\in E} \left(\bigvee_{i=0}^{log_2 k-1} (X_v^i \oplus X_u^i) \right)$$

In altre parole, per "ogni vertice" (u, v) (big AND), "esiste" almeno (big OR) un bit diverso.
La cardinalità della seguente formula è $|\phi_{\text{color}}| = O(|E|\ log_2 k)$.

Se portassimo la formula ϕ_{color} in CNF, avremo che i congiunti siano molto lunghi (in quanto abbiamo molti bit da considerare). Supponendo per esempio che $|C| = 2$ e quindi $log_2 k = 1$, cioè che per ogni vertice c'è un bit. Ma se per ogni vertice c'è un bit allora avremo variabili con lunghezza 1: (polinomiale come si può vedere di seguito)

$$\{X_v | v \in V\}$$

Di conseguenza possiamo scrivere ϕ_{color} (scrivendo lo XOR come segue) come:

$$\phi_{\text{color}} \triangleq \bigwedge_{(v,u)\in E} ((X_v \wedge \neg X_u) \vee (\neg X_v \wedge X_u))$$

Portandola in CNF (sfruttando la distributività):

$$\phi_{\text{color}} \triangleq \bigwedge_{(v,u)\in E} (((X_v \wedge \neg X_u) \vee \neg X_v) \wedge ((X_v \wedge \neg X_u) \vee X_u))$$

Che diventa (sfruttando la distributività):

$$\phi_{\text{color}} \triangleq \bigwedge_{(v,u)\in E} ((X_v \vee \neg X_v) \wedge (\neg X_u \vee \neg X_v) \wedge (X_v \vee X_u) \wedge (\neg X_u \vee X_u))$$

Le formule $(X_v \vee \neg X_v)$ e $(\neg X_u \vee X_u)$ vengono tolte poiché sono true, ottenendo:

$$\phi_{\text{color}} \triangleq \bigwedge_{(v,u)\in E} ((\neg X_u \vee \neg X_v) \wedge (X_v \vee X_u))$$

Il risultato trovato è una 2CNF e i problemi 2CNF sono risolvibili in tempo polinomiale.

Proposizione 15.2.1

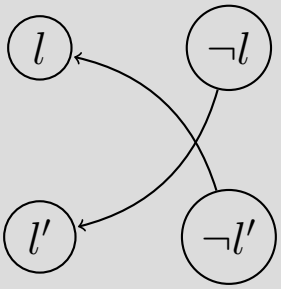
2CNF è risolvibile in tempo polinomiale

Dimostrazione 15.2.1

L'idea alla base è che se ogni clausola ha due letterali l e l'.
Possiamo osservare che un OR è equivalente a un'IMPLICAZIONE:

$$(l \vee l') \equiv (\neg l' \to l) \qquad (l \vee l') \equiv (\neg l \to l')$$

Allora possiamo rappresentare questa **relazione di dipendenza** tra la verità dei letterali (cioè se è vero uno dei due letterali deve essere vero anche l'altro e viceversa) attraverso un grafo dove gli archi sono interpretabili come relazione di dipendenza:



2CNF è insoddisfacibile $\iff$ se nel grafo c'è una componente fortemente connessa che contiene sia il letterale che il suo complementare.

In altre parole se $\neg l$ è vero, deve essere vero anche l' e di conseguenza non esiste nessun assegnamento che possa rendere vero sia $\neg l'$ che l', mentre la loro dipendenza dice che devono essere entrambi veri (ma questo è impossibile). Quindi il problema della soddisfacibilità si riduce a un problema di verifica delle componenti fortemente connesse e poi verifica se ce n'è una che contiene un letterale e il suo complemento.

15.2.3 Problema: Vertex Cover

Descrizione: Dato un grafo $G = (V, E)$, verificare se esiste un sottoinsieme di vertici $S \subseteq V$ che garantisce la più piccola copertura dei vertici (https://www.geeksforgeeks.org/introduction-and-approximate-solution-for-vertex-cover-problem/) dove:

- $\forall (v, u) \in E$ tale che $\{v, u\} \cap S \neq \emptyset$ (almeno uno degli estremi di ciascun arco deve stare in S)
- $|S| \leq k$, dove k è un input fissato

L'idea alla base è che essendo $S \subseteq V$ allora per rappresentare una soluzione basta avere tante variabili booleane (o proposizioni atomiche) quanto sono i vertici:

$$\{X_v \mid v \in V\}$$

Ogni assegnamento assegnerà un qualche valore di verità vero o falso e intuitivamente, quelle variabili con valore vero saranno contenute in S. Preso quindi un assegnamento α, avremo che:

$$S_\alpha = \{v \mid \alpha(X_v) = 1\}$$

Definiamo quindi una formula:

$$\phi_{\text{cover}} \triangleq \bigwedge_{(v,u) \in E} (X_v \vee X_u)$$

Questa formula garantisce che l'insieme S è una copertura, cioè "per ogni" (big AND) vertice, almeno uno dei due deve essere in S. Possiamo inoltre osservare che è una formula in 2CNF e quindi verificarla è semplice, ma il problema è verificare il vincolo che $|S| <= k$

Preso un assegnamento α, sappiamo che $\alpha : X \to \{0, 1\}$, cioè indurrà a un insieme S_α e vorremmo rispettare il vincolo che $|S_\alpha| \leq k$. Possiamo osservare che la cardinalità di S_α è il numero di vertici che hanno assegnamento uguale a 1 (true), che in altre parole può essere visto come il numero di 1 che α associa nell'encoding.

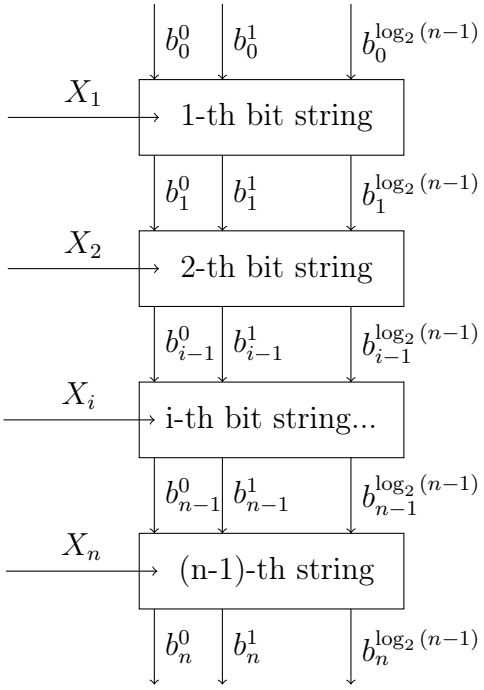
Quello che dovremmo fare è scrivere una formula che conti quante $X_v = 1$. Una volta ottenuto il conteggio, bisognerà confrontarlo con k dato in input..

Intuitivamente tutto ciò è fattibile poiché se abbiamo un numero finito di valori booleani, per calcolare quanti di questi valori sono uguali a 1 è possibile costruire un **circuito**. L'output del circuito corrisponderà a una sequenza di bit la cui lunghezza sarà la lunghezza massima che mi servirà per codificare il massimo numero di 1 e confronteremo tale lunghezza utilizzando un ulteriore circuito. Di conseguenza se c'è un circuito, c'è anche una formula booleana che rappresenta quel circuito.

Allora se abbiamo n input del circuito X_1, ..., X_n, nel caso peggiore (cioè se fossero tutte quante le varibili booleane vere) otterremo n come numero massimo. Di conseguenza per rappresentare n in binario ci serviranno $log_2(n)$ bit.

Questo circuito rappresenta una sorta di incrementer, cioè che per ogni variabile booleana X_1, ..., X_n se questa è vera attraverso l'assegnamento α, si aggiunge 1 al conteggio corrente. Possiamo immaginare il tutto come un circuito a stadi diversi, dove ogni stadio produce una configurazione, cioè una stringa di bit rappresentante il valore del contatore.

La codifica di S richiederà $n\ log_2(n)$ variabili.



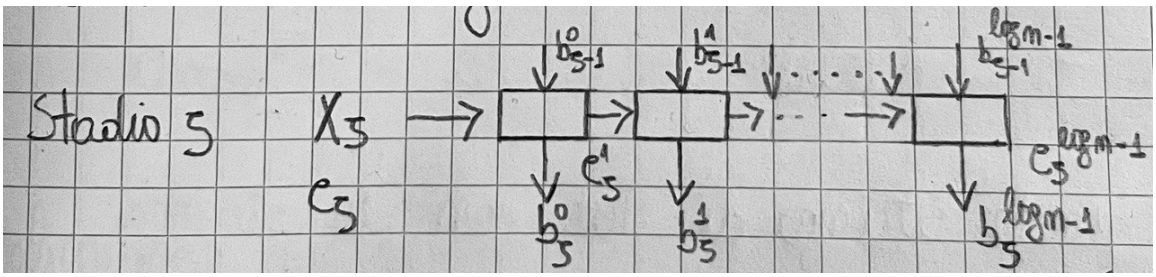
Introduciamo un primo insieme di variabili b_j^i (che rappresentano i bit della sequenza di bit):

$$\{b_j^i \mid i \in \{0, \ldots, log_2(n-1)\},\ j \in \{0, \ldots, n\}\}$$

Definiremo una formula B_{INIT} che si occuperà di inizializzare con zero tutti i valori delle variabili booleane dello stato iniziale b_0:

$$B_{INIT} = \bigwedge_{i=0}^{\log_2(n-1)} \neg b_0^i$$

Osservando in dettaglio lo stadio j-imo del circuito, avremo:



Ora bisogna definire come generare data una configurazione, la configurazione successiva considerando il potenziale riporto nel sommare due bit con valore 1. Quindi quello che faremo è introdurre delle variabili di riporto (*carry*), una per ogni blocco del circuito:

$$carry : \{c_j^i \mid i \in \{q, \ldots, \log_2(n-1)\},\ j \in \{0, \ldots, n\}\}$$

Definiamo i valori che le variabili b_j^i devono assumere, definendo la seguente formula:

$$B_j^i \triangleq (b_j^i \leftrightarrow (c_j^i \oplus b_{j-1}^i))$$

Questa formula definisce qual è il valore delle variabili che viene calcolato prendendo il bit corrente (b_{j-1}^i) e mettendolo in XOR con il *carry*, ottenendo 0 se il valore del bit corrente e quello del *carry* sono uguali; altrimenti otterremo 1, cioè uno è 0 e l'altro è 1 (o viceversa).

Definiamo i valori che le variabili *carry* devono assumere, defininendo la seguente formula:

$$C_j^{i+1} \triangleq (c_j^{i+1} \leftrightarrow (c_j^i \wedge b_{j-1}^i))$$

Questa formula definisce qual è il valore del *carry* che viene calcolato prendendo il bit corrente (b_{j-1}^i) e mettendolo in AND con il *carry*, ottenendo per il carry c_j^{i+1} il valore 1 se $b_{j-1}^i = 1$ e $c_j^i = 1$; altrimenti otterremo il valore 0.

Definiremo due formule chi si occuperanno di descrivere l'intero stadio $j - imo$ del circuito:

$$B_j \triangleq \bigwedge_{i=0}^{\log_2(n-1)} B_j^i$$

Questa formula descrive per l'intero stadio $j - imo$ il valore delle variabili definite in precedenza.

$$C_j \triangleq (C_j^0 \leftrightarrow x_j) \wedge \bigwedge_{i=0}^{\log_2(n-1)} C_j^{i+1}$$

Questa formula descrive la configurazione del *carry* per l'intero stadio $j - imo$. Possiamo notare che il big AND descrive la configurazione del *carry* da $C^1, ..., C^{log(n-1)}$, a cui andremo ad aggiungere il *carry* iniziale C^0 che dipenderà dall'input X_j.

Arrivati a questo punto definiamo una formula che descrive completamente l'intero circuito, mettendo assieme tutte le formule precedenti per tutti gli stadi del circuito:

$$\phi_{count} \triangleq B_{INIT} \wedge \bigwedge_{j=1}^{n} B_j \wedge \bigwedge_{j=0}^{n-1} C_j$$

Allora per ogni assegnamento α che soddisferà sia ϕ_{cover} che ϕ_{count}, avremo che:

$$b_n^0, b_n^1, \ldots, b_n^{\log_2(n-1)} = |\phi_{cover}|$$

In altre parole, preso un assegnamento che soddisferà le due formule, produrrà in output una stringa di bit la cui lunghezza è uguale al numero di vertici selezionati.

Arrivati a questo punto quello che ci resterà fare è confrontare questa stringa d'output (che rappresenta il numero di vertici selezionati per la copertura minima) con la stringa rappresentante il numero k dato in input.

Quindi di conseguenza anche per k la sua stringa di bit:

$$k^0, \ldots, k^{\log_2(k-1)} \text{ variabili booleane}$$

Definiremo la formula che ci indicherà quali variabili devono essere vere e quali dovranno essere false, cioè quali saranno a 0 e quali saranno a 1:

$$\phi_k \triangleq \bigwedge_{i \in S_k} k^i \wedge \bigwedge_{i \notin S_k} \neg k^i \quad \text{con } S_k = \{i \mid \text{i-esimo bit di k è a 1}\}$$

Ovviamente le variabili vere corrispondono a un bit della stringa con valore a 1; viceversa le variabile false corrisponderanno a un bit della stringa con valore 0.

Quindi fino a questo punto abbiamo che:

$$size = b_n^0\ b_n^1 \ldots b_n^{\log_2(n-1)}$$

$$k = k^0 \dots k^{\log_2(k-1)}$$

E vorremmo codificare che $size \leq k$ attraverso una formula.

Per semplicità, proviamo a scrivere una formula per una sequenza con un solo bit. Possiamo esplicitare la relazione $b_n^0 \leq k^0$ come:

$$b_n^0 \leq k^0 \equiv (\neg b_n^0 \wedge k^0) \vee (b_n^0 \leftrightarrow k^0)$$

Questa formula controlla che il primo bit ha valore più piccolo del secondo o entrambi i bit hanno valore uguale.

Se considerassimo invece ora due numeri a due bit. Possiamo esplicitare la relazione come:

$$b_n^0\ b_n^1 \leq k^0\ k^1 \equiv (\neg b_n^1 \wedge k^1) \vee (b_n^1 \leftrightarrow k^1 \wedge b_n^0 \leq k^0)$$

Questa formula controlla che il bit più significativo della prima stringa (b_n^1) è più piccolo (cioè ha valore false) rispetto a quello più significativo della seconda stringa (k_n^1) oppure se entrambi i bit più significativi hanno lo stesso valore, necessariamente il secondo bit della prima stringa deve essere minore o uguale rispetto a quello della seconda stringa ($b_n^0 \leq k_n^0$) (cioè si confrontano i bit meno significativi della due stringa).

Se considerassimo invece due numeri a tre bit, avremmo che:

$$b_n^0\ b_n^1\ b_n^2 \leq k^0\ k^1\ k^2 \equiv (\neg b_n^2 \wedge k^2) \vee (b_n^2 \leftrightarrow k^2 \wedge (b_n^0 b_n^1 \leq k^0 k^1))$$

Possiamo osservare la ricorsione che sussiste nel confronto tra i numeri.

In generale quello che faremo è introdurre una serie di variabili booleane, una per ogni posizione.
Quindi date le due stringhe di bit:

$$b_n^0 \dots b_n^i \leq k^0 \dots k^i$$

Definiremo per ricorsione allora:

- **Caso Ricorsivo**:
$$Comp^i \leftrightarrow (\neg b_n^i \wedge k^i) \vee (b_n^i \leftrightarrow k^i \wedge Comp^{i-1})$$
$$\text{con } i \in \{-1, 0, \dots, \log_2(k-1)\}$$
- **Caso Base**:
$$Comp^{-1} = T$$

Arrivati a questo punto possiamo definire la seguente formula:

$$\phi_{Comp} \triangleq \bigwedge_{i=-1}^{\log_2(k-1)} Comp^i$$

Questa formula si occupa di descrivere il comportamento del comparatore per le due stringhe.

Intuitivamente, se considerassimo $log_2(k-1)$ bit, il comparatore restituirà 1 se:

$$Comp^{\log_2(k-1)} = 1 \iff \text{i primi } \log_2(k-1) \text{ bit di size} \leq \text{primi } \log_2(k-1) \text{ di k}$$

Arrivati a questo punto possiamo scrivere la formula complessiva per il comparatore:

$$\phi_{size}^{\leq k} \triangleq \phi_{Comp} \wedge \bigwedge_{i=\log_2(k)}^{\log_2(n-1)} \neg b_n^i \wedge Comp^{\log_2(k-1)}$$

Allora la formula finale risulterà:

$$\phi \triangleq \phi_{Cover} \wedge \phi_{Count} \wedge \phi_k \wedge \phi_{size}^{\leq k}$$

Possiamo inoltre osservare che:

- $\phi_{cover} = |E|$
- $\phi_{count} = n \log_2(n)$
- $\phi_k = \log_2(k)$
- $\phi_{size}^{\leq k} = ?$

Allora ϕ è polinomiale sulla dimensione del grafo.

Capitolo 16

Lezione 16

16.1 Dalla Logica Booleana alla Logica del Primo Ordine

La logica booleana (o proposizionale) risulta essere abbastanza espressiva fintanto che il dominio considerato è finito. Per esempio, qualsiasi grafo *finito* può essere tradotto in logica booleana. Se considerassimo domini infiniti, la logica booleana risulterebbe poco efficiente.

Esempio 16.1.1
Per esempio, non esiste in logica booleana la possibilità di scrivere:

$$\textit{Tutti i } P \textit{ sono } Q$$

Se "Tutti", cioè gli individui di cui voglio parlare, fosse un qualcosa di finito potremmo usare la logica booleana e scrivere:

$$\bigwedge_{a \in D} P_a \to Q_a$$

Per esempio, non esiste in logica booleana la possibilità di scrivere:

$$\textit{Alcuni } P \textit{ sono } Q$$

Se "Alcuni", cioè gli individui di cui voglio parlare, fosse un qualcosa di finito potremmo usare la logica booleana e scrivere:

$$\bigvee_{a \in D} (P_a \to Q_a)$$

Possiamo osservare che se tutti i P_a fossero falsi, l'implicazione risulterebbe comunque vera per la semantica dell'implicazione (cioè i suoi valori di verità).

16.2 Logica del Primo Ordine

La logica del primo ordine ha la possibilità di "parlare" di oggetti del dominio e delle loro proprietà e relazioni. Inoltre, è possibile definire delle "trasformazioni" agli oggetti del dominio.

16.2.1 Linguaggio L

Possiamo definire il linguaggio L come:

$$L = (X, C, F, R, \text{arietà})$$

- X le variabili
- C simboli costanti (nomi tramite i quali mi riferisco agli oggetti)
- F simboli di funzione
- R simboli di relazione

Inoltre, per ogni $f \in F$ e $p \in R$ è associata una **arietà**, ovvero un numero che specifica il numero di parametri che quella funzione o quella relazione accetta.

Esempio 16.2.1

$$f(x, y, z) \quad \text{allora } arieta'(f) = 3$$

16.2.1.1 Assunzione

L'unica assunzione che faremo è avere potenzialmente infinite variabili, infinite costanti, ecc...

16.2.2 Termine

Definizione 16.2.1
I termini sono i modi in cui ci riferiamo agli oggetti del dominio e sono costruiti a partire:

- Dalle variabili, dove $x \in X$ è un termine
- Dalle costanti, dove $c \in C$ è un termine
- Dai simboli di funzione, dove $f(t_1 \dots t_n)$ è un termine se:
  - $t_1, \dots, t_n$ sono termini e $f \in F$ con arietà n

Esempio 16.2.2

- 1
- (0+1)
- (2-1)

Sono tutti termini diversi che si riferiscono allo stesso oggetto.

16.2.3 Variabili del termine

Definizione 16.2.2
Sia t termine,

$Var(t)$ è l'insieme delle variabili che occorrono nel termine t

Esempio 16.2.3
Se t è un termine tale che $t = h(x, c, g(y, z))$, con $c \in C$ e $x, y, z \in X$, allora:

$$Var(t) = \{x, y, z\}$$

16.2.3.1 Termine Ground

Definiremo **termine ground** (o chiuso) un termine senza variabili, cioè se $Var(t) = \varnothing$.

16.3 Linguaggio L e Simboli Logici

Il linguaggio L contiene anche **simboli logici**, cioè conterrà tutti i simboli della Logica Proposizionale (PL) come $\{\wedge, \vee, \dots\}$, a cui andremo ad aggiungere $\{\forall, \exists\}$.

16.3.1 Formule Ben Formate: FO-WFF

Definizione 16.3.1
Una formula del primo ordine ben formata è una sequenza di simboli definita come:

- **Caso Base** (Formula Atomica o Formule Base):

  1. $P(t_1, \dots, t_n)$ è una formula atomica ben formata con:
     - $P \in R$ e con arietà n
     - $t_1, \dots, t_n$ sono termini di L
  2. $t_1 = t_2$ è una formula atomica ben formata con:
     - t_1 e t_2 sono termini

  Inoltre, 1 e 2 saranno formule atomiche indipendentemente dal fatto che i termini contengano o meno variabili.

- **Caso Induttivo**: (Formule Composte)
  - ϕ_1 op ϕ_2 se ϕ_1 e ϕ_2 formule ben formate e $op \in \{\wedge, \vee, \rightarrow, \leftrightarrow, \dots\}$
  - $\neg\phi$ è una formula se ϕ è una formula ben formata
  - $\forall x.\phi$ e $\exists x.\phi$ se ϕ è una formula ben formata

Esempio 16.3.1
$x = 3$ è una formula atomica riscrivibile anche come $(x, 3)$
$s \leq 3$ è una formula atomica, come anche $s \leq (3 + i)$

16.3.2 Variabili Libere

Definizione 16.3.2
Una variabile x è **libera** se non c'è un quantificatore ($\forall$, $\exists$), cioè se la variabile non si trova nello scope del quantificatore.

$(\forall x \dots y)$ y è libera

16.3.3 Variabili Legate

Definizione 16.3.3
Una variabile x è **legata** se c'è un quantificatore ($\forall$, $\exists$), cioè se la variabile si trova nello scope del quantificatore.

$(\forall x \dots x)$ x è legata

Esempio 16.3.2
Dati $x, y, z \in X$ e $c \in C$

$$\phi = (\forall x.P(f(c), g(x), f(x, y))) \wedge (\exists y.(P(x, y) \vee P(y, z)))$$

- $\forall x$ e $\exists y$ sono quantificatori
- $P(f(c), g(x), f(x, y))$ e $(P(x, y) \vee P(y, z))$ sono formule atomiche
- Nella prima formula le variabili x sono *scope di* $\forall$ e nella seconda le variabili y sono *scope di* $\exists$

Possiamo osservare che:

- x e y sono sia legate che libere nella formula
- z è solo libera nella formula

16.3.4 Insieme delle Variabili Libere

Definizione 16.3.4
Siano ϕ e ψ formule, l'insieme delle variabili **libere** (FV) di ϕ:

$$FV(\phi) = \begin{cases} \bigcup_{i=1}^{n} Var(t_i) & \text{se } \phi = P(t_1, \dots, t_n) \\ FV(\psi) & \text{se } \phi = \neg\psi \\ FV(\psi_1) \cup FV(\psi_2) & \text{se } \phi = \psi_1 \ op \ \psi_2 \quad op \in \{\wedge, \vee, \rightarrow, \dots\} \\ FV(\psi) \setminus \{x\} & \text{se } \phi = Qx.\psi \quad \text{con } Q \in \{\forall, \exists\} \end{cases}$$

16.3.5 Insieme delle Variabili Legate

Definizione 16.3.5
Siano ϕ e ψ formule, l'insieme variabili **legate** (BV) di ϕ:

$$BV(\phi) = \begin{cases} \varnothing & \text{se } \phi = P(t_1, \ldots, t_n) \\ BV(\psi) & \text{se } \phi = \neg\psi \\ BV(\psi_1) \cup BV(\psi_2) & \text{se } \phi = \psi_1 \ op \ \psi_2 \quad op \in \{\wedge, \vee, \rightarrow, \ldots\} \\ BV(\psi) \cup \{x\} & \text{se } \phi = Qx.\psi \quad \text{con } Q \in \{\forall, \exists\} \end{cases}$$

16.3.6 Formula Chiusa

Definizione 16.3.6
Se $FV(\phi) = \varnothing$ allora ϕ è detta **formula chiusa** (o proposizione). In altre parole, una formula è chiusa se il suo insieme di variabili libere è vuoto.

16.3.7 Formula Aperta

Definizione 16.3.7
Se $FV(\phi) \neq \varnothing$ allora ϕ è detta **formula aperta**. In altre parole, una formula è aperta se il suo insieme di variabili libere non è vuoto.

Esempio 16.3.3
$x = y$ è una formula aperta
$\forall.x \ x = x$ è una formula chiusa

Capitolo 17

Lezione 17

17.1 Logica per Uguaglianza

La relazione "=" ha arietà 2 e se è simbolo logico del linguaggio (cioè con $= \in L$), allora avremo la **logica del primo ordine con uguaglianza**.

$=$ è una relazione **interpretata a priori**

Interpretata a priori sta a significare che il suo significato non dipende dal modello (o dai modelli) che stiamo considerando.

Esempio 17.1.1
Sappiamo già che i termini possono essere visti come modi diversi per parlare del dominio.

Supponendo che il dominio sia quello dei numeri naturali e che abbiamo i seguenti tre simboli costanti:

$$\overline{1} \quad \overline{2} \quad \overline{3}$$

E supponiamo di avere un simbolo di funzione $\overline{+}$

Allora $\overline{+}(\overline{1}, \overline{2})$ rappresenta un nome alternativo a $\overline{3}$

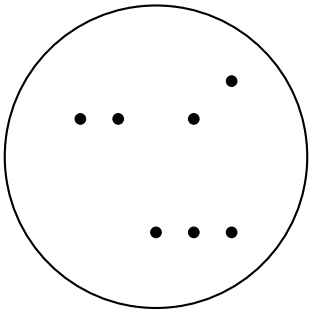
Possiamo anche esprimere il seguente termine ground $\overline{+}(\overline{1}, \overline{2})$ con la notazione infissa, cioè:

$$\overline{+}(\overline{1}, \overline{2}) \xrightarrow{\text{notazione infissa}} [\overline{1} \ \overline{+} \ \overline{2}]$$

17.2 Semantica

Dato il seguente linguaggio $L = (C, F, R)$, le variabili X e i simboli logici (visti in precedenza). L'intuizione è quella di volersi riferire a "oggetti" con dei nomi.

Ipotizziamo di avere un dominio D: insieme non vuoto di individui o elementi ($D \neq \emptyset$).



D

L'idea è che ognuno di questi elementi potrebbe avere un nome del linguaggio che è o una costante del linguaggio oppure un termine (costruito a partire da costanti e funzioni). Inoltre, le variabili X sono utilizzate per riferirsi a oggetti del dominio D.

Di conseguenza, ogni interpretazione sensata della logica del primo ordine richiede un dominio. L'unico requisito richiesto è che sia non vuoto, ma può essere finito o infinito.

Arrivati a questo punto, dobbiamo defininire una corrispondenza tra nomi ed oggetti e dare un significato agli oggetti e alle relazioni.

Esempio 17.2.1

$$P \in R \quad \text{e con } arity(P) = 2$$
$$a, b \in C$$

Ma $P(a, b)$ a cosa si riferisce?

17.3 FO-Structure: Struttura del Primo Ordine

Il concetto di **semantica** parte dal concetto di **struttura** del primo ordine.

Definizione 17.3.1
La struttura del primo ordine (FO-Structure) è una coppia:

$$M = (D, I)$$

Con D **dominio** ed I **funzione di interpretazione**.

17.3.1 Funzione D'Interpretazione

Definizione 17.3.2
La **funzione di interpretazione** si occupa di collegare il linguaggio L al dominio D. L'associazione avviene nel seguente modo:

- $I(c) \in D$ se $c \in C$ (cioè cattura il concetto di nome: c è il nome dell'elemento d in L).

- $I(f) : D^k \to D$ se $f \in F$ di arietà k (cioè è un *mapping* che mette in relazione k elementi del dominio con un elemento del dominio)
- $I(p) \subseteq \underbrace{D \times D \times \cdots \times D}_{k \text{ volte}} = D^k$ se $p \in R$ di arietà k

  In altre parole quello che sta a significare è che:

  $I(p) = \{(d_1, ..., d_k) | I(p)(d_1, ..., d_k)\}$ insieme di k tuple che stanno nella relazione p tra di loro

Esempio 17.3.1
Se $I(c) = d \in D$, allora c è il nome di $d \in L$

Esempio 17.3.2

$$I(\overline{+}) = + \quad \text{nome di operazione di somma}$$

L'interpretazione di $\overline{+}$ è esattamente l'operazione di somma.

Questo conclude quindi il concetto di struttura, che di fatto sono gli **oggetti matematici** che possono fungere da modelli per le formule della logica proposizionale.

17.4 Introduzione all'interpretazione del termine

Denoteremo con $[|\ldots|]^M$ l'interpretazione del termine e dove M rappresenta la struttura rispetto alla quale voglio calcolare l'interpretazione del termine (per i termini ground, cioè senza variabili). Per i termini non ground specificheremo anche la funzione d'assegnamento σ.

Esempio 17.4.1

$$[|\overline{1}\overline{+}\overline{2}|]^M = 3$$

$$I(\overline{1}) = 1 \quad I(\overline{2}) = 2 \quad I(\overline{+}) = +$$

Ma non siamo ancora in grado di dire $\overline{1} + x$ a cosa si riferisce? Cioè l'informazione che il modello (o struttura) M ci dà non ci dice nulla su come interpretare le variabili.

É ovvio che sia così, perché una volta fissata una struttura M, cioè un dominio con delle operazioni (funzioni, relazioni), noi vogliamo poterci riferire in maniera arbitraria agli elementi di quella struttura. Per esempio potremmo chiederci se "al variare di x è uguale ad un altro termine oppure no?". In altre parole, vogliamo parlare in termini generici.

Il concetto è simile a quello che succede in matematica, cioè per esempio quando ci chiediamo $x + 1 = 2$ e non conosciamo a priori x che deve poter variare sugli elementi della struttura. Quindi dovremmo chiederci per quali valori di x è vera l'equazione.

17.5 Ambiente: Assegnamento delle Variabili

Oltre alla semantica è necessario considerare un altro oggetto chiamato **assegnamento delle variabili** (oppure **ambiente**).

Definizione 17.5.1
Un assegnamento σ è rappresentato come una funzione

$$\sigma : X \to D$$

Dove X è un insieme infinito e D è il dominio. Proprio per questo motivo, non è possibile definire l'assegnamento se non vi è dominio della struttura. Nel nostro caso non avremo la necessità di rappresentare tutto σ, che potenzialmente potrebbe contenere infinite variabili, ma solo delle variabili della formula che considereremo.

Andremo ora quindi a dare:

1. Interpretazione ai termini di L
2. Interpretazione alle WFF di L

A noi ci interesserà 2, però per poter dare significato alle formule che in genere sono fatte da termini, daremo prima il significato (un'interpretazione) ai termini.

17.5.1 Relazione di Base

Per dare un significato a una formula avremo due oggetti:

- M = struttura (o modello)
- σ = assegnamento

$$M, \sigma \vDash \phi$$

Si legge come: ϕ è soddisfatto in M da σ

17.6 Interpretazione Termini di L

Definizione 17.6.1
Sia $t \in Term(L)$ e σ assegnamento,

$$[|t|]^M_\sigma \quad \text{denotazione di un termine (non ground)}$$

Sostanzialmente rappresenta l'oggetto del dominio che la semantica associa al termine t.

- **Caso Base**:
  - se $t = c \in C$ allora $[|t|]^M_\sigma = I(c)$ [la denotazione coincide con l'interpretazione]
  - se $t = x \in X$ allora $[|t|]^M_\sigma = \sigma(x)$ [l'elemento associato dall'assegnamento è l'interpretazione]

- **Caso Induttivo**:
  - se $t = f(t_1, ..., t_n)$
    con $f \in F$ di arietà n e $t_1, ..., t_n$ sono termini di L

    allora $[|t|]^M_\sigma = I(f)([|t_1|]^M_\sigma, ..., [|t_n|]^M_\sigma)$

    Dove:
    1. $I(f) : D^n \to D$
    2. $[|t_i|]^M_\sigma \in D$

    Allora $([|t_1|]^M_\sigma, ..., [|t_n|]^M_\sigma) \in D^n$

    In altre parole, la tupla $([|t_1|]^M_\sigma, ..., [|t_n|]^M_\sigma) \in D^n$ rappresenta l'interpretazione di ogni termine e viene data in input alla all'interpretazione di f, cioè $I(f)$, ottenendo un elemento del dominio D.

Esempio 17.6.1
Allora per interpretare $\overline{1}\dot{+}x$:

Utilizzeremo $M = (\mathbb{N}, I^{\mathbb{N}})$ e $\sigma(x) = 3$, ottenendo $[|\overline{1}\dot{+}x|]^M_\sigma = 4$

Se scegliessimo invece $\sigma'(x) = 5$ otterremmo $[|\overline{1}\dot{+}x|]^M_{\sigma'} = 6$

17.7 Interpretazione delle Formule WFF di L

Definizione 17.7.1
Dati M, σ e $\phi \in WFF(L)$

Definiremo:

$$M, \sigma \models \phi$$

Dove $\models$ è la relazione di soddisfazione. L'intuizione è che vale tale relazione se ϕ è vera (o soddisfatta) nel modello (o struttura) M. Faremo una definizione ricorsiva partendo dalle formule atomiche $P(t_1, ..., t_n)$ e $t_1 = t_2$.

- **Casi Base**: $M, \sigma \models \phi$
  - Se $\phi = P(t_1, ..., t_k)$
    con $P \in R$ di arietà k e $t_1, ..., t_k$ termini di L,
    allora avremo che:
    $$([|t_1|]^M_\sigma, ..., [|t_k|]^M_\sigma) \in I(P)$$
  - Se $\phi \overset{\text{metà ling.}}{=} t_1 = t_2$ allora:
    $$[|t_1|]^M_\sigma \overset{\text{rel. di ident.}}{=} [|t_2|]^M_\sigma$$
- **Casi Induttivi**:
  - Se $\phi = \neg\psi$ allora $\quad M, \sigma \nvDash \psi$

- Se $\phi = \psi_1 \wedge \psi_2$ allora $\quad M, \sigma \vDash \psi_1$ e $M, \sigma \vDash \psi_2$
- $\vdots$
- Se $\phi = \forall x.\psi$

Definizione 17.7.1 [Assegnamento per Sostituzione]
Dato $\sigma : X \to D$ sia $\sigma[x \leftarrow d]$ un nuovo assegnamento tale che:

$$\forall y \in X \quad \sigma[x \leftarrow d](y) = \begin{cases} d & \text{se } y = x \\ \sigma(y) & \text{altrimenti} \end{cases}$$

Questo nuovo assegnamento è uguale a σ per tutte le variabili, tranne per x poiché ad x assegna il valore d.

Allora **per ogni** $d \in D$

$$M, \sigma[x \leftarrow d] \vDash \psi$$

Noi vogliamo dire quando è vero questo: $M, \sigma \models \phi($ ma $\phi = \forall x.\psi)$, ma questo è vero se e solo se qualsiasi valore assegni ad x, quello che ottengo soddisfa ψ.

Esempio 17.7.1
Dati $M, \sigma \vDash \forall x.\psi$ allora

$$M, \sigma \vDash \forall x.P(x)$$

$$\text{se e solo se } \forall d \in D \quad d \in I(P)$$

Oppure equivalentemente possiamo dire che $I(P) = D$. In altre parole, tutti gli elementi del dominio devono avere la proprietà P e l'insieme P e D coincidono perché contengono gli stessi elementi.

- se $\phi = \exists x\ \psi$ allora $\quad$ **per qualche** $d \in D$ avremo che $M, \sigma[x \leftarrow d] \vDash \psi$

17.8 Soddisfacibilità in M

Definizione 17.8.1
Una formula ϕ è soddisfacibile in M se $\exists \sigma$ tale che $M, \sigma \vDash \phi$ (la definizione coincide con quella ovvia).

In altre parole, ϕ soddisfacibile in M se esistono M e σ tale che $M, \sigma \vDash \phi$ (cioè ϕ è soddisfatto in M da σ).

Esempio 17.8.1

$$x + \overline{1} = \overline{2}$$

É soddisfacibile nel modello dei numeri naturali perché esiste un $\sigma(x) = 1$. Allora $x + \overline{1} = \overline{2}$ è soddisfacibile in $M = (\mathbb{N}, I^{\mathbb{N}})$

17.9 Formula Vera

Definizione 17.9.1
Una formula si dice vera ($M \models \phi$) se e solo se $\forall \sigma \ M, \sigma \models \phi$

Esempio 17.9.1

$$\forall x.x = x$$

É sempre vera in $M = (\mathbb{N}, I^{\mathbb{N}})$ per qualsiasi ϕ, perché qualunque σ si scelga che assegni a x un elemento d, allora $d = d$ sarà sempre identico a se stesso (cioè vale la relazione d'identità). In realtà stiamo esprimendo un qualcosa di più forte valido in qualsiasi M.

17.10 Formula Valida

Definizione 17.10.1
Una formula ϕ è valida($\vDash \phi$) se per ogni struttura M, si ha che ϕ è vera in M.

Esempio 17.10.1

$$\forall x.x = x \quad \text{è anche formula valida}$$

17.11 Modello

Definizione 17.11.1
Quando $M \vDash \phi$ allora M è modello di ϕ

- Il modello del linguaggio è una qualsiasi struttura che interpreta i simboli del linguaggio
- Il modello di una formula è una struttura del linguaggio che rende vera quella formula

17.12 Conseguenza Logica

Definizione 17.12.1
$\Gamma \vDash \phi$ se e solo se $\forall M$ e $\forall \sigma$ vale che:

$$M, \sigma \vDash \Gamma \text{ allora } M, \sigma \vDash \phi$$

In altre parole ϕ è conseguenza logica (o semantica) di Γ, se e solo se per ogni struttura M e assegnamento σ vale che $M, \sigma \vDash \Gamma$ (cioè soddisfa ciascuna delle formule in Γ), allora $M, \sigma \vDash \phi$.

Osserviamo che stiamo facendo una quantificazione sia su tutti i modelli M sia su tutti gli assegnamenti σ.

17.13 Proprietà

Proprietà 17.13.1
Inoltre valgono le seguenti proprietà:

1. ϕ è valida se e solo se $\neg\phi$ è insoddisfacibile
2. ϕ è soddisfacibile se e solo se $\neg\phi$ non è valida
3. $\Gamma \models \phi$ (cioè ϕ è conseguenza logica di Γ) se e solo se $\Gamma \cup \neg\phi$ è insoddisfacibile
4. Se ϕ è soddisfacibile in M e ϕ è chiusa allora $M \vDash \phi$

Al punto 4 la soddisfacibilità in M richiede che $\exists\sigma, M_\sigma \vDash \phi$, mentre $M \vDash \phi$ richiede che $\forall\sigma, M_\sigma \vDash \phi$. In altre parole, la verità in un modello e la soddisfacibilità di una formula in quel modello coincidono.

Esempio 17.13.1

$$x \bar{+} \overline{1} = \overline{2}$$

É soddisfacibile nel *modello* $(\mathbb{N}, I^{\mathbb{N}})$ ma non è vera in $(\mathbb{N}, I^{\mathbb{N}})$

Esempio 17.13.2

$$\exists x.x \bar{+} \overline{1} = \overline{2}$$

É soddisfacibile nel *modello* $(\mathbb{N}, I^{\mathbb{N}})$, cioè esiste un assegnamento σ tale che

$$M, \sigma \vDash \exists x.x \bar{+} \overline{1} = \overline{2}$$

$$\text{cioè se e solo se } M, \sigma[x \leftarrow d] \vDash x \bar{+} \overline{1} = \overline{2} \text{ per qualche } d \in \mathbb{N}$$

Se scelgo d=1 soddisfa l'assegnamento σ, in realtà vale per qualsiasi σ che sostituisce ad x il valore $d = 1$. Inoltre, qualsiasi valore sia assegnato alle altre variabili non ha nessun impatto sulla formula poiché in questa formula non si parla di quelle variabili. Di conseguenza, avremo che questa formula è anche una formula vera in $(\mathbb{N}, I^{\mathbb{N}})$.

17.14 Equivalenza

Definizione 17.14.1
Due formule sono equivalenti ($\phi \equiv \psi$) se per ogni M e σ, vale che

$$M, \sigma \vDash \phi \text{ se e solo se } M, \sigma \vDash \psi$$

In altre parole, qualsiasi coppia M, σ assegna lo stesso valore di verità alle due formule (ϕ e ψ).

Sfruttando la semantica di $\rightarrow$ (implicazione) possiamo provare che:

$$\phi \equiv \psi \text{ se e solo se } \vDash \phi \leftrightarrow \psi \text{ se e solo se } \phi \vDash \psi \text{ e } \psi \vDash \phi$$

Questo significa che se due formule sono equivalenti ($\phi \equiv \psi$), se e solo se $\phi \leftrightarrow \psi$ è valida. Una formula è valida, se per ogni struttura M e assegnamento σ,

$$M, \sigma \models \phi \leftrightarrow \psi \overset{\text{semantica doppia implicazione}}{\iff} M, \sigma \models \phi \text{ se e solo se } M, \sigma \models \psi$$

$$\overset{\text{def conseguenza logica}}{\iff} \phi \models \psi \quad \text{e} \quad \psi \models \phi$$

17.15 Esercizi: Da Naturale a Primo Ordine

Come possiamo esprimere frasi in linguaggio naturale in logica del primo ordine.

1. "C'è un cane che abbaia"

   Decidiamo il linguaggio in cui le *costanti* sono i nomi dei cani.

   - Nel nostro dominio possiamo elementi che sono cani e elementi che non sono cani.
   - Dobbiamo ora definire una relazione (predicato) che ci permette di distinguere da cosa è cane e cosa non è cane.

   - $Cane \in R$ e arietà di $Cane = 1$
   - $Ab \in R$ ed arietà di $Ab = 1$

   Cioè la proprietà del cane è di abbaiare: $Cane(c)$ e $Ab(c)$, dove c è un $Cane$ che ha la proprietà Ab, cioè la proprietà di abbaiare. Possiamo esprimere la frase in logica del primo ordine come segue:

   $$\exists x.(Cane(x) \land Ab(x))$$

   In altre parole, esiste un elemento del dominio che è un cane che ha la proprietà di abbaiare.

2. "C'è un solo cane che abbaia"

   Questa frase può essere espressa come "C'è un cane che abbaia e questo cane è unico". In altre parole, bisogna esprimere l'unicità del cane.

   $$\exists x.(Cane(x) \land Ab(x) \land \underbrace{\forall y.(Cane(y) \land Ab(y)) \rightarrow x = y)))}_{\text{x è unico}}$$

3. "C'è un solo cane e abbaia"

   Questa frase può somigliare alla 2 in quanto c'è ancora un concetto di unicità che è legato solo alla proprietà di essere cane (e non di essere cane e di abbaiare).

   $$\exists x.(Cane(x) \land Ab(x) \land \underbrace{\forall y(Cane(y) \rightarrow x = y))}_{\text{x è unico}}$$

   Nella 2 il dominio poteva avere tanti cani ma ce ne era uno solo che abbaiava. In questa invece, c'è un unico cane e questo cane abbaia pure.

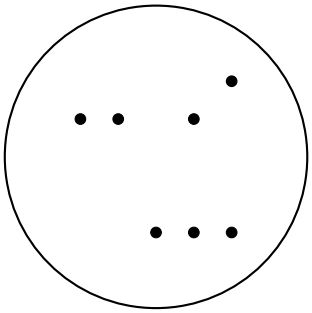
4. "C'è al massimo un cane"

   Nel dominio descritto da questa frase abbiamo che o c'è al massimo un cane ed è unico oppure non ci sono cani.

$$\forall x.\forall y.((Cane(x) \land Cane(y)) \rightarrow x = y)$$

   Possiamo osservare che il "C'è" non viene tradotto con $\exists$ ma con il $\forall$, perché sta quantificando su tutti gli elementi del dominio. Quello che vorremmo dire è che qualsiasi elemento del dominio io prenda o non è un cane o se è un cane, allora è l'unico cane. In altre parole, stiamo cercando di scartare tutti quei modelli in cui c'è più di un cane.

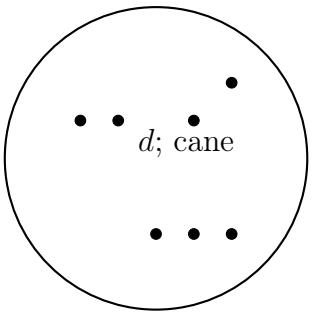
   Caso 1: Non ci sono cani



D

   Allora di conseguenza la premesse dell'implicazione $(Cane(x) \land Cane(y))$ sarà falsa quindi l'implicazione sarà vera soddisfacendo allora la formula.

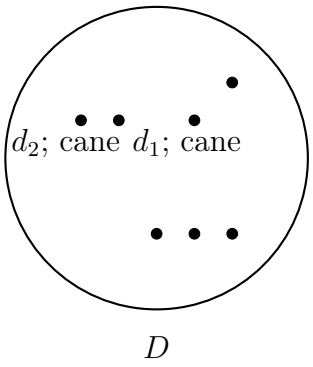
   Caso 2: C'è un solo cane (cioè $I(Cane) = \{d\}$)



D

   Allora di conseguenza, qualsiasi y scelga diverso da d, la premessa dell'implicazione sarà falsa (perché abbiamo un unico cane che è d) quindi allora l'implicazione sarà vera soddisfacendo la formula. Se invece scelgo un $y = d$, allora la premessa dell'implicazione sarà vera e anche la conseguenza sarà vera $(x = y)$ quindi l'implicazione sarà vera e soddisfarrò ancora la formula

Caso 3: Ci sono due (o più cani) cani $d1$ e $d2$ (cioè $I(Cane) = \{d_1, d_2\}$)



Allora di conseguenza avremo la premessa vera, ma la conseguenza dell'implicazione falsa ($d1 \neq d2$ e quindi l'implicazione sarà false non soddisfacendo la formula.

Possiamo notare che l'assegnamento soddisfa $Cane(d_1) \wedge Cane(d_2)$:

$$\sigma[x \leftarrow d_1][y \leftarrow d_2] \vDash Cane(x) \wedge Cane(y)$$

Ma possiamo osservare che invece non soddisfa $d_1 = d_2$:

$$M, \sigma[x \leftarrow d_1][y \leftarrow d_2] \nvDash x = y$$

Capitolo 18

Lezione 18

18.1 Esercizi: Da Naturale e Primo Ordine

1. "Ogni cane che abbaia è senza padrone"
   - Decidiamo i simboli per formalizzare
     C(·) notazione soddisfatta dal dominio dei cani
     A(·) sta per abbaia
     P(·, ·) in cui il primo argomento è padrone del secondo argomento

$$\forall x.((C(x) \land A(x)) \to \forall z.\neg P(z,x))$$

2. "I cani che hanno padrone non abbaiano"

$$\forall x.((C(x) \land \exists z.P(z,x)) \to \neg A(x))$$

Osservazione 18.1.1
Possiamo osservare che le formule 1 e 2 sono semanticamente uguali ma non sintatticamente.

Se considerassimo la struttura della frase 1 abbiamo:

- P = C(x)
- Q = A(x)
- R = ¬A(x)

Quindi avremo la seguente struttura:

$$(P \land Q) \to R \iff \neg(P \land Q) \lor R \iff (\neg P \lor \neg Q) \lor R$$

Quindi la formula 1 può essere riscritta come:

$$\forall x.(\neg C(x) \lor \neg A(x) \lor \forall z.\neg P(z,x))$$

La formula 2 può essere riscritta come:

$$\forall x.(\neg C(x) \lor \neg \exists z.P(z,x) \lor \neg A(x))$$

Notiamo che, le due frasi differiscono per $\neg\exists z.P(z,x)$ e $\forall z.\neg P(z,x)$ ma che in realtà risultano equivalenti.

3. "Ogni genitore ha due figli"

   - Decidiamo i predicati per formalizzare
     G(·) che rappresenta l'essere genitore
     F(·, ·) in cui il primo argomento è figlio del secondo

   Questa frase può essere interpretata in due modi:

   (a) "Ogni genitore ha almeno due figli"

   $$\forall x.(G(x) \to \exists y.\exists z.(\neg(y = z) \land F(y, x) \land F(z, x))$$

   Per semplicità potremmo indicare $\neg(y = z)$ con $(y \neq z)$.

   Oppure la frase può essere interpretata come:

   (b) "Ogni genitore ha solo due figli"

   $$\forall x.(G(x) \to \exists y.\exists z.(\neg(y = z) \land F(y, x) \land F(z, x) \land \forall k.(F(k, x) \to ((k = y) \lor (k = z)))))$$

   Se volessimo rappresentarla in forma normale avremo:

   $$\forall x.(G(x) \to \exists y.\exists z.(\neg(y = z) \land F(y, x) \land F(z, x) \land \forall k.(\neg F(k, x) \lor (k = y) \lor (k = z))))$$

4. "Ci sono esattamente due numeri primi minori di 5"

   N(·) individua i numeri
   P(·) individua i numeri primi
   $\cdot < \cdot$ notazione di minore (relazione binaria in forma infissa)
   5 simbolo costante

   $$\exists x.\exists y.(N(x) \land N(y) \land P(x) \land P(y) \land x < 5 \land y < 5 \land \neg(x = y) \land \forall z.((N(z) \land P(z) \land z < 5) \to (z = x \lor z = y)))$$

5. "Ci sono infiniti numeri primi"

   N(·) individua i numeri
   P(·) individua i numeri primi
   $\cdot < \cdot$ notazione di minore (relazione binaria in forma infissa)

   $$\forall x.((N(x) \land P(x)) \to \exists z.(N(z) \land P(z) \land x < z))$$

6. "Ci sono finiti numeri primi"

   N(·) individua i numeri
   P(·) individua i numeri primi
   $\cdot < \cdot$ notazione di minore (relazione binaria in forma infissa)

   $$\exists k.(N(k) \land \forall x.((N(x) \land P(x)) \to x < k))$$

Osservazione 18.1.2
Un ulteriore punto da tenere in considerazione è che non è possibile esprimere che nel dominio ci sono infiniti elementi o un finito numero di elementi. Ma quello che possiamo fare è dire che ce ne sono almeno o al più fissato un numero come segue nelle prossime frasi.

7. "Ci sono almeno k individui"

$$\underbrace{\exists x_1.\exists x_2.,\ldots,\exists x_k.}_{\exists x_1,\ldots,x_k}\left(\bigwedge_{i=1}^{k-1}\bigwedge_{j=i+1}^{k}\neg(x_i = x_j)\right) \triangleq \phi_{\geq k}$$

Se avessimo solo 3 elementi, otterremmo:

$$\exists x_1., \exists x_2., \exists x_3.(\neg(x1 = x2) \land \neg(x1 = x3) \land \neg(x2 = x3))$$

Possiamo osservare che le coppie vengono prese considerando che il primo elemento della coppia abbia pedice minore del secondo elemento della coppia.

8. "Ci sono al massimo k elementi"

$$\forall x_1., \forall x_2., \ldots, \forall x_k., \forall x_{k+1}. \left(\bigvee_{i=1}^{k} \bigvee_{j=i+1}^{k+1} (x_i = x_j) \right) \triangleq \phi_{\leq k}$$

Il concetto che vogliamo esprimere è che prendendo $k+1$ elementi, almeno una delle coppia è uguale.

9. "Ci sono esattamente k elementi"

$$\phi_{=k} \triangleq \phi_{\leq k} \land \phi_{\geq k}$$

18.2 Proprietà di due assegnamenti diversi ma uguali

Proprietà 18.2.1
Sia $t \in Term$ e $Var(t) = \{x_1, \ldots, x_n\}$ e siano σ_1 e σ_2 due assegnamenti tali che:

$$\sigma_1(x) = \sigma_2(x) \text{ per ogni } x \in Var(t)$$

In altre parole, σ_1 e σ_2 sono due assegnamenti diversi ma che assegnano gli stessi valori alle variabili considerate $\{x_1, \ldots, x_n\}$.

Allora per ogni modello $M(D, I)$:

$$[|t|]^M_{\sigma_1} = [|t|]^M_{\sigma_2}$$

In altre parole, prendo il termine t e lo interpreto con lo stesso modello utilizzando i due assegnamenti ottengo lo stesso elemento del dominio.

Dimostrazione 18.2.1
Dimostrazione per induzione strutturale sulla struttura del termine.

- **Caso Base**
  - Se $t = c$ (con c come costante) è ovvio perché $[|t|]^M_{\sigma_1} = I(t) = [|t|]^M_{\sigma_2}$
  - Se $t = x$ (con $x \in Var(t)$) è ovvio perché $[|t|]^M_{\sigma_1} = \sigma_1(x) = \sigma_2(x) = [|t|]^M_{\sigma_2}$
- **Caso Induttivo**
  - Se $t = f(t_1, \ldots, t_k) \implies I(f)([|t_1|]^M_{\sigma_1}, \ldots, [|t_k|]^M_{\sigma_1}) = [|t|]^M_{\sigma_1}$
    Per **Ipotesi Induttiva** perché t_1 e t_k sono più semplici di t, allora le variabili di t_1 e t_k saranno incluse in quelle di t.
    In generale per le variabili di un termine t_i avremo che $\text{Var}(t_i) \subseteq \text{Var}(t)$.
    Allora, applicando l'**ipotesi induttiva**, avremo che:

$$[|t_1|]^M_{\sigma_1} = [|t_1|]^M_{\sigma_2}, \ldots, [|t_k|]^M_{\sigma_1} = [|t_k|]^M_{\sigma_2}$$

E di conseguenza avremo che:

$$[|t|]^M_{\sigma_1} = I(f)([|t_1|]^M_{\sigma_1}, \ldots, [|t_k|]^M_{\sigma_1}) = I(f)([|t_1|]^M_{\sigma_2}, \ldots, [|t_k|]^M_{\sigma_2}) = [|t|]^M_{\sigma_2}$$

18.3 Teorema delle Formule

Teorema 18.3.1
Sia $\phi \in FOL$ e $FV(\phi) = \{x_1, \ldots, x_n\}$ e siano σ_1 e σ_2 assegnamenti tali che $\sigma_1(x) = \sigma_2(x)$ per ogni $x \in FV(\phi)$.

Allora per ogni $M = (D, I)$:

$$M, \sigma_1 \vDash \phi \text{ iff } M, \sigma_2 \vDash \phi$$

In altre parole, se mantengo il valore delle variabili $x_1, \ldots, x_n$ uguali usando σ_1 e prendendo $\sigma_2(x) = \sigma_1(x)$ otteniamo un assegnamento σ_2 che ha le stesse proprietà di soddisfazione per ϕ di quello di partenza.

18.4 Concetto di Sostituzione per Termini

Definizione 18.4.1
Sia $t \in Term$, $x \in Var$ e $t' \in term$, denoteremo con $t^x_{t'}$ il termine ottenuto da t rimpiazzando tutte le occorrenze di x con t'.

Esempio 18.4.1
Sia $t = f(g(x, y), h(z, x))$

- $t^x_{f(y)} = f(g(f(y), y), h(z, f(y)))$
- $t^z_y = f(g(x, y), h(y, x))$
- $t^k_{f(y)} = t$ (perché k non compare).

18.5 Concetto di Sostituzione per Formule

Definizione 18.5.1
Sia $\phi \in FOL$, $x \in Var$ e $t \in Term$, denoteremo con ϕ^x_t la formula ottenuta da ϕ rimpiazzando ogni occorrenza libera di x con t.

Esempio 18.5.1
Sia $\phi = \exists y.(R(x, y, x) \wedge \forall x.P(x, y))$

- $\phi^x_{f(z)} = \exists y.(R(f(z), y, f(z)) \wedge \forall x.P(x, y))$
- $\phi^y_c = \phi$ (perché non c'è y libera)

Capitolo 19

Lezione 19

19.1 Lemma di sostituzione

Teorema 19.1.1
Sia $t, t' \in Term$ e $x \in Var$

Allora per ogni struttura (o modello) $\forall M$ e assegnamento $\forall \sigma$ vale che:

$$\underbrace{[|t^x_{t'}|]^M_\sigma}_{sostituzione\ sintattica} = \underbrace{[|t|]^M_{\sigma[x \leftarrow [|t'|]^M_\sigma]}}_{sostituzione\ semantica}$$

In altre parole, quello che questo lemma dice è che l'interpretazione del termine t a cui sostituisco ad x il termine t' risulta essere uguale all'interpretazione del termine t nell'assegnamento che ottengo assegnando ad x l'interpretazione del termine t'.

Sostanzialmente fare la sostituzione a livello sintattico (cioè dentro il linguaggio) è uguale a fare una sostituzione semantica (cioè dentro l'assegnamento).

Dimostrazione 19.1.1
Dimostrazione per induzione sulla struttura sul concetto di termine.

- **Caso Base:** t è una costante o una variabile
  - Se $t \in C$ (costante) allora $t^x_{t'} = t$, allora $[|t^x_{t'}|]_\sigma = [|t|]_\sigma = I(t) = [|t|]_{\sigma[x \leftarrow [|t'|]_\sigma]}$
  - Se $t \in Var$ (variabile)
    1. $t \neq x \Rightarrow t^x_{t'} = t$, allora $[|t^x_{t'}|] = [|t|]_\sigma = \sigma(t) = \sigma[x \leftarrow [|t'|]]$
    2. $t = x \Rightarrow t^x_{t'} = t'$, allora $[|t^x_{t'}|]_\sigma = [|t'|]_\sigma = [|t|]_{\sigma[x \leftarrow [|t'|]_\sigma]}$
- **Caso Induttivo:**
  - $t = f(t_1, ..., t_k)$, allora $t^x_{t'} = f(t^x_{1t'}, ..., t^x_{kt'})$
    Ma t_1 e t_k sono più semplici di t, allora per **ipotesi induttiva** si ha che:

$$[|t^x_{it'}|]_\sigma = [|t_i|]_{\sigma[x \leftarrow [|t'|]_\sigma]}$$

Utilizzando l'annotazione $f^M \equiv I(f)$ avremo:

$$[|t'^x_{t'}|]_\sigma = f^M([|t^x_{1t'}|], ..., [|t^x_{kt'}|]) = f^M([|t_1|]_{\sigma[x \leftarrow [|t'|]_\sigma]}, ..., [|t_k|]_{\sigma[x \leftarrow [|t'|]_\sigma]}) = [|t|]_{\sigma[x \leftarrow [|t'|]_\sigma]}$$

19.2 Problema della Cattura: Vincolo alla Sostituzione

Utilizzando la sostituzione in questo modo non viene però preservata la soddisfacibilità (verità), denominato anche **problema della cattura**.

Esempio 19.2.1

$$\text{Sia } \phi : \forall x.P(x,y) \text{ e } t = f(x), \text{ allora } \phi^y_t = \forall x.P(x, f(x))$$

La formula aperta iniziale avente come variabile libera y, attraverso la sostituzione, diventa una formula chiusa che non ha variabili libere, poiché la sostituzione utilizzata senza nessuna vincolo non preserva la verità.

Esempio 19.2.2

Sia $\phi \triangleq \exists x.\ y < x$ e $t = x$, allora $\phi^y_t = \exists x.\ x < x$ non è più soddisfacibile nel modello considerato

Per questo motivo per preservare la verità è necessario aggiungere un **vincolo** sull'applicabilità della sostituzione.

Definizione 19.2.1
Sia t un termine e ϕ una formula,

t è libero per x in ϕ se nessuna delle variabili di t diventa legata dopo la sostituzione

Cioè se $Var(t)$ sono libere in ϕ^x_t .

Ricorda che con le formule si sostituiscono le occorrenze delle variabili libere (per il concetto di sostituzione delle formule).

Esempio 19.2.3
Nell'esempio precedente 19.2.2:

$$\text{Sia } \phi \triangleq \exists x.\ y < x \text{ e } t = x, \text{ allora } \phi^y_t = \exists x.\ x < x$$

Viene violato il vincolo perché y non è più variabile libera ma diventa legata in ϕ^y_t, cioè dopo la sostituzione.

19.3 Teorema di sostituzione

Teorema 19.3.1
Date ϕ, t, x, M e σ, se t è libera per x in ϕ allora

$$M, \sigma \vDash \phi_t^x \quad \text{se e solo se} \quad M, \sigma[x \leftarrow [|t|]_\sigma] \vDash \phi$$

In altre parole, l'interpretazione rispetto a σ della formula sostituita ϕ_t^x è identica all'interpretazione della formula originaria ϕ in cui la sostituzione la faccio per assegnamento.

Dimostrazione 19.3.1
Per induzione strutturale sulle formule.

- **Caso base:**
  Sia $\phi = P(t_1, ..., t_k)$, allora $\phi_t^x = P((t_1)_t^x, ..., (t_k)_t^x)$
  Ma $M, \sigma \vDash P((t_1)_t^x, ..., (t_k)_t^x) \iff ([|(t_1)_t^x|]_\sigma^M, ..., [|(t_k)_t^x|]_\sigma^M) \in P^M \overset{\text{Lemma di sostituzione}}{\iff}$

  $$([|(t_1)|]^M_{\sigma[x \leftarrow [|t|]^M_\sigma]}, \ \dots, \ [|(t_k)|]^M_{\sigma[x \leftarrow [|t|]^M_\sigma]}) \in P^M$$

  Che è equivalente a:

  $$M, \sigma[x \leftarrow [|t|]_\sigma^M] \models \phi$$

- **Caso Induttivo:**
  Sia $\phi = \forall y.\psi$, allora:

  1. $x = y$: allora $\phi_t^x = \phi$ perché x (cioè y) non è libera in ϕ e neanche in ϕ_t^x, allora:

     $$M, \sigma \vDash \phi_t^x \iff M, \sigma[x \leftarrow [|t|]_\sigma^M] \vDash \phi$$

     In altre parole, l'impatto della sostituzione sulle variabili non libere non ha impatto sulla soddisfacibilità.

  2. $x \neq y$: allora $\phi_t^x = \forall y.\psi_t^x$
     Per **Ipotesi Induttiva**, poiché ψ è più semplice di ϕ:

     $$M, \overline{\sigma} \vDash \psi_t^x \iff M, \overline{\sigma}[x \leftarrow [|t|]_\sigma^M] \vDash \psi$$

     Allora $M, \sigma \vDash \underbrace{\forall y.\psi_t^x}_{\phi_t^x}$

     Prendendo un arbitrario elemento del dominio d lo sostituisco al posto di y in σ:
     $\iff$ per ogni d : $M, \underbrace{\sigma[y \leftarrow d]}_{\overline{\sigma}} \vDash \psi_t^x$

     $$\iff \text{per ogni } d: \ M, \sigma[y \leftarrow d][x \leftarrow [|t|]^M_{\sigma[y \leftarrow d]}] \vDash \psi$$

     Data questa uguaglianza:

     $$[|t|]^M_{\sigma[y \leftarrow d]} = [|t|]^M_\sigma$$

     L'uguaglianza sussiste perché t è libera in ϕ da x (cioè nessuna $Var(t)$ è diventa legata) e quindi t non può contenere y.

Allora otteniamo:

$$M, \sigma[y \leftarrow d][x \leftarrow [|t|]^M_\sigma \models \psi$$

Invertendo i due assegnamenti avremo che:

$$\text{per ogni } d: \quad M, \sigma[x \leftarrow [|t|]^M_\sigma][y \leftarrow d] \models \psi$$

Ma possiamo osservare che quanto scritto è proprio equivalente alla semantica di:

$$M, \sigma[x \leftarrow [|t|]^M_\sigma] \models \underbrace{\forall y.\psi}_{\phi} \iff M, \sigma \models \underbrace{\forall y.\psi^x_t}_{\phi^x_t}$$

19.4 Validità dei Quantificatori

Proprietà 19.4.1

- Se $\models \phi$ e $\models \phi \to \psi$ allora $\models \psi$ che corrisponde al Modus Ponens.
  In altre parole, se ϕ è valida ed è valida $\phi \to \psi$, allora è valida anche ψ.

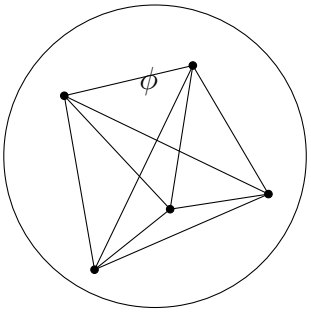
Sia $P(x)$ una qualsiasi formula:

- $\neg\exists x.P(x) \equiv \forall x.\neg P(x)$
- $\neg\forall x.P(x) \equiv \exists x.\neg P(x)$
- $\exists x.P(x) \equiv \neg\forall x.\neg P(x)$
- $\forall x.P(x) \equiv \neg\exists x.\neg P(x)$

19.5 Esercizi: Validità delle Formule

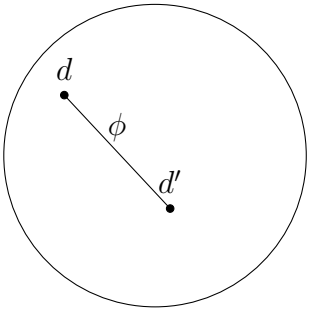
1. $\models \forall x.\forall y.\phi \leftrightarrow \forall y.\forall x.\phi$ è valida? Sì

   Per poter capire se la formula è valida, si prende un modello arbitrario che soddisfa la formula che è rappresentabile come un'insieme di elementi. In questo specifico caso gli elementi presi a coppia devono soddisfare la proprietà ϕ ed è possibile rappresentarli come archi non orientati. Di conseguenza, se sappiamo che ogni x e y a coppia soddisfano la proprietà, allora prendendoli a coppia ma invertendo l'ordine degli elementi si continuerà a soddisfare la proprietà.



2. $\vDash \exists x.\exists y.\phi \leftrightarrow \exists y.\exists x.\phi$ è valida? Sì.

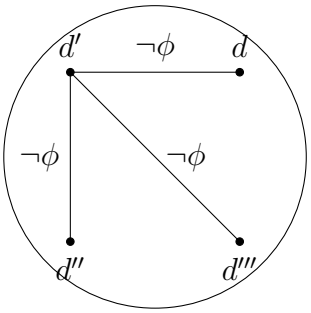
   In questo caso un modello prenderemo un modello in cui avremo una coppia di due elementi d e d' che soddisfa ϕ, allora invertendo l'ordine degli elementi continueranno a soddisfare la proprietà ϕ.



> **Osservazione 19.5.1**
> Scambiare quantificatore dello stesso tipo preserva la verità della formula.

3. $\vDash \exists x.\forall y.\phi \to \forall y.\exists x.\phi$ è valida? Sì.

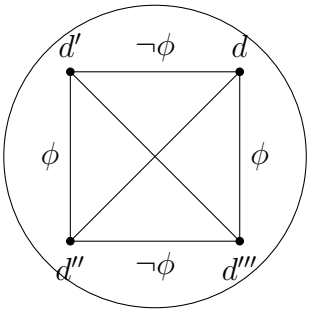
   Possiamo prendere un contro-modello per la premessa e verificheremo che sia un contro-modello anche per la conclusione. Sostanzialmente falsifichiamo la premessa, cioè prendiamo un elemento d' che con tutti gli altri elementi non verificano ϕ, cioè ottenendo $\neg\phi$. Possiamo anche osservare che anche contro-modello per la conclusione. Allora, possiamo quindi concludere che la formula è valida.



> **Osservazione 19.5.2**
> In generale per dimostrale la validità, dato che il dominio è infinito, è molto complicato in quanto è necessario che la validità sia preservata data l'infinità di configurazione e modelli possibili.
>
> Per questo motivo un'alternativa ragionevole è ragionerà per contrapposizione.

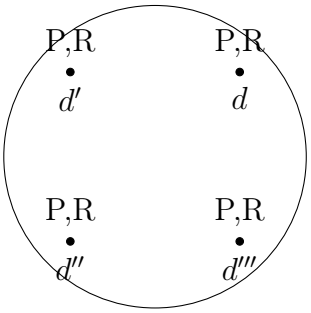
4. $\models \forall y.\exists x.\phi \to \exists x.\forall y.\phi$ è valida? No.

   Prendendo un modello, possiamo osservare che per ogni elemento y esiste un elemento x che soddisfa ϕ, ma non è vero che esiste un elemento x per cui ogni elemento y soddisfa ϕ.



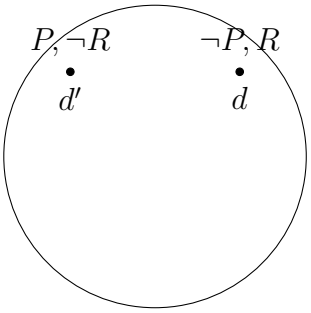
Osservazione 19.5.3
In generale per dimostrale la falsità (non validità) basta trovare un contro-modello.

5. $\forall x.(P(x) \wedge R(x)) \equiv \forall x.P(x) \wedge \forall x.R(x)$



6. $\forall x.(P(x) \vee R(x)) \not\equiv \forall x.P(x) \vee \forall x.R(x)$

   Prendendo questo modello, possiamo osservare che non è vero che tutti gli elementi soddisfano P o che tutti gli elementi soddisfano R.

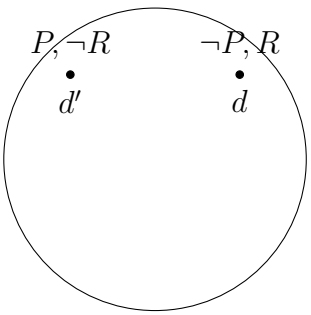


Osservazione 19.5.4
Possiamo notare che $\forall$ si comporta bene con $\wedge$ e male con $\vee$.

Viceversa, il $\exists$ si comporta bene con $\vee$ e male con $\wedge$.

7. $\exists x.(P(x) \vee R(x)) \equiv \exists x.P(x) \vee \exists x.R(x)$

8. $\exists x.(P(x) \wedge R(x)) \not\equiv \exists x.P(x) \wedge \exists x.R(x)$



   Questo contromodello soddisfa $\exists x.P(x) \wedge \exists x.R(x)$ ma non $\exists x.(P(x) \wedge R(x))$

19.6 Osservazione importante

Proposizione 19.6.1
Sia $\phi(y)$, con y variabile libera di ϕ, allora:

1. $\vDash \phi(y) \Rightarrow \vDash \forall y.\phi(y)$

   Se $\phi(y)$ è valida, allora è valida $\forall y.\phi(y)$.

   Per verificare la prima implicazione possiamo prendere un M arbitrario:

   $$(\forall \sigma : M, \sigma \vDash \phi(y)) \Rightarrow (\forall \sigma : M, \sigma \vDash \forall y.\phi(y))$$

   In altre parole, con σ al posto di y possiamo mettere qualsiasi cosa e quindi se mettendo qualsiasi cosa soddisfo ϕ, allora anche a destra stiamo dicendo la stessa cosa (cioè per ogni y soddisfo ϕ, visto che ad y posso mettere qualsiasi cosa, soddisfarò sempre).

2. $\phi(y) \nvDash \forall y.\phi(y)$

   Ma non è vero che $\forall y.\phi(y)$ è conseguenza logica di $\phi(y)$.

   Per quanto riguarda questa affermazione possiamo dimostrare che:

   $$\forall M, \sigma \qquad M, \sigma \vDash \phi(y) \Rightarrow M, \sigma \vDash \forall y.\phi(y)$$

   Questo è facilmente dimostrabile prendendo $\phi(y) = y < 5$ avremo che la parte sinistra dell'implicazione sarà vera, mentre la proposizione conseguente sarà falsa.

   In altre parole, le premesse agiscono da filtro su quale modello considerare e non possiamo assumere $\phi(y)$ e concludere generalizzando che questa proprietà valga per tutti. In altre parole, $\phi(y)$ può essere vera per alcuni valori di y nel modello, ma ciò non implica che $\phi(y)$ sia vera per tutti i possibili valori di y.

Capitolo 20

Lezione 20

20.1 Proprietà delle Fromule Chiuse

Proprietà 20.1.1
Se ϕ è una **formula chiusa** (cioè senza variabili libere), allora vale che:

$$M \vDash \phi \iff \forall\sigma \quad M, \sigma \vDash \phi \iff \exists\sigma \quad M, \sigma \vDash \phi$$

Questo significa che una formula chiusa ϕ è vera in un modello M (equivalentemente un modello M soddisfa la formula ϕ) se e solo se è vera per tutti gli assegnamenti σ, ma essendo ϕ chiusa, di conseguenza non avendo variabili libere, possiamo ridurci a dire che è vera per un unico arbitrario assegnamento σ (qualsiasi σ prenda è irrilevante).

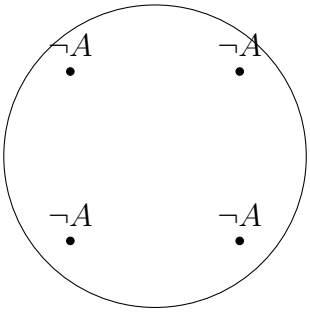
20.2 Esercizi: Validità delle Formule

Continuiamo gli esercizi della lezione precedente sulla validità di formule con quantificatori.

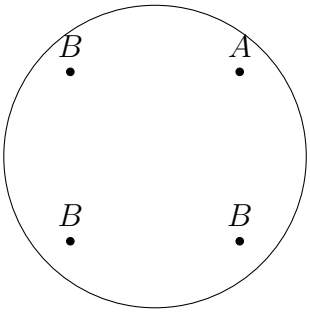
1. $\vDash \underbrace{(\exists x.A(x) \to \forall x.B(x))}_{premessa} \to \forall x.(A(x) \to B(x))$

   Caso 1: Considereremo i modelli che non hanno nessun elemento che soddisfa A e di conseguenza avremo che la prima implicazione non darà problemi poiché se la premessa è falsa avremo che la sua conclusione è vera qualunque sia il valore di verità di B, ottenendo che la premessa della formula è vera. In questo caso $\exists x.A(x) \to \forall x.B(x)$ è soddisfatta.

   Ora dobbiamo verificare che anche la conclusione della formula (cioè la seconda implicazione) è vera. Ma possiamo vedere che anche in questo caso avremo che $\forall x.(A(x) \to B(x))$ è soddisfatta perché la premessa è falsa (perché i modelli scelti non soddisfano A) e allora la conclusione è vera.

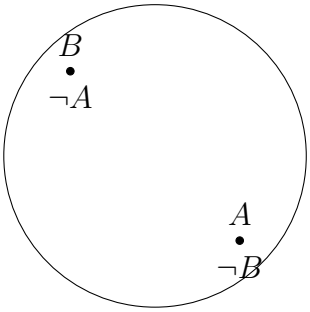


Caso 2: Considereremo i modelli che hanno almeno un elemento che soddisfa A e che B sia vero ovunque (per forza per via di A vero). Di conseguenza avremo che la premessa della formula (con premessa e conclusione vera) è vera e che la conclusione della formula (con premessa e conclusione vera) è vera.



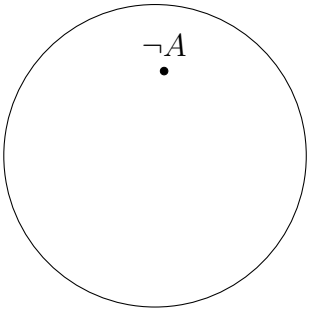
2. $\vDash \underbrace{(\exists x.A(x) \to \exists x.B(x))}_{premessa} \to (\forall x.(A(x) \to B(x))$

Possiamo osservare che abbiamo una premessa più debole rispetto alla formula precedente. Sappiamo che l'implicazione è falsa solo quando la premessa è vera e la conclusione è falsa, allora possiamo considerare un contro-modello come il seguente che soddisfa la premessa della formula, ma non soddisfa la conclusione della formula poiché abbiamo A e $\neg B$. Quindi possiamo concludere che allora la formula non è valida.

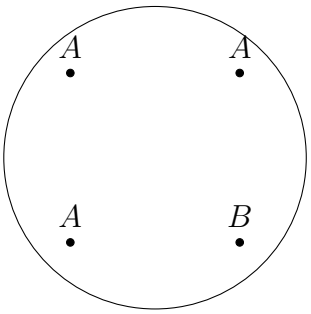


3. $\vDash \underbrace{\exists x.(A(x) \to B(x))}_{premessa} \to (\forall x.A(x) \to \exists x.B(x))$

Caso 1: Considereremo i modelli in cui A non è valida, ottenendo che la premessa della formula è sempre vera qualunque sia il valore di verità di B. La conseguenza della formula sarà anch'essa vera perché la sua premessa è falsa, di conseguenza sarà vera qualsiasi sia il valore di verità di B.



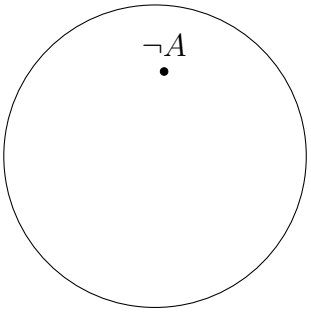
Caso 2: Considereremo i modelli in cui A è valida, ottenendo che la premessa della formula è vera se anche B è vero. La conseguenza della formula è vera perché la sua premessa è vera e la sua conseguenza è vera. Quindi possiamo concludere che allora la formula è valida.



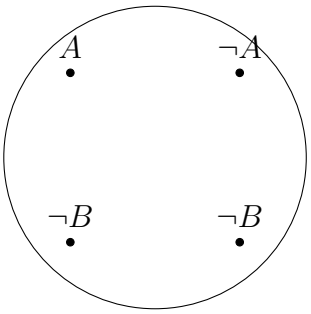
4. $\vDash \underbrace{\exists x.(A(x) \to B(x))}_{premessa} \to \underbrace{(\exists x.A(x) \to \exists x.B(x))}_{conclusione}$

Caso 1: Consideriamo i modelli c'è almeno un A non è valido di conseguenza la premessa della formula è soddisfatta qualsiasi sia il valore di verità di B. La conseguenza della formula è vera. Errore!.

L'idea è che avendo la premessa della conclusione meno restrittiva, si riduce meno il numero di modelli che posso scegliere e di conseguenza si rende più difficile rendere vera la conclusione. In altre parole, possiamo osservare che non si quantifica con il $\forall A(x)$ nella conclusione della formula, ovvero non è richiesto che per tutti gli A valga la premessa della conclusione.



Prendiamo un raffinamento di questo modello



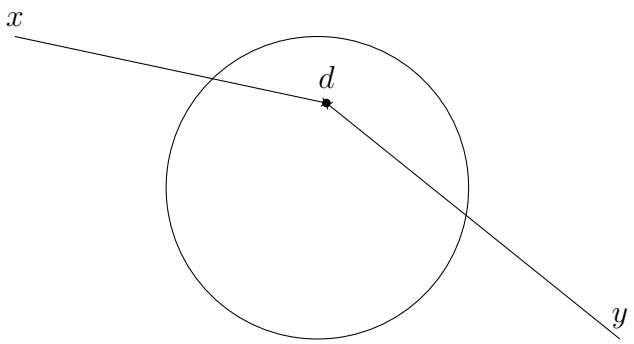
Considerando questo modello è possibile trovare un caso in cui possiamo verificare la premessa della formula considerando $\neg A$ e $\neg B$ e possiamo falsificare la conclusione considerando A e $\neg B$. Concluderemo affermando la non validità della formula.

5. $\vDash \neg \underbrace{\exists x.\forall y.(\neg R(x,x) \leftrightarrow R(x,y))}_{sotto-formula}$

Consideriamo solo la sotto-formula:

$$\exists x.\forall y.(\neg R(x,x) \leftrightarrow R(x,y))$$

Caso: Considerando il seguente modello



Vorremo che se esiste x, per ogni y valga $R(x,x)$ se solo se $R(x,y)$. Di conseguenza, fissato x, questo dovrà valere per tutti gli y del dominio, quindi anche quando $y = x$.

In modo informale avremo:

$$\neg R(d,d) \leftrightarrow R(d,d) \text{ con } d = x = y$$

$$\neg R(x,x) \leftrightarrow R(x, y = x)$$

In modo formale avremo:

$$(d,d) \notin R^M \iff (d,d) \in R^M$$

Possiamo osservare che la sotto-formula non è valida, cioè nessun modello la soddisfarà e di conseguenza soddisfaranno la sua negata (cioè la formula generale).

Osservazione 20.2.1
La formula generale $\phi = \neg\exists x.\forall y.(\neg R(x,x) \leftrightarrow R(x,y))$ può essere trasformata portando dentro la negazione, ricordandoci che la negazione del se solo se è lo *xor*:

$$\forall x.\exists y.((\neg R(x,x) \land \neg R(x,y)) \lor (R(x,x) \land R(x,y))) \text{ prendendo } x = y \text{ è soddisfacibile}$$

Un ulteriore modo di trasformarla è:

$$\forall x.\exists y.(R(x,x) \leftrightarrow R(x,y)) \text{ prendendo } x = y \text{ è soddisfacibile}$$

20.3 Teoria

Definizione 20.3.1
Una teoria è un insieme di formule che descrivono un particolare sottoinsieme di modelli. Queste formule sono degli assiomi, cioè formule che non sono assunzioni, ma delle asserzioni che prendiamo per vere, cioè sono vere in tutti i modelli.

Esempio 20.3.1
Una formula generica sarà vera in qualche modello e falsa in altri modelli.

$$\exists x.(P(x) \to \forall y.R(y))$$

20.3.1 Completezza di una Teoria di un Fenomeno

Definizione 20.3.2
Una Teoria è completa se e solo se $\forall \phi \in L$

$$\vDash_T \phi \quad \text{o} \quad \vDash_T \neg\phi$$

In altre parole, ogni formula o è conseguenza logica della teoria o lo è le sua negazione.

Quindi concludiamo avendo:

$$\vDash_T \phi \quad \text{o} \quad \vDash_T \neg\phi \quad = \quad \vdash_T \phi \quad \text{o} \quad \vdash_T \neg\phi$$

20.4 Assiomi del Fenomeno dell'Ordinamento

Proprietà 20.4.1
Sia $<$ simbolo del linguaggio L:

1. **Antisimmetria**: $\forall x.\neg(x < x)$
2. **Transitività**: $\forall x.\forall y.\forall z.((x < y \land y < z) \to x < z)$
3. **Antiriflessività**: $\forall x.\forall y.(x < y \to \neg(y < x))$
4. **Totalità:** $\forall x.\forall y.(x > y \lor y < x \lor x = y)$

   Osservazione 20.4.1

   - Se un ordinamento rispetta 1 e 2 avremo il domini infiniti, cioè 5 e lo garantisce 1.
   - Se un ordinamento rispetta 1, 2 e 3 sarà ordinamento parziale.
   - Se un ordinamento rispetta 1, 2, 3 e 4 sarà ordinamento lineare (o totale).

5. **Infinito:** $\forall x.\exists y.(x < y)$
   Possiamo osservare che senza le proprietà precedenti potrebbe presentarsi un loop che va contro la teoria dell'ordinamento.
   $$d < d' < d'' < d$$
   $d < d$ contraddice la 1
6. **Densità:** $\forall x.\forall y.(x < y \to \exists z.\ x < z \land z < y)$
7. **Discretezza:** $\forall x.(\exists y.(x < y) \to (\exists y.((x < y) \land \forall z.(x < z \to (z = y \lor (y < z))))))$

20.5 Assiomi dell'Uguaglianza

Proprietà 20.5.1
Come uguaglianza consideriamo la congruenza $\approx$ che è una relazione di equivalenza che rispetta funzioni e relazioni.

1. **Riflessività:** $\forall x. \quad x \approx x$
2. **Simmetria:** $\forall x.y. \quad x \approx y \to y \approx x$
3. **Transitività:** $\forall x.\forall y.\forall z.((x \approx y \land y \approx z) \to x \approx z)$
4. $(x_1 \approx y_1 \land \ldots \land x_n \approx y_n) \to f(x_1, \ldots, x_n) \approx f(y_1, \ldots, y_n)$
5. $((x_1 \approx y_1 \land \ldots \land x_n \approx y_n) \to (P(x_1, \ldots, x_n) \to P(y_1, \ldots, y_n)))$

Osservazione 20.5.1

- Se si rispetta 1, 2 e 3 sarà un'equivalenza.
- Se si rispetta 1, 2, 3, 4 e 5 sarà una congruenza.

Questa teoria non è completa perché non caratterizza in maniera precisa l'identità.

Esempio 20.5.1
I numeri 3 e 7 sono congruenti mod 4 ma non sono identici.

20.6 Regole di introduzione ai Quantificatori Universali

Definizione 20.6.1

$$\frac{\phi_y^x}{\forall x.\ \phi}\forall i. \quad \text{PER OGNI INTRODUCTION}$$

Dove ϕ_y^x significa prendere tutte le occorrenze libere di x e le sostituisci con y. L'idea è che non si può assumere la proprietà di un oggetto e poi farla diventare proprietà per tutti gli oggetti. Questa introduzione si può fare:

1. Se y non libera ne in alcuna assunzione non ancora scartata di ϕ_y^x
2. Se y non è libera in $\forall x.\ \phi$

$$\frac{\forall x.\phi}{\phi_t^x}\forall e \quad \text{PER OGNI ELIMINATION}$$

1. A patto che t sia libero (cioè le sue variabili) per x in ϕ, cioè nessuna variabile di t diventa legata in ϕ sostituendola con x.

L'idea è che se ϕ è una proprietà vera per tutti gli x, allora qualsiasi nome sostituissi ad x allora verrà soddisfatta.

Capitolo 21

Lezione 21

21.1 Esempio $\forall$ introduzione ed eliminazione

Esempio 21.1.1
Esempio di introduzione del $\forall$ che viola i vincoli:

$$\frac{\forall x.\ x = x}{z = z}\forall_e$$

Possiamo osservare che z è libera e non diventa legata dopo la sostituzione (ok, vincolo rispettato).

$$\frac{z = z}{\forall y.\ z = y}\forall_i$$

$$\frac{\forall y.\ z = y}{\forall x.\forall y.\ x = y}\forall_i$$

Possiamo osservare che da un qualcosa di valido, cioè $\forall x.\ x = x$, siamo arrivati a dedurre un qualcosa di non valido, cioè $\forall x.\forall y.\ x = y$. Questo è dovuto al fatto che non si è rispettato il vincolo nell'applicazione del $\forall_i$.

Andiamo ad analizzare il problema partendo dalla conclusione (cioè dalla seconda introduzione) e consideriamo la nostra formula: $\phi : x = y$ e con la sostituzione otteniamo $\phi^x_z : z = y$.

Risalendo $\phi : z = y$ ed effettuando la sostituzione abbiamo $\phi^y_z : z = z$. Ma dopo la sostituzione di y con z, otteniamo che z è libero. Nello specifico z deve essere non libero in $\forall y.\ \phi$ e non libero in ϕ^y_z (facendo attenzione alle variabili nell'applicazione della regola d'introduzione). In questo caso abbiamo che z è libero in $\forall y.\ z = y$ e quindi viola il vincolo (2).

La versione corretta dovrebbe essere questa:

$$\frac{\forall x.\ x = x}{z = z}\forall_e$$

$$\frac{z = z}{\forall y.\ y = y}\forall_i$$

21.2 $\exists$ introduzione ed eliminazione

Definizione 21.2.1

$$\frac{\phi_t^x}{\exists x.\ \phi}\exists_i \qquad \text{EXISTS INTRODUCTION}$$

1. Se t è libero per x in ϕ

$$\frac{\begin{array}{c}[\phi_y^x]\\ \vdots \\ \psi \end{array} \quad \exists x.\ \phi}{\psi}\exists_e \qquad \text{EXISTS ELIMINATION}$$

1. Se y non è libera in assunzioni non ancora scartate di ψ in ψ
2. Se y non è libera in $\exists x.\phi$

Si vuole l'arbitrarietà di y, questo si vuole anche nel $\forall_i$

Esempio 21.2.1

$$\frac{\forall x.x}{z=z}\forall_e$$
$$\frac{z=z}{\exists y.\ z=y}\exists_i$$
$$\frac{\exists y.\ z=y}{\exists x\exists y\ (x=y)}\exists_i$$

Per fare la verifica della correttezza della conclusione abbiamo $\phi : x = y$ e con la sostituzione abbiamo $\phi_x^z : (z = y)$.

Risalendo, abbiamo che $\phi : z = y$ e con la sostituzione abbiamo $\phi_z^y : (z = z)$ (ok, z è libero in ϕ).

21.3 Esempi di deduzione

Esercizio 21.3.1 $\forall x.(\psi \to R(x)) \to (\psi \to \forall x.R(x))$

Assumendo x non occorra libera in ψ
Prendiamo le assunzioni che sono $\forall x.(\psi \to R(x))$ e ψ:

$$\dfrac{[\forall x.(\psi \to R(x))]_2}{\psi \to R(y)}\forall_e$$

$$\dfrac{\psi \to R(y) \qquad [\psi]_1}{R(y)} \to_e$$

$$\dfrac{R(y)}{\forall x.R(x)}\forall_i$$

$$\dfrac{\forall x.R(x)}{\psi \to \forall x.R(x)} \to_{i1}$$

$$\dfrac{\psi \to \forall x.R(x)}{\forall x.(\psi \to R(x)) \to (\psi \to \forall x.R(x))} \to_{i2}$$

Esercizio 21.3.2 $\forall x.(R(x) \to \psi) \to ((\forall x.R(x)) \to \psi)$
Assumendo sempre che x non occorra libera in ψ
Abbiamo le seguenti assunzioni $\forall x.(R(x) \to \psi)$ e $\forall x.R(x)$:

$$\dfrac{[\forall x.(R(x) \to \psi)]_2}{R(z) \to \psi}\forall_e \qquad \dfrac{[\forall x.R(x)]_1}{R(z)}\forall_e$$

$$\dfrac{R(z) \to \psi \qquad R(z)}{\psi} \to_e$$

$$\dfrac{\psi}{(\forall x.R(x)) \to \psi} \to_{i1}$$

$$\dfrac{(\forall x.R(x)) \to \psi}{\forall x.(R(x) \to \psi) \to ((\forall x.R(x)) \to \psi)} \to_{i2}$$

Esercizio 21.3.3

$\forall x.(A(x) \wedge B(x)) \leftrightarrow (\forall x.A(x) \wedge \forall x.B(x))$

- $\forall x.(A(x) \wedge B(x)) \to (\forall x.A(x) \wedge \forall x.B(x))$
  Abbiamo le seguente assunzione $\forall x.(A(x) \wedge B(x))$

$$\dfrac{[\forall x.(A(x) \wedge B(x))]_1}{A(z) \wedge B(z)}\forall_e \qquad \dfrac{[\forall x.(A(x) \wedge B(x))]_1}{A(z) \wedge B(z)}\forall_e$$

$$\dfrac{A(z) \wedge B(z)}{A(z)} \wedge_{el} \qquad \dfrac{A(z) \wedge B(z)}{B(z)} \wedge_{er}$$

$$\dfrac{A(z)}{\forall x.A(x)}\forall_i \qquad \dfrac{B(z)}{\forall x.B(x)}\forall_i$$

$$\dfrac{\dfrac{\forall x.A(x) \qquad \forall x.B(x)}{\forall x.A(x) \wedge \forall x.B(x)}\wedge_i}{\forall x.(A(x) \wedge B(x)) \to (\forall x.A(x) \wedge \forall x.B(x))} \to_{i1}$$

- $\forall x.(A(x) \wedge B(x)) \leftarrow (\forall x.A(x) \wedge \forall x.B(x))$

$$\dfrac{\dfrac{\dfrac{[\forall x.A(x) \wedge \forall x.B(x)]_1}{A(z) \wedge B(z)}\forall_e}{A(z)}\wedge_{el} \qquad \dfrac{\dfrac{[\forall x.A(x) \wedge \forall x.B(x)]_1}{A(z) \wedge B(z)}\forall_e}{B(z)}\wedge_{er}}{\dfrac{\dfrac{A(z) \qquad B(z)}{A(z) \wedge B(z)}\wedge_i}{\dfrac{A(z) \wedge B(z)}{\forall x.A(x) \wedge B(x)}\forall_i}}$$

$$\dfrac{\forall x.A(x) \wedge B(x)}{(\forall x.A(x) \wedge \forall x.B(x)) \to \forall x.(A(x) \wedge B(x))} \to_{i1}$$

Esercizio 21.3.4 $\exists x.(P(x) \vee R(x)) \leftrightarrow (\exists x.P(x) \vee \exists x.R(x))$

- $\exists x.(P(x) \vee R(x)) \to (\exists x.P(x) \vee \exists x.R(x))$

$$\dfrac{\dfrac{\dfrac{[P(y)]_1}{\exists x.P(x)}\exists_i}{\exists x.P(x) \vee \exists x.R(x)}\vee_{il} \qquad \dfrac{\dfrac{[R(y)]_2}{\exists x.R(x)}\exists_i}{\exists x.P(x) \vee \exists x.R(x)}\vee_{ir}}{\dfrac{\dfrac{\exists x.P(x) \vee \exists x.R(x) \qquad \exists x.P(x) \vee \exists x.R(x) \qquad [P(y) \vee R(y)]_3}{\exists x.P(x) \vee \exists x.R(x)}\vee_{e12}}{\dfrac{\exists x.P(x) \vee \exists x.R(x) \qquad [\exists x(P(x) \vee R(x))]_4}{\exists x.P(x) \vee \exists x.R(x)}\exists_{e3}}}$$

$$\dfrac{\exists x.P(x) \vee \exists x.R(x)}{\exists x.(P(x) \vee R(x)) \to (\exists x.P(x) \vee \exists x.R(x))} \to_{i4}$$

- $\exists x.(P(x) \vee R(x)) \leftarrow (\exists x.P(x) \vee \exists x.R(x))$

$$\dfrac{\dfrac{\dfrac{\dfrac{[P(y)]_1}{P(y) \vee R(y)}\vee_i}{\exists x.(P(y) \vee R(y))}\exists_i \qquad \dfrac{\dfrac{[R(y)]_2}{P(y) \vee R(y)}\vee_i}{\exists x.(P(y) \vee R(y))}\exists_i}{\exists x.(P(y) \vee R(y)) \qquad [\exists x.P(x)]_3}\exists_{e2} \qquad \dfrac{\exists x.(P(y) \vee R(y)) \qquad [\exists x.P(x)]_4}{\exists x.(P(y) \vee R(y))}\exists_{e2}$$

$$\dfrac{\dfrac{\exists x.(P(y) \vee R(y)) \qquad \exists x.(P(y) \vee R(y)) \qquad [\exists x.P(x) \vee \exists x.R(x)]_5}{\exists x(P(x) \vee R(x))} \vee_{e3,4}}{(\exists x.P(x) \vee \exists x.R(x)) \to (\exists x.(P(x) \vee R(x)))} \to_{i5}$$

Capitolo 22

Lezione 22

22.1 Proprietà Fondamentali dei Quantificatori

In questa lezione, esamineremo alcune proprietà fondamentali sui quantificatori.

Proprietà 22.1.1

$$\forall x.P(x) \equiv \forall y.P(y)$$

Questa espressione afferma che se una proprietà $P(x)$ vale per tutti gli x, allora la stessa proprietà $P(y)$ vale per tutti gli y. Questo è intuitivo perché la scelta del nome della variabile di quantificazione non altera il significato dell'affermazione.

Osservazione 22.1.1
Consideriamo ora un modello M e un'assegnazione σ. Se il modello e l'assegnazione soddisfano la proprietà universale per tutte le x, scriveremo:

$$M, \sigma \vDash \forall x.P(x)$$

Questo significa che, per ogni elemento d nel dominio, la proprietà $P(x)$ è vera quando x è assegnato a d, riscrivibile anche come:

$$M, \sigma_{[x \leftarrow d]} \vDash P(x) \quad \forall d$$

Poiché la scelta del nome della variabile è irrilevante, abbiamo anche che:

$$M, \sigma_{[x \leftarrow d]} \vDash P(y) \quad \forall d$$

A questo punto, consideriamo un'altra espressione quantificata:

$$\forall x.P(x, y) \not\equiv \forall y.P(y, y)$$

Qui notiamo che i legami tra le variabili sono diversi. Mentre la prima affermazione riguarda una proprietà P che dipende sia da x che da y, la seconda riguarda una proprietà P in cui entrambe le variabili sono y. Questo cambiamento può alterare il significato dell'affermazione e quindi le due espressioni non sono logicamente equivalenti.

Ora, torniamo alla nostra affermazione iniziale:

$$\forall x.P(x) \equiv \forall y.P(y)$$

Se vale allora, utilizzando un ragionamento analogo, possiamo affermare che vale anche:

$$\exists x.P(x) \equiv \exists y.P(y)$$

In altre parole, se una proprietà $P(x)$ esiste per qualche x, allora esiste anche per qualche y, e viceversa. Questo perché, ancora una volta, il nome della variabile di quantificazione non cambia il significato dell'espressione esistenziale.

Adesso dimostriamo che vale la formula

$$\forall x.P(x) \equiv \forall y.P(y)$$

Esercizio 22.1.1

$$\forall x.P(x) \to \forall y.P(y)$$

L'assunzione che bisogna fare è che y deve essere libera per x in $\phi = P(x)$ per applicare il $\forall_e$

$$\frac{\forall x.P(x)}{P(y)}\forall e$$

$$\frac{P(y)}{\forall y.P(y)}\forall i$$

$$\frac{\forall y.P(y)}{\forall x.P(x) \to \forall y.P(y)} \to i$$

L'altro verso della dimostrazione è simile.

Esercizio 22.1.2

$$\exists x.P(x) \to \exists y.P(y)$$

L'assunzione che bisogna fare è $\exists x.P(x)$, ma per applicare $\exists_e$ partiremo da $P(y)$.

$$\frac{[P(y)]_1}{\exists y.P(y)}\exists i$$

$$\frac{[\exists x.P(x)]_2 \quad \exists y.P(y)}{\exists y.P(y)}\exists e_1$$

$$\frac{\exists y.P(y)}{\exists x.P(x) \to \exists y.P(y)} \to_{i_2}$$

Definizione 22.1.1
Le formule appena dimostrate possono essere generalizzate con:

$$Qx.\ \phi \equiv Qy.\ \phi_y^x$$

Dove $Q \in \{\forall, \exists\}$ e y non occorre proprio in ϕ.

Esercizio 22.1.3

$$\exists x \forall y. P(x,y) \vdash \forall y \exists x. P(x,y)$$

In questo caso possiamo assumere $\exists x \forall y. P(x,y)$, ma per lo stesso motivo precedente dobbiamo effettuare $\exists_e$ partendo da un'altra assunzione.

$$\frac{[\forall y. P(x,y)]_1}{P(x,y)} \forall e$$

$$\frac{P(x,y)}{\exists x. P(x,y)} \exists i$$

$$\frac{\exists x. P(x,y)}{\forall y \exists x. P(x,y)} \forall i$$

$$\frac{[\exists x \forall y. P(x,y)]_2 \quad \forall y \exists x. P(x,y)}{\forall y \exists x. P(x,y)} \exists e_1$$

$$\frac{\forall y \exists x. P(x,y)}{\exists x \forall y. P(x,y) \to \forall y \exists x. P(x,y)} \to i_2$$

Esercizio 22.1.4

$$\exists x. P(x) \to \forall x. P(x) \vdash \forall x (P(x) \to R(x))$$

Assumendo $\exists x. P(x) \to \forall x. P(x)$ su cui dovremo applicare $\to_e$ e quindi assumeremo anche $P(z)$.

$$\frac{[P(z)]_1}{\exists x. P(x)} \exists i$$

$$\frac{[\exists x. P(x) \to \forall x. R(x)]_2 \quad \exists x. P(x)}{\forall x. R(x)} \to e$$

$$\frac{\forall x. R(x)}{R(z)} \forall e$$

$$\frac{R(z)}{P(z) \to R(z)} \to i_1$$

$$\frac{P(z) \to R(z)}{\forall x.(P(x) \to R(x))} \forall i$$

$$\frac{\forall x.(P(x) \to R(x))}{(\exists x.P(x) \to \forall x.P(x)) \to (\forall x(P(x) \to R(x)))} \to i_2$$

Esercizio 22.1.5

$$\exists x.\neg P(x) \leftrightarrow \neg \forall x.P(x)$$

Dimostriamo prima il verso $\to$

$$\exists x.\neg P(x) \to \neg \forall x.P(x)$$

Assumiamo $\exists x.\neg P(x)$ su cui utilizzeremo $\exists_e$, ma prima assumiamo per assurdo $\forall x.P(x)$.

$$\frac{[\forall x.P(x)]_1}{P(z)} \forall e$$

Adesso assumiamo $\neg P(z)$ per ricavare $\bot$

$$\frac{[\neg P(z)]_2 \quad P(z)}{\bot} \bot i$$

$$\frac{\bot}{\neg \forall x.P(x)} \neg i_1$$

$$\frac{[\exists x.\neg P(x)]_3 \quad \neg \forall x.P(x)}{\neg \forall x.P(x)} \exists e_2$$

$$\frac{\neg \forall x.P(x)}{\exists x.\neg P(x) \to \neg \forall x.P(x)} \to i_3$$

Dimostriamo l'altro verso $\leftarrow$

$$\neg \forall x.P(x) \to \exists x.\neg P(x)$$

$$\frac{[\neg P(z)]_1}{\exists x.\neg P(x)} \exists i$$

$$\frac{\exists x.\neg P(x) \quad [\neg \exists x.\neg P(x)]_2}{\bot} \neg e$$

$$\frac{\bot}{P(z)} R.A.A_1$$

$$\frac{P(z)}{\forall x.P(x)} \forall i$$

$$\frac{[\neg\forall x.P(x)]_3 \quad \forall x.P(x)}{\bot}\bot i$$

$$\frac{\bot}{\exists x.\neg P(x)}R.A.A_2$$

$$\frac{\exists x.\neg P(x)}{\neg\forall x.P(x) \to \exists x.\neg P(x)} \to i_3$$

Da ciò concludiamo che vale

$$\exists x.\neg P(x) \leftrightarrow \neg\forall x.P(x)$$

Esercizio 22.1.6

$$\forall x.P(x) \leftrightarrow \neg\exists x.\neg P(x)$$

Dimostriamo prima il verso $\leftarrow$

$$\forall x.P(x) \leftarrow \neg\exists x.\neg P(x)$$

$$\frac{[\neg P(z)]_2}{\exists x.\neg P(x)}\exists i$$

$$\frac{[\neg\exists x.\neg P(x)]_1 \quad \exists x.\neg P(x)}{\bot}\neg e$$

$$\frac{\bot}{P(z)}R.A.A_2$$

$$\frac{P(z)}{\forall x.P(x)}\forall i$$

$$\frac{\forall x.P(x)}{\forall x.P(x) \to \neg\exists x.\neg P(x)} \leftarrow i_1$$

Dimostriamo il verso $\rightarrow$

$$\forall x.P(x) \to \neg\exists x.\neg P(x)$$

$$\frac{[\forall x.P(x)]_1}{P(z)}\forall_e$$

$$\frac{P(z) \quad [\neg P(z)]_3}{\bot}\bot i_1$$

$$\frac{\bot \quad [\exists x.\neg P(x)]_2}{\bot}\exists e_3$$

$$\frac{\bot}{\neg\exists x.\neg P(x)}\neg i_2$$

$$\frac{\neg\exists x.\neg P(x)}{\forall x.P(x)\to\neg\exists x.\neg P(x)}\to i_1$$

Esercizio da fare per casa: $\vdash \exists x\exists y.(P(x,y)\to P(y,x))$

Esercizio 22.1.7

$$\exists x.(P(x)\wedge Q(x)),\neg\exists x.(Q(x)\wedge R(x))\vdash\exists x.(P(x)\wedge\neg R(x))$$

$$\frac{[P(z)\wedge Q(z)]_3}{Q(z)}\wedge e_r$$

$$\frac{[R(z)]_1\quad Q(z)}{Q(z)\wedge R(z)}\wedge i$$

$$\frac{Q(z)\wedge R(z)}{\exists x.(Q(x)\wedge R(x))}\exists i$$

$$\frac{\neg\exists x.(Q(x)\wedge R(x))\quad\exists x.(Q(x)\wedge R(x))}{\bot}\bot i$$

$$\frac{\bot}{\neg R(z)}\neg i_1$$

Mettiamo in sospeso questo ramo per effettuare successivamente $\wedge_i$. Lavoriamo sull'altro ramo, assumendo $P(z)\wedge Q(z)$.

$$\frac{[P(z)\wedge Q(z)]}{P(z)}\wedge e_l$$

Riprendiamo quanto messo in sospeso e applichiamo $\wedge_i$.

$$\frac{P(z)\quad\neg R(z)}{P(z)\wedge\neg R(z)}\wedge_i$$

$$\frac{P(z)\wedge\neg R(z)}{\exists x.(P(x)\wedge\neg R(x))}\exists i$$

$$\frac{\exists x.(P(x)\wedge Q(x))\quad\exists x.(P(x)\wedge\neg R(x))}{\exists x.(P(x)\wedge\neg R(x))}\exists e_2$$

$$\frac{(\exists x.(P(x)\wedge\neg R(x))}{\exists x.(P(x)\wedge Q(x)),\neg\exists x.(Q(x)\wedge R(x)))\to(\exists x.(P(x)\wedge\neg R(x)))}$$

22.2 Regole per l'Uguaglianza

Definizione 22.2.1
Introduciamo delle regole senza premessa.

1. **Riflessività:**
$$\frac{}{t = t} RFL$$
Equivalente nel mondo assiomatico a $\forall x.x = x$

2. **Sub t:**
$$\frac{t_1 = s_1 \quad t_2 = s_2 \dots t_k = s_k}{(t)^{x_1 \dots x_k}_{t_1 \dots t_k} = (t)^{x_1 \dots x_k}_{s_1 \dots s_k}}$$
$$(t)^{x_1 \dots x_k}_{t_1 \dots t_k} = (t)^{x_1 \dots x_k}_{s_1 \dots s_k}$$
Rappresenta l'abbreviazione di:
$$\left(\left(\left(t^{x_1}_{t_1}\right)^{x_2}_{t_2}\right)\dots\right)^{x_k}_{t_k} = t^{x_1 \dots x_k}_{t_1 \dots t_k}$$

3. **Sub ϕ:**
$$\frac{t_1 = s_1 \dots t_k = s_k}{\phi^{x_1 \dots x_k}_{t_1 \dots t_k} \to \phi^{x_1 \dots x_k}_{s_1 \dots s_k}}$$

Capitolo 23

Lezione 23

23.1 Analisi della Validità

Partiamo da una formula su cui vogliamo capire perché non è valida.

$$\exists x.\neg P(x) \equiv \neg\exists x.P(x)$$

Ci si aspetta che questa equivalenza non sia valida.

Esempio 23.1.1
Per assurdo, supponiamo che valga l'equivalenza, allora di conseguenza il calcolo non sarebbe corretto (o incompleto) in quanto ci permette di dedurre cose non corrette.

Dimostriamo il verso $\leftarrow$

$$\exists x.\neg P(x) \leftarrow \neg\exists x.P(x)$$

Questa formula è valida ma può essere falsificata se non ci sono elementi nel modello. Ma per definizione un modello deve avere almeno un elemento. Altrimenti tutte le formule con $\exists$ sarebbero false e con $\forall$ vere.

$$\frac{[\neg P(y)]_1}{\exists x.\neg P(x)}\exists i$$

$$\frac{\exists x.\neg P(x) \quad [\neg\exists x\neg P(x)]_2}{\bot}\neg e$$

$$\frac{\bot}{P(y)}R.A.A_1$$

$$\frac{P(y)}{\exists x.P(x)}\exists i$$

$$\frac{\exists x.P(x) \quad \neg\exists x.P(x)}{\bot}\neg e$$

$$\frac{\bot}{\exists x.\neg P(x)}R.A.A_2$$

Adesso cerchiamo di dimostrare il verso $\rightarrow$

$$\exists x.\neg P(x) \rightarrow \neg\exists x.P(x)$$

Questa non dovrebbe essere valida.

$$\dfrac{[P(y)]_1 \quad [\neg P(y)]_2}{\bot}\neg e$$

$$\dfrac{\bot \quad [\exists x.P(x)]_3}{\bot}\exists e_1$$

$$\dfrac{\exists x \neg P(x) \quad \bot}{\bot}\exists e_2$$

$$\dfrac{\bot}{\neg\exists x.P(x)}\neg i_3$$

La dimostrazione risulta corretta sebbene non lo debba essere. Per inquadrare il problema diamo un'occhiata a tutte le regole utilizzate. Le uniche che potrebbero non essere valide sono $\exists e_1$ e $\exists e_2$ in quanto sono le uniche che hanno dei vincoli per essere applicate.

In particolare, $\exists e$ è applicabile quando la variabile corrente (cioè quella sostituita ad x, ovvero y) non deve occorrere libera in assunzioni non scartate di ψ (cioè di $\bot$) e non deve occorrere libera in $\exists x.\phi$ (cioè $\exists x \neg P(x)$). In questo caso, $\exists e_1$ è sbagliata poiché y è libera in $[\neg P(y)]_2$ e non è stata ancora scaricata.

23.2 Esempi di Deduzione con Uguaglianza

23.2.1 Riflessività

Esercizio 23.2.1

$$\forall x.x = x$$

$$\dfrac{}{y = y}RFL$$

$$\dfrac{y = y}{\forall x.x = x}\forall i$$

Ricordiamo che per utilizzare la regola $\forall i$ la variabile y, sostituita all'argomento x, non deve occorrere libera in alcuna assunzione non scartata (ma in questo caso non ce ne sono, quindi ok) e nè nella conclusione (cioè nel $\forall x.\phi$, ma ok).

Esercizio 23.2.2

$$\exists x.\exists y.x = y$$

$$\dfrac{}{a = a}RFL$$

$$\frac{a = a}{\exists y.a = y}\exists i$$

Per applicare $\exists i$ dobbiamo avere che t deve essere libero in $\exists.\phi$. In questo specifico caso abbiamo che $\phi_t^x = \phi_a^x : a = a$ e quindi $t = a$, di conseguenza vogliamo che a sia libero in $\exists.\phi = \exists y.a = y$ (quindi ok).

$$\frac{\exists y.a = y}{\exists x.\exists y.\ x = y}\exists i$$

Per applicare $\exists i$ dobbiamo avere che t deve essere libero in $\exists.\phi$. In questo specifico caso abbiamo che $\phi_t^x = \phi_a^x : \exists y.a = y$ e quindi $t = a$, di conseguenza vogliamo che a sia libero in $\exists x.\exists y.\ x = y$ (quindi ok).

Esercizio 23.2.3

$$P(t_1), \neg P(t_2) \vdash \neg(t_1 = t_2)$$

Assunzioni: $t_1 = t_2$ per scaricare tramite $\neg(t_1 = t_2)$

$$\frac{[t_1 = t_2]_1}{P(t_1) \to P(t_2)}SUB_\phi$$

$$\frac{P(t_1) \to P(t_2) \quad P(t_1)}{P(t_2)} \to e$$

$$\frac{P(t_2) \quad \neg P(t_2)}{\bot}\neg e$$

$$\frac{\bot}{\neg(t_1 = t_2)}\neg i_1$$

23.2.2 Simmetria

Esercizio 23.2.4

$$x = y \vdash y = x$$

Assunzioni: $x = y$ e $\frac{}{x=x}RFL$.

Considerando $t_1 = x$, $s_1 = y$ e $t_2 = x$, $s_2 = x$, avremo che:

$$\frac{[x = y]_1 \quad \frac{}{x=x}RFL}{x = x \to y = x}SUB_\phi$$

$$\frac{\frac{}{x=x}RFL \quad x = x \to y = x}{y = x} \to e$$

$$\frac{y = x}{x = y \to y = x} \to i_1$$

Tutta questa riduzione può essere considerata anche come una regola derivata per semplificare prove più elaborate, definita come:

$$\frac{x = y}{y = x}\ SYM$$

Osservazione 23.2.1
L'ordine tra $x = y$ e $\frac{}{x=x}RFL$ conta poichè cambia l'approccio, infatti inverdendo non arriveremmo a qualcosa di non utile:

$$\frac{\frac{}{x=x}RFL \quad x = y}{x = x \to x = y}SUB_\phi$$

Quindi l'ordine conta poiché la sostituzione viene fatta con un ordine specifico. A seconda dell'utilizzo si può decidere se considerare quest'ordine o il precedente.

23.2.3 Transitività

Esercizio 23.2.5

$$x = y, y = z \vdash x = z$$

Considerando $t_1 = x$, $s_1 = y$ e $t_2 = y$, $s_2 = x$, avremo che:

$$\frac{x = y \quad \frac{}{x=x}RFL}{x = x \to y = x}SUB_\phi$$

$$\frac{\frac{}{x=x}RFL \quad x = x \to y = x}{y = x} \to e$$

Considerando $t_1 = y$, $s_1 = x$ e $t_2 = y$, $s_2 = z$, avremo che:

$$\frac{y = x \quad y = z}{y = y \to x = z}SUB_\phi$$

$$\frac{y = y \to x = z \quad \frac{}{y=y}RFL}{x = z} \to e$$

per effettuare la prova utilizziamo

$$\vdash (x = y \land y = z) \to x = z$$

Tutta questa riduzione può essere considerata anche come una regola derivata per semplificare prove più elaborate, definita come:

$$\frac{x = y \quad y = z}{x = z}\ TRANS$$

23.3 Altri Esempi Deduzioni con Uguaglianza

Esercizio 23.3.1

$$\vdash y = z \leftrightarrow \forall x.(x = y \rightarrow x = z)$$

Dimostrazione verso $\rightarrow$

$$\vdash y = z \rightarrow \forall x.(x = y \rightarrow x = z)$$

Riscrivibile anche come:

$$y = z \vdash \forall x.(x = y \rightarrow x = z)$$

Assunzioni: $k = y$

$$\frac{\frac{y=z}{z=y}SYM \quad \frac{[k=y]_1}{y=k}SYM}{z = k}TRANS$$

$$\frac{z = k}{k = z}SYM$$

$$\frac{k = z}{k = y \rightarrow k = z} \rightarrow i_1$$

$$\frac{k = y \rightarrow k = z}{\forall x.(x = y \rightarrow x = z)}\forall i$$

Prenderemo come assunzione $k = y$ perché ci viene "regalata" nel risalire dall'implicazione, in particolar modo a questo passo qui:

$$\frac{k = z}{k = y \rightarrow k = z} \rightarrow i_1$$

Dimostrazione verso $\leftarrow$

$$\vdash \forall x.(x = y \rightarrow x = z) \rightarrow y = z$$

Riscrivibile anche come:

$$\forall x.(x = y \rightarrow x = z) \vdash y = z$$

$$\frac{\forall x.(x = y \rightarrow x = z)}{y = y \rightarrow y = z}\forall e$$

Con la regola $\forall e$ non abbiamo fatto altro che sostituire x con y.

$$\frac{\frac{}{y=y}RFL \quad y = y \rightarrow y = z}{y = z} \rightarrow e$$

Esercizio 23.3.2

$$\exists x.P(x) \vdash \forall x.(x = a \vee x = b) \to (\neg P(a) \to P(b))$$

Assunzioni Iniziali: $\forall x.(x = a \vee x = b)$, $\neg P(a)$ e $\exists x.P(x)$
L'assunzione $\forall x.(x = a \vee x = b)$ la prendiamo perché risalendo dal basso la posso scartare con $\to_i$.
Stesso ragionamento vale per $\neg P(a)$ che risalendo dal basso la posso scartare con $\to_i$.

Dalle assunzioni possiamo ricavare anche queste altre assunzioni.
Nello specifico $\forall x.(x = a \vee x = b)$ ci regala: $y = a$ e $y = b$ che possiamo scartare con $\vee_e$.
Mentre $\exists x.P(x)$ ci regala: $P(y)$ che possiamo scartare con $\exists$

Assunzioni Finali: $\forall x.(x = a \vee x = b)$, $\neg P(a)$, $\exists x.P(x)$, $y = a$, $y = b$ e $P(y)$

$$\frac{[y = a]_1}{P(y) \to P(a)} SUB_\phi$$

$$\frac{P(y) \to P(a) \quad [P(y)]_4}{P(a)} \to e$$

$$\frac{P(a) \quad [\neg P(a)]_5}{\bot} \neg e$$

Mettiamo in sospesa questo ramo dell'albero.

$$\frac{[y = b]_2}{P(y) \to P(b)} SUB_\phi$$

$$\frac{P(y) \to P(b) \quad [P(y)]_4}{P(b)} \to e$$

$$\frac{P(b) \quad [\neg P(b)]_3}{\bot} \neg e$$

Mettiamo in sospeso anche questo ramo.

$$\frac{\forall x.(x = a \vee x = b)}{y = a \vee y = b} \forall e$$

Adesso utilizziamo tutte le conclusioni ottenute.

$$\frac{\bot \quad \bot \quad y = a \vee y = b}{\bot} \vee e_{1,2}$$

$$\frac{\bot}{P(b)} R.A._3$$

$$\frac{P(b) \quad \exists x.P(x)}{P(b)} \exists e_4$$

$$\frac{P(b)}{\neg P(a) \to P(b)} \to i_5$$

$$\frac{\neg P(a) \to P(b)}{\forall x.(x = a \vee x = b) \to (\neg P(a) \to P(b))} \to i$$

Esercizi per Casa

1. $\exists x.\neg P(x) \to \neg \forall x.P(x)$
2. $\neg \forall x.P(x) \to \exists x.\neg P(x)$
3. $\neg \exists x.P(x) \to \forall x.\neg P(x)$
4. $\forall x.\neg P(x) \to \neg \exists x.P(x)$
5. $\forall x.(\exists y.P(y) \to Q(x)) \vdash \forall x.\exists y.(P(y) \to Q(x))$
6. $\forall x.(P(x,x) \vee \forall y.Q(x,y)) \vdash \forall x.(\exists y.P(x,y) \vee Q(x,x))$
7. $\exists x.(P(x,x) \wedge \forall y.Q(x,y)) \vdash \exists x.(\exists y.P(x,y) \wedge Q(x,x))$
8. $\forall x.\forall y.(R(x,y) \leftrightarrow x = y) \vdash \forall x.R(x,x)$
9. $\forall x.\neg R(x,x), R(a,b) \vdash \exists x.\exists y.\neg(x = y)$
10. $\forall x.\forall y.((P(x) \wedge x = y) \to \neg Q(y)) \vdash \forall z.\neg(P(z) \wedge Q(z))$
11. $b = a, a = c \vdash b = c$ [senza sym e TRANS]

Capitolo 24

Lezione 24

24.1 FOL: Correttezza e Completezza

Correttezza FOL

$\Gamma \vdash_{ND} \phi \Rightarrow \Gamma \models \phi$

La dimostrazione è per induzione sulla lunghezza della deduzione

Caso Base ($l = 1$)

Se la deduzione ha lunghezza $l = 1$, allora $\phi \in \Gamma$ poiché nessuna regola è stata applicata, allora otteniamo direttamente la conclusione.

Caso Induttivo ($l > 1$)

Su deduzioni con $l < m$ supponiamo che vale l'IPOTESI D'INDUZIONE (H.I)

Γ
$\vdots$
ϕ

Allora visto che ϕ ha lun $< m$, supponiamo che per H.I vale che: $\Gamma \vdash \phi \Rightarrow \Gamma \models \phi$

Allora dobbiamo verificare che applicando l'ultima regola vale ancora

1) $\forall i$ (ultima regola) (dimostrazione analoga per $\exists e$)

Γ
$\vdots$
ϕ^x_y
$\overline{\forall x.\phi}\ \forall i$

Vogliamo dimostrare che: $\Gamma \models \forall x.\phi$ ($\forall x.\phi$ è conseg. logica di Γ)

Per H.I sappiamo che $\Gamma \vdash \phi^x_y \Rightarrow \Gamma \models \phi^x_y$

ma $\Gamma \models \phi^x_y \overset{\text{def. conseg. logica}}{\Longleftrightarrow} \forall M, \sigma \quad M, \sigma \models \Gamma \Rightarrow M, \sigma \models \phi^x_y$ (#)

Inoltre per i vincoli del $\forall i$ sappiamo che y non è libera Γ e preso un arbitrario M, σ vale che $M, \sigma \models \Gamma \Longleftrightarrow M, \sigma[y \leftarrow d] \models \Gamma$. (*) Quanto affermato è dovuto al fatto che la sostituzione di una variabile non libera (cioè y) non ha impatto sulla soddisfacibilità.

Di conseguenza dalle due affermazioni (sottolineate) otteniamo che:

(Δ) $\forall M, \sigma \quad M, \sigma[y \leftarrow d] \models \Gamma \Rightarrow M, \sigma[y \leftarrow d] \models \phi^x_y \quad \forall d \in M$ (cioè per ogni d elemento del dominio)

Se per $\forall M, \sigma$ vale (#)

e

per un arbitrario M', σ' vale (*), allora per M', σ' vale (#)

Possiamo osservare che ogni occorrenza di y in ϕ^x_y è un'occorrenza libera di x in ϕ (e viceversa). Allora possiamo affermare che:

$$M, \sigma[y \leftarrow d] \models \phi^x_y \iff M, \sigma[x \leftarrow d] \models \phi$$

In altre parole, mettere $y \leftarrow d$ dove ho sostituito y con le occorrenze libere di x (in ϕ^x_y) è la stessa cosa di mettere direttamente d nelle occorrenza libere di x (in ϕ).

Di conseguenza quanto ottenuto in verde può essere sostituito con quello in azzurro nell'affermazione (Δ) ottenendo: $\forall M, \sigma \quad M, \sigma[y \leftarrow d] \models \Gamma \Rightarrow M, \sigma[x \leftarrow d] \models \phi$

Analogo, da (#) otteniamo allora che:

$$\forall M, \sigma \quad \boxed{M, \sigma \models \Gamma} \Rightarrow \underbrace{\forall d \quad M, \sigma[x \leftarrow d] \models \phi}$$

$$\forall M, \sigma \quad M, \sigma \models \Gamma \Rightarrow M, \sigma \models \forall x.\phi \quad \text{allora} \quad \Gamma \models \forall x.\phi$$

cioè la def. di conseguenza logica

2) $\forall e$ (ultima regola) (Dimostrazione analoga per $\exists i$)

$$\frac{\begin{array}{c}\Gamma \\ \vdots \\ \forall x.\phi\end{array}}{\phi^x_t}\ \forall e$$

Vogliamo dimostrare che $\Gamma \models \phi^x_t$

con t libera per x in ϕ

Per H.I sappiamo che $\Gamma \vdash \forall x.\phi \Rightarrow \Gamma \models \forall x.\phi$

$\Updownarrow$ Def. cons. logica

$$\boxed{\forall M, \sigma \quad M, \sigma \models \Gamma} \Rightarrow M, \sigma \models \forall x.\phi$$

$\Downarrow$ Per la semantica $\forall i$ (Interpretazione di $\forall x.\phi$)

$$M, \sigma[x \leftarrow d] \models \phi \quad \forall d \in M$$

+

Preso un t generico, $[\![t]\!]^M_\sigma \in M$

Allora otteniamo (+):

$$M, \sigma[x \leftarrow [\![t]\!]^M_\sigma] \models \phi \iff \boxed{M, \sigma \models \phi^t_x}$$

T. SOSTITUZIONE (cont libera per x in ϕ)

Allora per TRANSITIVITÀ:

$\forall M, \sigma \quad M, \sigma \models \Gamma \Rightarrow M, \sigma \models \phi^t_x$ che è proprio la definizione della cons. logica $\Gamma \models \phi^t_x$

Completezza FOL

$\Gamma \models \phi \Rightarrow \Gamma \vdash_{N.D} \phi$

Partiamo da un insieme Γ di formule chiuse (proposizioni) e vogliamo dimostrare che

Γ è consistente $\Longrightarrow$ Γ soddisfacibile

Il ragionamento che faremo è sulle proposizioni, ma questo può essere generalizzato a insiemi di formule arbitrarie

L'idea è sempre quella di aggiungere delle formule fino ad arrivare a un insieme massimale consistente e dovremmo poi costruire un modello.

Il punto1 è che non è detto che un linguaggio abbia un numero sufficiente di nomi per poter parlare di tutti gli elementi del modello che servono a soddisfare la formula.

Di conseguenza allora preso $\mathcal{L}_0 = \mathcal{L}$ allora, costruiremo nuovi linguaggi aggiungendo delle costanti nel seguente modo:

$$\left.\begin{array}{l} \mathcal{L}_1 = \mathcal{L}_0 \cup \{c_5^1 \mid 5 \in \mathbb{N}\} \\ \vdots \qquad\qquad \vdots \qquad\qquad \vdots \\ \mathcal{L}_{i+1} = \mathcal{L}_i \cup \{c_5^i \mid 5 \in \mathbb{N}\} \end{array}\right\} \text{Arricchiamo con un insieme infinito di costanti}$$

Il punto 2 è che aggiungendo dobbiamo preservare la consistenza e quindi bisogna garantire l'esistenza di un "testimone" per tutte le formule esistenziali.

Henkin: $\Delta \subseteq FOL$ è un insieme di Henkin se per ogni formula di $\mathcal{L}$ della forma $\exists x.\phi$ esiste un termine ground (cioè senza variabili) di $\mathcal{L}$ tale che:

$\underbrace{\exists x.\phi \to \phi_t^x}$ sta in Δ

Assioma di Henkin

Allora partendo da $\Gamma_0 = \Gamma$

* ψ_{c_5} deve avere un'unica variabile libera

$\vdots$ 5 è indice della formula di $\mathcal{L}_i$ nell'enumerazione

$$\Gamma_{i+1} = \Gamma_i \cup \{\exists x.\psi_5 \Rightarrow \psi_{c_5}^x \mid \exists x.\psi_5 \text{ è formula di } \mathcal{L}_i\}$$

Aggiungo l'assioma di Henkin

In sostanza, prendo da $\mathcal{L}_i$ una formula $\exists x.\psi$ e gli aggiungo l'assioma di Henkin prendendo una costante c_5 dove 5 rappresenta l'indice di ψ in $\mathcal{L}_i$.

Esempio:

$\mathcal{L}_1 = (\phi, \phi, \{P, R\})$ cioè senza costanti e senza funzioni, ma con relazioni

Busa la formula $\underbrace{\exists x.\, \exists y.\, (P(x,y) \wedge R(y))}_{\text{è in forma } \exists x.\, \psi} \in \Gamma_0$

Allora nella costruzione di: Assioma Henkin

$$\Gamma_1 = \Gamma_0 \cup \left\{ \exists x.\, \exists y.\, (P(x,y) \wedge R(y)) \Rightarrow \underbrace{\exists y.\, (P(c_{\psi_1}, y) \wedge R(y))}_{\text{ha un'unica variabile libera ed è una formula di } \mathcal{L}_1} \right\}$$

Possiamo osservare che ha un'unica variabile libera che possiamo sostituire con una nuova costante $c_{\psi_2} \in \mathcal{L}_2$

Assioma Henkin

$$\Gamma_2 = \Gamma_1 \cup \left\{ \exists y.\, (P(c_{\psi_1}, y) \wedge R(y)) \Rightarrow (P(c_{\psi_1}, c_{\psi_2}) \wedge R(c_{\psi_2})) \right\}$$

Allora $\mathcal{L}' = \bigcup_{i \geq 0} \mathcal{L}_i'$ e $\Gamma' = \bigcup_{i \geq 0} \Gamma_i$

Se Γ_0 era consistente, anche Γ' lo sarà poiché per la costruzione abbiamo usato gli assiomi di Henkin.

Applichiamo ora il "meccanismo di saturazione" partendo da:

$\Gamma_0' = \Gamma'$ e costruiamo usando la regola $\Gamma_{i+1}' = \begin{cases} \Gamma_i' \cup \{\phi_{i+1}\} & \text{se è consistente} \\ \Gamma_i' & \text{altrimenti} \end{cases}$

Allora verrà preservata la consistenza ad ogni passo fino ad ottenere un

$$\Gamma^* = \bigcup_{i \geq 0} \Gamma_i'$$ che è un insieme consistente, massimale e di Henkin.

Arrivati a questo punto possiamo osservare che Γ^* ha tutte le proprietà per la costruzione di un modello:

$\mathcal{M}^* = (D^*, \mathcal{J}^*)$ con $D^* = \{ t \in \mathcal{L}' \mid t \text{ è ground} \}$

- $\mathcal{J}^*(c) = c^{\mathcal{M}^*} = c \quad \forall c$ costante di $\mathcal{L}'$
- $\mathcal{J}^*(f^n) = f^{\mathcal{M}^*} : (D^*)^n \to D^* \qquad f^{\mathcal{M}^*}(t_1, \dots, t_n) = f(t_1, \dots, t_n)$
- $\mathcal{J}^*(P^n) = \{(t_1, \dots, t_n) \in (D^*)^n \mid P^n(t_1, \dots, t_n) \in \Gamma^*\}$