Logica Computazionale

AlpT (@freaknet.org)
http://freaknet.org/alpt

February 8, 2009

Abstract

Questo testo e' una rielaborazione personale degli appunti presi durante il corso di Logica Computazionale, tenuto dalla Prof.ssa Marianna Nicolosi Asmundo presso il dipartimento di Matematica, Catania, A.A. 2008/2009.
Saro' ben lieto di correggere ogni eventuale errore che mi comunicherai.

Buon lettura.

^_^

Copyright ©2008 Andrea Lo Pumo aka AlpT <alpt@freaknet.org>. All rights reserved.

This document is free; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This document is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this document; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

Contents

1 Introduzione **1**

2 Prefazione **1**

3 Induzione **1**

4 Logica proposizionale **7**
- 4.1 Forme normali . . . 15
  - 4.1.1 CNF e DNF . . . 18
- 4.2 Tableaux . . . 22
- 4.3 Risoluzioni . . . 25
- 4.4 Completezza . . . 28
- 4.5 Compattezza . . . 41
- 4.6 Risoluzioni ristrette . . . 42
- 4.7 Conseguenza proposizionale . . . 47
- 4.8 Sistemi di Hilbert . . . 48
- 4.9 Sistema di deduzione naturale . . . 55
- 4.10 Procedura di Davis Putnam . . . 61

5 Logica del I ordine **66**
- 5.1 Semantica . . . 75
- 5.2 Termini chiusi . . . 77
- 5.3 Notazione uniforme . . . 82
- 5.4 Sistemi di deduzione . . . 84
  - 5.4.1 Tableaux I ordine . . . 89
  - 5.4.2 Risoluzioni I ordine . . . 90
  - 5.4.3 Hilbert del I ordine . . . 91
  - 5.4.4 Deduzione naturale I ordine . . . 94
- 5.5 Completezza semantica . . . 94
- 5.6 Skolemizzazione . . . 100

6 * **107**

1 Introduzione

Il nostro scopo sara' quello di costruire un programma che dimostri automaticamente i teoremi. Per far questo, avremo bisogno di definire un linguaggio, costruire su di esso una logica e trovare delle buone tecniche algoritmiche.

2 Prefazione

Definition 2.1. Un albero[1] e' infinito $\overset{\text{def}}{\Leftrightarrow}$ ha un numero infinito di nodi.
Un n-ramo e' una n-upla di nodi $(x_1, \ldots, x_n)$, dove x_1 la radice dell'albero, x_{i+1} e' un figlio di x_i.
Un n-ramo terminato e' un n-ramo $(x_1, \ldots, x_n)$, con x_n nodo senza figli.
Un ramo infinito e' una sequenza infinita $(x_1, x_2, \ldots,)$, dove $\forall n \in \mathbb{N}^*$ la sottosequenza $(x_1, \ldots, x_n)$ e' un n-ramo.
Un albero e' ha diramazione finita $\overset{\text{def}}{\Leftrightarrow}$ ogni nodo ha un numero finito di foglie (figli immediati).
Un nodo e' infinito $\overset{\text{def}}{\Leftrightarrow}$ e' radice di un albero infinito.
Se A e' un albero, definiamo i seguenti insiemi:

$$\begin{aligned}
&\text{ramo}\,(A) = \{\text{tutti i rami di } A\}\\
&\text{ramo}_t\,(A) = \{\text{tutti i rami terminati di } A\}\\
&\text{nodo}\,(A) = \{\text{tutti i nodi di } A\}
\end{aligned}$$

Theorem 2.2. *Lemma di Konig.*
Un albero infinito, a diramazione finita, ha almeno un ramo infinito.

Proof:
$\langle 1\rangle 1$. Dim. che un nodo X infinito ha almeno un figlio infinito
Se per assurdo non avesse un figlio infinito, potremmo contare i nodi dell'albero che ha per radice X:

$$\begin{aligned}
&\text{Sia } n(X) = |\,\{\text{nodi nell'albero che comincia da } X\}\,|\\
&n(X) = n(X_1) + \cdots + n(X_n) \underbrace{<}_{n(X_i)<\infty\ \forall i} \infty\\
&\text{dove } X_1, \ldots, X_n \text{ sono i figli immediati di } X
\end{aligned}$$

e questo e' assurdo.
Sia X la radice dell'albero infinito. X e' quindi un nodo infinito, allora per il passo $\langle 1\rangle 1$ avra' un figlio X_1 infinito, che a sua volta avra' un figlio X_2 infinito, e cosi' via per induzione. Abbiamo cosi' trovato il ramo infinito

$$X, X_1, X_2, X_3, \ldots$$

□

3 Induzione

Definition 3.1. $\mathcal{P}(U)$ e' l'insieme della parti di U.
Con A^∞ intendiamo l'insieme

$$A^\infty \overset{\text{def}}{=} \bigcup_{n\in N} A^n$$

[1] Per il concetto di albero vedi http://it.wikipedia.org/wiki/Albero_(informatica)

Definition 3.2. Data la relazione $r \subseteq \mathcal{U}^{n+1}$, poniamo

$$\hat{r} \stackrel{\text{def}}{=} n+1$$
$$r(u_1, \ldots, u_n) \stackrel{\text{def}}{=} \{u \in \mathcal{U} \mid (u_1, \ldots, u_n, u) \in r\}$$

Alle volte, per brevita' scriveremo $\overrightarrow{u} = (u_1, \ldots, u_n)$
Se $T \subseteq \mathcal{U}^n$, poniamo

$$r(T) \stackrel{\text{def}}{=} \bigcup_{\overrightarrow{u} \in T} r(\overrightarrow{u})$$

Ricordiamo infine che

$$r \text{ e' una } \textit{funzione } r : \mathcal{U}^n \longrightarrow \mathcal{U} \;\Leftrightarrow\; r \subseteq \mathcal{U}^{n+1}, \text{ e inoltre } |r(\overrightarrow{u})| = 1 \;\; \forall \overrightarrow{u} \in \mathcal{U}^n$$

Definition 3.3. Sia $\mathcal{U}$ il nostro universo e sia

$$B \subseteq \mathcal{U}$$
$$\mathcal{K} \subseteq \left\{r \mid r \subseteq \mathcal{U}^{n+1},\ n \in \mathbb{N}\right\}$$
$$\mathbb{S}(B, \mathcal{K}) = \left\{S \subseteq \mathcal{U} \mid B \subseteq S,\ \ r(S^{\hat{r}}) \subseteq S,\ \ r \in \mathcal{K}\right\}$$

In pratica, un insieme $S \in \mathbb{S}(B, \mathcal{K})$ contiene tutti gli elementi di B e tutti quegli elementi u che completando un vettore in $S^{\hat{r}}$ soddisfano r.
Sia $A, B \subseteq \mathcal{U}$,

$$A \text{ e' un insieme induttivo costruito da } B, \mathcal{K} \stackrel{\text{def}}{\Leftrightarrow} A = \bigcap_{S \in \mathbb{S}(B,\mathcal{K})} S$$

Proposition 3.4. *Si vede facilmente che*

$$A = \min \mathbb{S}(B, \mathcal{K})$$

A e' quindi semplicemente il piu' piccolo degli elementi di $\mathbb{S}$.

Proposition 3.5.

$$A = B \cup \bigcup_{r \in \mathcal{K}} r(A^{\hat{r}})$$

Proof:
$\langle 1 \rangle 1$. Dim. $\supseteq$

$$B \subseteq S \;\; \forall S \in \mathbb{S}(B, \mathcal{K}) \Rightarrow B \subseteq \bigcap_{S \in \mathbb{S}(B,\mathcal{K})} S = A$$
$$\overrightarrow{a} \in A^{\hat{r}} \Rightarrow \overrightarrow{a} \in S^{\hat{r}} \;\; \forall S \in \mathbb{S}(B, \mathcal{K}) \underbrace{\Rightarrow}_{\text{def. di } \mathbb{S}} r(\overrightarrow{a}) \subseteq S \;\; \forall S \in \mathbb{S} \Rightarrow r(\overrightarrow{a}) \subseteq A$$
$$r(\overrightarrow{a}) \subseteq A \;\; \forall \overrightarrow{a} \in A^{\hat{r}} \Rightarrow r(A^{\hat{r}}) \subseteq A$$

$\langle 1 \rangle 2$. Dim $\subseteq$

Sia $C = B \cup \bigcup_{r \in \mathcal{K}} r(A^{\hat{r}})$, si ha

$$B \subseteq C$$
$$C^{\hat{r}} \underbrace{\subseteq}_{\text{passo } \langle 1 \rangle 1} A^{\hat{r}} \Rightarrow r\left(C^{\hat{r}}\right) \subseteq r\left(A^{\hat{r}}\right) \underbrace{\subseteq}_{\text{def. di } C} C$$
$$\Rightarrow C \in \mathbb{S}(B, \mathcal{K}) \Rightarrow A = \min \mathbb{S}(B, \mathcal{K}) \subseteq C$$

□

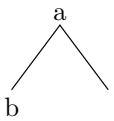
Example 3.6. Costruiamo induttivamente l'insieme A degli alberi binarii.
Definiamo il nostro universo:

$$\mathcal{U}_0 = \{\mathbb{N}\}$$
$$\mathcal{U}_i = \{\mathcal{P}(\mathcal{U}_j) \times \mathcal{P}(\mathcal{U}_h) \times \mathcal{P}(\mathcal{U}_k) \mid j, h, k < i\}$$
$$\mathcal{U} = \bigcup_{n \in N} \left(\bigcup_{U \in \mathcal{U}_n} U \right)$$

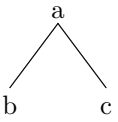
Definiamo B e $\mathcal{K}$:

$$B = \{\{a\} \mid a \in \mathbb{N}\}$$
$$\mathcal{K} = \{\text{tree}\}, \text{ dove}$$
$$\text{tree} : \mathcal{U}^3 \longrightarrow \mathcal{U}$$
$$\text{tree}(\{0\}, \{0\}, \{0\}) = \{0\}$$
$$\text{tree}(a, b, c) = (\{a\}, \{b\}, \{c\})$$

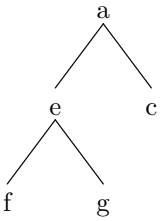
- $\{0\}$ verra' considerato come l'albero nullo.
- $\{a\} \in \mathcal{B} \backslash \{\{0\}\}$ l'albero con un solo nodo
- $(\{a\}, \{b\}, \{0\})$ il seguente albero



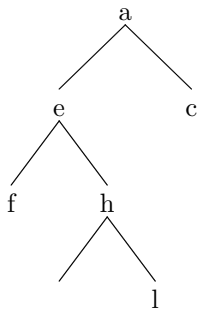
- $(\{a\}, \{b\}, \{c\})$ il seguente albero



- $(\{a\}, \{(\{e\}, \{f\}, \{g\})\}, \{c\})$ il seguente albero



- $(\{a\},\{(\{e\},\{f\},\{(\{h\},\{0\},\{l\})\})\},\{c\})$ il seguente albero



Definition 3.7. Sia $m \geq 1$. B e $\mathcal{K}$ hanno lo stesso significato della definizione [3.3,pg.2],

$u_1, \ldots, u_m \in \mathcal{U}$ e' un sequenza di costruzione $\overset{\text{def}}{\Leftrightarrow}$
$\Leftrightarrow \forall 1 \leq i \leq m \;\; u_i \in B \;\vee\; \exists r \in \mathcal{K} : \; u_i \in r(u_{i_1}, \ldots, u_{i_n}),$ con $i_j < i \;\; \forall j = 1, \ldots, n$

Sia $A \subseteq \mathcal{U}$.
A e' costruito tramite sequenze su $B, \mathcal{K}$ $\overset{\text{def}}{\Leftrightarrow}$ $\forall a \in A \;\; \exists u_1, \ldots, u_m \in \mathcal{U}$ sequenza di costruzione t.c. $a = u_m$

Example 3.8. Costruiamo tramite una sequenza i numeri di Fibonacci.

$$\begin{aligned}
&\mathcal{U} = \mathbb{N} \\
&B = \{0, 1\} \\
&\mathcal{K} = \{+\}, \text{ dove } + \text{ e' la normale addizione}
\end{aligned}$$

Costruiamo il numero di Fibonacci 8:

$$\begin{aligned}
u_1 &= 0 \in B \\
u_2 &= 1 \in B \\
u_3 &= +(u_1, u_2) = 1 \\
u_4 &= +(u_2, u_3) = 2 \\
u_5 &= +(u_4, u_3) = 3 \\
u_6 &= +(u_5, u_4) = 5 \\
u_7 &= +(u_6, u_5) = 8
\end{aligned}$$

Proposition 3.9. *Sia $\overline{A}$ un insieme costruito tramite sequenze su $B, \mathcal{K}$, allora*

$$\overline{A} = \min \mathbb{S}(B, \mathcal{K})$$

ovvero, $\overline{A}$ e' l'insieme indotto da $B, \mathcal{K}$.
In altre parole, costruire per sequenza e' equivalente a costruire tramite induzione.

Proof: Sia $A = \min \mathbb{S}(B, \mathcal{K})$.
$\langle 1 \rangle 1$. Dim. $\overline{A} \in \mathbb{S}(B, \mathcal{K})$

Infatti,

1. $B \subseteq \overline{A}$
2. $u \in r(a_1, \ldots, a_m) \underbrace{\Rightarrow}_{\text{per def.}} u$ e' costruito dalla sequenza $\underbrace{a_{11}, \ldots, a_{1n_1}}_{\text{sequenza di } a_1}, \underbrace{a_{21}, \ldots, a_{2n_2}}_{\text{sequenza di } a_2}, \ldots, \underbrace{a_{m1}, \ldots, a_{mn_m}}_{\text{sequenza di } a_m}, u$

Quindi,

$$A = \min \mathbb{S}(B, \mathcal{K}) \subseteq \overline{A} \in \mathbb{S}(B, \mathcal{K})$$

Supponiamo adesso per assurdo che $\exists u \in \overline{A} \backslash A$, si ha

$$u \notin A \underbrace{\Rightarrow}_{B \subseteq A} u \notin B \quad (1)$$

$$u \in \overline{A} \Rightarrow u \text{ e' costruito da una sequenza } u_1, \ldots, u_m \text{ con } u_m = u \quad (1.1)$$

$$(1), (1.1) \underbrace{\Rightarrow}_{\text{def. di } \overline{A}} \exists \overrightarrow{u} = (u_{i_1}, \ldots, u_{i_n}) \in \mathcal{U}^n \; \exists r \in \mathcal{K} : \quad u \in r(\overrightarrow{u}), \quad \text{con } i_j < m$$

Se per assurdo $\overrightarrow{u} \in A^n$, si avrebbe:

$$\overrightarrow{u} \in A^n \underbrace{\Rightarrow}_{A \in \mathbb{S}(B, \mathcal{K})} u \in r(\overrightarrow{u}) \subseteq A$$

Assurdo contro Hp. Quindi,

$$\overrightarrow{u} \notin A^n \Rightarrow \exists u_{i_j} \text{ componente di } \overrightarrow{u} : \quad u_{i_j} \notin A$$

Tuttavia, u_{i_j} fa parte della sequenza di costruzione $\overrightarrow{u}$, quindi a sua volta e' costruito dai suoi elementi della sequenza $u_1, \ldots, u_m$ a lui precedenti. Poiche' $u_{i_j} \notin A$, possiamo ripetere lo stesso ragionamento di prima sostituendo u con u_{i_j}. Risulterebbe cosi' che u e' costruito da una sequenza infinita, e questo e' assurdo. □

Definition 3.10. Sia A costruito induttivamente da $B, \mathcal{K}$,

$$\text{La costruzione di } A \text{ e' libera } \overset{\text{def}}{\Leftrightarrow} \forall a \in A \;\; a \in B \vee \exists! r \in \mathcal{K} \exists! \overrightarrow{u} \in A^{\hat{r}} : \; a \in r(\overrightarrow{u})$$

Proposition 3.11. *Induzione strutturale.*
Sia $A = \min \mathbb{S}(B, \mathcal{K})$ e sia $P : A \longrightarrow \{\mathbf{t}, \mathbf{f}\}$ una proposizione riguardante gli elementi di A. Se le seguenti condizioni sono soddisfatte:

1. $P(b) = \mathbf{t} \;\; \forall b \in B$
2. $\forall r \in \mathcal{K} \; \forall \overrightarrow{u} \in A^{\hat{r}} \;\; ((\forall i = 1, \ldots, \hat{r} \;\; P(u_i) = \mathbf{t}) \Rightarrow P(a) = \mathbf{t} \; \forall a \in r(\overrightarrow{u}))$

allora si ha

$$P(a) = \mathbf{t} \;\; \forall a \in A$$

Proof: Sia

$$C = \{a \in A \mid P(a) = \mathbf{t}\}$$

Per la 1. e 2. si ha

$$B \subseteq C$$
$$r(C^{\hat{r}}) \subseteq C$$

quindi, per definizione di $\mathbb{S}$ si ha

$$C \in \mathbb{S}(B, \mathcal{K}) \Rightarrow A = \min \mathbb{S}(\mathcal{B}, K) \subseteq C$$

ma d'altronde, per definizione di C, $C \subseteq A$. E quindi la tesi. □

Definition 3.12. Sia $a \in A$, con A insieme creato per sequenze su $B, \mathcal{K}$.

$$d(a) \stackrel{\text{def}}{=} \min\{m \in \mathbb{N} \mid u_1, \ldots, u_m \text{ sequenza di costruzione}, u_m = a\} - 1$$

$d(a)+1$ e' la lunghezza delle sequenze di costruzione piu' corte che generano a.

$$u_1, \ldots, u_{m+1} \text{ e' una sequenza minimale per } u_m \stackrel{\text{def}}{\Leftrightarrow} m = d(u_{m+1})$$

Proposition 3.13. *Sia $a_1, \ldots, (a_{m+1} = a)$ una sequenza di costruzione su $B, \mathcal{K}$. Si ha*

1. $d(a) = 0 \Leftrightarrow a \in B$
2. $d(a_i) \leq i - 1$
3. $d(a_{m+1}) = m \Rightarrow d(a) > d(a_i) \ \ \forall i \leq m$

Proof:
⟨1⟩1. Dim. 3.
Se per assurdo $m = d(a_{m+1}) \leq d(a_i)$, con $i \leq m$, si avrebbe

$$m = d(a_{m+1}) \leq d(a_i) \underbrace{\leq}_{2.} i - 1$$

$$i - 1 \leq i \leq m \leq i - 1 \Rightarrow i = i - 1 \Rightarrow -1 = 0 \text{ assurdo}$$

□

Proposition 3.14. *Sia $A = \min \mathbb{S}(B, \mathcal{K})$ e sia $P : A \longrightarrow \{\mathbf{t}, \mathbf{f}\}$ una proposizione riguardante gli elementi di A. Si ha*

$$P(a) = \mathbf{t} \ \ \forall a \in A \Leftrightarrow \forall n \in \mathbb{N}\,(P(a) = \mathbf{t} \ \ \forall a \in A : \ d(a) = n)$$

Questa proposizione e' utile per dimostrare "$P(a) = \mathbf{t} \ \ \forall a \in A$" tramite l'induzione sui naturali.

Proof:
⟨1⟩1. Dim. $\Rightarrow$
Chiaro.
⟨1⟩1. Dim. $\Leftarrow$

$$a \in A \underbrace{\Rightarrow}_{(*)} \exists m \in \mathbb{N} : \ d(a) = m \Rightarrow P(a) = \mathbf{t}$$

(*) Ogni elemento di A e' costruito (vedi [3.9,pg.4])

□

Definition 3.15. Funzione induttiva.
Sia A un insieme costruito induttivamente su $B, \mathcal{K}$. Sia V un secondo universo, non necessariamente uguale a $\mathcal{U}$. Sia data una funzione h tale che:

$$h : A \cup \mathcal{K} \longrightarrow V \cup \left\{f : V^{\hat{r}} \longrightarrow V \mid r \in \mathcal{K}\right\}$$
$$h(a) \in V \ \ \forall a \in A$$
$$h(r) \in \left\{f : V^{\hat{r}} \longrightarrow V \mid r \in \mathcal{K}\right\} \ \ \forall r \in \mathcal{K}$$

Una funzione $g : A \longrightarrow V$ si dira' induttiva su $A, h \Leftrightarrow \forall a \in A$ si ha

1. $\forall a \in B \ \ g(a) = h(a)$
2. $\forall a \in A \backslash B \ \ \exists r \in \mathcal{K} \exists \overrightarrow{u} \in A^{\hat{r}} : \ \begin{cases} a \in r(\overrightarrow{u}) \\ g(a) = (h(r))(g(u_1), \ldots, g(u_{\hat{r}})) \end{cases}$

Nota: Prima di tutto questa definizione non assicura che esistano g induttive su A, h, inoltre, poiche' nella condizione 2. richiediamo semplicemente che $\exists r, \exists \overrightarrow{u}$, fissata una h, potremmo avere diverse g induttive. Nella proposizione seguente, vedremo che quando A e' un insieme induttivo libero, per ogni h esistera' una sola g.

Proposition 3.16. *Nelle Hp della def [3.15,pg.6] si ha:*

$$A \text{ insieme induttivo libero } \Rightarrow \exists! \text{ funzione induttiva } g \text{ su } A, h$$

Proof: Costruiamo la funzione g ponendo:

$$\forall a \in B \;\; g(a) = h(a)$$
$$\forall a \in A \backslash B \;\; g(a) = (h(r))(g(u_1), \ldots, g(u_{\hat{r}}))$$
$$\text{dove } \overrightarrow{u} = (u_1, \ldots, u_{\hat{r}}) \in A^{\hat{r}} \text{ e } r \in \mathcal{K} \text{ sono gli unici t.c. } a \in r(\overrightarrow{u})$$

L'unicita' e l'esistenza di $\overrightarrow{u}, r$ sono garantiti dal fatto che A e' libero (vedi [3.10,pg.5]). Dimostriamo adesso che g e' ben definita. Procediamo per induzione strutturale, dimostrando che $\forall a \in A \;\; \exists! v \in V : \; g(a) = v$

- $$g(a) = h(a) \;\; \forall a \in B$$
  e poiche' h e' una funzione
  $$\forall a \in B \; \exists! v \in V : \; h(a) = v$$
- Sia $r \in \mathcal{K}, \overrightarrow{u} \in A^{\hat{r}}$. Sia $a \in r(\overrightarrow{u}) \backslash B$. Per ipotesi induttiva abbiamo:
  $$\forall u_i \text{ componente di } \overrightarrow{u} \; \exists! v_i \in V : \; g(u_i) = v_i$$
  $$a \in A \backslash B \Rightarrow g(a) = (h(r))(g(u_1), \ldots, g(u_{\hat{r}})) = (h(r))(v_1, \ldots, v_{\hat{r}})$$
  Poiche' $(h(r))$ e' una funzione, anche in questo caso il valore di $g(a)$ e' fissato ed unico.

Per costruzione, g e' una funzione induttiva su A, h. Dimostriamo adesso che e' unica: supponiamo che esista $f \neq g$ funzione induttiva su A, h

$$g(a) = h(a) = f(a) \;\; \forall a \in B \underbrace{\Rightarrow}_{f \neq g} \exists a \in A \backslash B : \; g(a) \neq f(a)$$

$$a \in A \backslash B \Rightarrow (h(r))(g(u_1), \ldots, g(u_{\hat{r}})) = g(a) \neq f(a) = (h(r))(f(u_1), \ldots, f(u_{\hat{r}})) \;\; (1)$$

Nota: stiamo usando lo stesso r, $\overrightarrow{u}$ sia per $g(a)$ che per $f(a)$ perche' sono gli unici che costruiscono a (A e' libero).

$$(1) \Rightarrow \exists i : \; g(u_i) \neq f(u_i) \Rightarrow u_i \in A \backslash B$$

A questo punto possiamo ripetere il ragionamento dalla (1) e avremo che la sequenza di costruzione di a e' infinita. Assurdo. □

4 Logica proposizionale

Definition 4.1. Il linguaggio della logica proposizionale e' formato dall'insieme $\text{Var} = \{p_1, p_2, \ldots, \}$ di lettere proposizionali, le lettere $TF = \{\mathbf{t}, \mathbf{f}\}$, e i simboli dei connettivi binari:

$$\text{Bin} = \{ \vee \, , \, \wedge \, , \, \triangledown \, , \, \triangle \, , \, \Rightarrow \, , \, \Leftarrow \, , \, \Leftrightarrow \, , \, \oplus \, , \not\Leftarrow, \not\Rightarrow \}$$

Nota[2] . Infine, abbiamo l'insieme dei simboli delle parentesi $\{(,)\}$.
L'insieme delle *formule atomiche* e' cosi' definito:

$$\text{AT} \stackrel{\text{def}}{=} \text{Var} \cup TF$$

[2] XOR = $\oplus$, NOR = $\triangledown$, NAND = $\triangle$

L'insieme Form delle formule e' definito induttivamente:

$\mathcal{U} = (\text{AT} \cup \text{Bin} \cup \{(,), \neg\})^\infty$ cioe', tutte le possibili stringhe che possiamo formare con i simboli sopracitati
$B = \text{AT}$
$\mathcal{K} = \{\text{neg}\} \cup \{\text{bin}_\circ \mid \forall \circ \in \text{Bin}\}$
Dove neg e' la funzione $\text{neg} : \mathcal{U} \longrightarrow \mathcal{U}$, $\text{neg}(P) = \neg P$
$\text{bin}_\circ$ e' la funzione $\text{bin}_\circ : \mathcal{U}^2 \longrightarrow \mathcal{U}$, $\text{bin}_\circ(P, Q) = P \circ Q$

In altri termini, Form e' il piu' piccolo sottoinsieme di $\mathcal{U}$ tale che

$$\begin{aligned} &\text{AT} \subseteq \text{Form} \\ &P \in \text{Form} \Rightarrow \neg P \in \text{Form} \\ &P, Q \in \text{Form} \Rightarrow P \circ Q \in \text{Form} \end{aligned}$$

Data una formula $Y \in \text{Form}$ definiamo l'insieme delle sottoformule immediate:

$$P \in \mathcal{P}_{\text{imm}}(Y) \overset{\text{def}}{\Leftrightarrow} \begin{cases} P = X \\ Y = \neg X \end{cases} \quad \text{oppure} \begin{cases} P = A \text{ oppure } P = B \\ Y = (A \circ B) \end{cases}$$

L'insieme di tutte le sottoformule di Y e' il piu' piccolo sottoinsieme $\mathcal{P}_{\text{form}}(Y) \subseteq \mathcal{U}$ t.c.

$$\begin{aligned} &Y \in \mathcal{P}_{\text{form}}(Y) \\ &Q \in \mathcal{P}_{\text{form}}(Y) \Rightarrow \mathcal{P}_{\text{imm}}(Q) \subseteq \mathcal{P}_{\text{form}}(Y) \end{aligned}$$

Definition 4.2. Adesso diamo una *interpretazione* ad ogni simbolo della logica proposizionale.
Ad ogni $\circ \in \text{Bin}$ associeremo la relativa funzione binaria, ad esempio, alla lettera $\vee$ associamo la funzione $AND : TF^2 \longrightarrow TF$.
Al simbolo $\neg$ associamo la funzione negazione $\neg : TF \longrightarrow TF$.
E cosi' via, in maniera del tutto naturale.
Per evitare confusioni, sottolineeremo il testo che indica una stringa di lettere, ad esempio

$$\underline{(A \vee B) \wedge B}$$

e' una stringa, in particolare e' una formula, quindi possiamo scrivere:

$$\underline{(A \vee B) \wedge B} \in \text{Form}$$

Definition 4.3. Valutazione booleana.
Una funzione $v : \text{Form} \longrightarrow TF$ si chiama valutazione $\Leftrightarrow$

1. $v(\underline{\mathbf{t}}) = \mathbf{t}$
2. $v(\underline{\mathbf{f}}) = \mathbf{f}$
3. $v(\underline{\neg X}) = \neg v(\underline{X})$
4. $v(\underline{X \circ Y}) = v(\underline{X}) \circ (\underline{Y}) \quad \forall \circ \in \text{Bin}$

Sia $\mathcal{V}$ l'insieme di tutte le valutazioni.

Lemma 4.4. *Data una funzione* $f : \text{Var} \longrightarrow TF$ *si ha*

$$\exists! v_f \text{ valutazione booleana: } \forall p \in \text{Var } \ v_f(p) = f(p)$$

Quindi le valutazioni booleane esistono, cioe' $\mathcal{V} \neq \emptyset$. Sono anche infinite perche' le Var *e' infinito e quindi abbiamo infinite funzioni del tipo* $\text{Var} \longrightarrow TF$

Proof: Definiamo induttivamente v_f in modo da rispettare le condizioni della def [4.3,pg.8] e la condizione $\forall p \in \text{Var } \ v_f(p) = f(p)$.
Poiche' Form e' un insieme induttivo libero, per la [3.16,pg.7] abbiamo la tesi. □

Lemma 4.5. *Sia* $X \in \text{Form}$ *e sia* $S \subseteq \text{Var}$ *l'insieme di tutte le lettere proposizionali di* X*, ovvero*

$$S = \mathcal{P}_{\text{form}}(X) \cap \text{Var}$$

allora

$$v_1, v_2 \text{ valutazioni: } v_1(P) = v_2(P) \ \ \forall P \in S \Rightarrow v_1(X) = v_2(X)$$

Quindi se due valutazioni sono identiche sulle lettere di una formula, allora saranno indentiche sulla stessa formula.

Proof: Si dimostra per induzione strutturale su Form:

$v_1(\mathbf{t}) = \mathbf{t} = v_2(\mathbf{t})$

$v_1(\mathbf{f}) = \mathbf{f} = v_2(\mathbf{f})$

$X \in \text{Var} \underbrace{\Rightarrow}_{\text{per Hp}} v_1(X) = v_2(X)$

$v_1(X) \underbrace{=}_{X=\neg Y} v_1(\neg Y) = v_1(\neg Y) = \neg v_1(Y) \underbrace{=}_{\text{Hp induttiva}} \neg v_2(Y) = v_2(\neg Y) = v_2(X)$

$v_1(X) = v_1(Z \circ Y) = v_1(Z) \circ v_1(Y) \underbrace{=}_{\text{Hp induttiva}} v_2(Z) \circ v_2(Y) = v_2(Z \circ Y) = v_2(X)$

□

Definition 4.6. Per ogni $X \in \text{Form}$, posto $S(X) = \mathcal{P}_{\text{form}}(X) \cap \text{Var}$, stabiliamo la seguente relazione d'equivalenza:

$$v_1 \overset{X}{\sim} v_2 \Leftrightarrow v_{1|S} = v_{2|S}$$

Per il lemma [4.5,pg.9], tutte le v di una stessa classe d'equivalenza avranno lo stesso valore su X, ovvero:

$$v_1(X) = v(x) \ \ \forall v \overset{X}{\sim} v_1$$

In questo modo, per conoscere tutte le possibili valutazioni su X, possiamo limitarci a considerare i rappresentati delle classi d'equivalenza $\overset{X}{\sim}$.
Se $|S(X)| = n$, ovvero se X ha n variabili proposizionali $P_1, \ldots, P_n$, il numero di rappresentanti sara' 2^n. Infatti, per ogni $f : \text{Var} \longrightarrow TR$ abbiamo la valutazione v_f. Tutte le f distinte su $P_1, \ldots, P_n$ sono tante quante le funzioni del tipo $\{P_1, \ldots, P_n\} \longrightarrow TR$, ovvero 2^n.
In pratica, calcolare tutte le valutazioni su X equivale a calcolare la tabella di verita' della formula X. Ad esempio, calcolare la tavola di verita' di $X = \underline{P_1 \Rightarrow (P_2 \Rightarrow P_1)}$.

Definition 4.7.
Sia $v \in \mathcal{V}$, poniamo
$v \models X \overset{\text{def}}{\Leftrightarrow} v(X) = \mathbf{t}$
$X \in$ Form e' una tautologia $\overset{\text{def}}{\Leftrightarrow} \forall v \in \mathcal{V}\ \ v \models X$
X e' insoddisfacibile $\overset{\text{def}}{\Leftrightarrow} \forall v \in \mathcal{V}\ \ v \not\models X$
X e' soddisfacibile $\Leftrightarrow \exists v \in \mathcal{V} :\ \ v \models X$

Proposition 4.8. *X tautologia $\Leftrightarrow \neg X$ insoddisf.*

Proof:

$$X \text{ tautologia } \Leftrightarrow v \models X\ \ \forall v \Leftrightarrow v(X) = \mathbf{t}\ \ \forall v \Leftrightarrow \neg v(X) = \mathbf{f}\ \ \forall v \Leftrightarrow v(\neg X) = \mathbf{f}\ \ \forall v$$

□

Definition 4.9. Sia $S \subseteq$Form, $X \in$ Form, poniamo

$$v \models S \overset{\text{def}}{\Leftrightarrow} v \models s\ \ \forall s \in S$$

$$S \text{ e' insoddisfacibile } \overset{\text{def}}{\Leftrightarrow} v \not\models S\ \ \forall v \in \mathcal{V}$$

$$S \text{ e' soddisfacibile } \Leftrightarrow \exists v \in \mathcal{V} :\ \ v \models S$$

$$S \models X \overset{\text{def}}{\Leftrightarrow} \forall v \in \mathcal{V}\ \ (v| = S \Rightarrow v| = X)$$

Quando $S \models X$, diremo che X e' una conseguenza logica di S o che X e' dedotto da S. (In pratica possiamo pensare S come un insieme di assiomi e X come una conseguenza degli assiomi).

Proposition 4.10. *Con $\models X$ intendiamo $\emptyset \models X$*

$$\emptyset \models X \Leftrightarrow v \models X\ \ \forall v \in \mathcal{V} \Leftrightarrow X \text{ e' una tautologia}$$

Proof:

$\emptyset \models X \Leftrightarrow \forall v \in \mathcal{V}\ (v \models \emptyset \Rightarrow v \models X)$ (1)
$v \models \emptyset \Leftrightarrow v \models s\ \ \forall s \in \emptyset \Leftrightarrow\ (s \in \emptyset \Rightarrow v \models s) \Leftrightarrow\ (\mathbf{f} \Rightarrow v \models s) \Leftrightarrow$ vero
Quindi: $v \models \emptyset \Leftrightarrow \mathbf{t}$ (2)
$(2), (1) \Rightarrow \forall v \in \mathcal{V}\ (\mathbf{t} \Rightarrow v \models X) \Rightarrow \forall v \in \mathcal{V}\ \ v \models X \Leftrightarrow v$ tautologia

□

Lemma 4.11. *Sia $S \subseteq$Form, $x \in$ Form,*

$$S \models X \Leftrightarrow S \cup \{\neg X\} \text{ e' insoddisfacibile}$$

Proof:
$\langle 1 \rangle 1$. Dim. $\Rightarrow$
Supponiamo per assurdo che $S \cup \{\neg X\}$ sia soddisfabile, ovvero che

$$\exists v \in \mathcal{V} :\ \ v \models S \cup \{\neg X\} \Rightarrow v \models \neg X \Rightarrow v(\neg X) = \mathbf{t} \Leftrightarrow v(X) = \mathbf{f}$$

$$\begin{cases} v \models S \\ S \models X \end{cases} \Rightarrow v \models X \Rightarrow v(X) = \mathbf{t}$$

assurdo.
$\langle 1 \rangle 2$. Dim. $\Leftarrow$
Sia $v \in \mathcal{V} :\ v \models S$, vogliamo mostrare che $v \models X$.

$$v \models S,\ S \cup \{\neg X\} \text{ insoddisfacibile } \Rightarrow v \not\models \neg X \Rightarrow v(\neg X) = \mathbf{f} \Leftrightarrow v(X) = \mathbf{t} \Leftrightarrow v \models X$$

□

Theorem 4.12. *della deduzione semantica.*
Sia $S \subseteq \text{Form}$, $X, Y \in \text{Form}$,

$$S \cup \{X\} \models Y \Leftrightarrow S \models \underline{(X \Rightarrow Y)}$$

Nota: usando la prop [4.10,pg.10] e ponendo $S = \emptyset$*, abbiamo il teorema di deduzione:*

$$X \models Y \Leftrightarrow \underline{X \Rightarrow Y} \text{ e' una tautologia}$$

Questo teorema giustifica l'usuale tecnica di dimostrazione delle implicazioni $A \Rightarrow B$*: si suppone* A *vera e si* deduce B.

Proof:
$\langle 1 \rangle 1$. Dim. $\Rightarrow$
Sia $v \in \mathcal{V} :\ v \models S$, dimostriamo che $v \models \underline{X \Rightarrow Y}$.

$$v(X \Rightarrow Y) = (v(X) \Rightarrow v(Y)) \quad (1)$$

Supponiamo $v(X) = \mathbf{t}$, cioe' $v \models X$, e sfruttiamo l'ipotesi:

$$\begin{cases} v \models S \\ v \models X \\ S \cup \{X\} \models Y \end{cases} \Rightarrow v \models Y \Rightarrow v(Y) = \mathbf{t}$$

$$v(x) = v(Y) = \mathbf{t} \Rightarrow (v(X) \Rightarrow v(Y)) = \mathbf{t}$$

Nel caso in cui $v(X) = \mathbf{f}$, si ha sempre

$$(v(X) \Rightarrow v(Y)) = \mathbf{t}$$

Quindi l'implicazione (1) e' vera, cioe'

$$v(X \Rightarrow Y) = \mathbf{t}$$

$\langle 1 \rangle 2$. Dim. $\Leftarrow$
Sia $v \in \mathcal{V} :\ v \models S \cup \{X\}$ dimostriamo che $v \models Y$.

$$v \models S \cup \{X\} \Rightarrow v \models X \Rightarrow v(X) = \mathbf{t}$$

$$v \models S \underbrace{\Rightarrow}_{\text{Hp}} v \models \underline{X \Rightarrow Y} \Rightarrow (v(X) \Rightarrow v(Y)) = \mathbf{t} \underbrace{\Rightarrow}_{v(X)=\mathbf{t}} v(Y) = \mathbf{t}$$

□

Corollary 4.13. *Sia* $\{X_1, \ldots, X_n\} \subseteq \text{Form}$,

$$\{X_1, \ldots, X_n\} \models Y \Leftrightarrow \models \underline{X_1 \Rightarrow (X_2 \Rightarrow (\ldots \Rightarrow (X_n \Rightarrow Y) \ldots))}$$

Con questo corollario possiamo ricondurre il problema di verificare la deduzione di Y *dagli assiomi* S *a quello di verificare una catena di implicazioni.*

Proof: Procediamo per induzione su n.
Per $n = 1$ basta rileggere la nota del thm [4.12,pg.11],

$$\{X_1, \ldots, X_n\} \models Y \Leftrightarrow \{X_1, \ldots, X_{n-1}\} \cup \{X_n\} \models Y \underbrace{\Leftrightarrow}_{[4.12,\text{pg.}11]} \{X_1, \ldots, X_{n-1}\} \models \underline{(X_n \Rightarrow Y)} \Leftrightarrow$$

$$\underbrace{\Leftrightarrow}_{\text{Hp induttiva}} X_1 \Rightarrow (X_2 \Rightarrow (\ldots \Rightarrow (X_{n-1} \Rightarrow (X_n \Rightarrow Y) \ldots))$$

□

Definition 4.14. Siano $\circ, \bullet : TR^2 \longrightarrow TR$ due funzioni binarie,

$$\circ \text{ dual } \bullet \overset{\text{def}}{\Leftrightarrow} \neg(a \circ b) = (\neg a \bullet \neg b)$$

Proposition 4.15.

$$\circ \text{ dual } \bullet \Leftrightarrow \bullet \text{ dual } \circ$$

Proof:

$$\neg(a \circ b) = (\neg a \bullet \neg b) = \neg(\neg(\neg a \bullet \neg b)) \underbrace{\Leftrightarrow}_{\text{negando ambo i membri}} a \circ b = \neg(\neg a \bullet \neg b) \Leftrightarrow \neg a \circ \neg b = \neg(a \bullet b)$$

□

Proposition 4.16. *Tutti i connettivi binari sono a due a due duali:*

$$\begin{array}{l} \vee \ , \ \ \wedge \\ \mathbf{t}, \mathbf{f} \\ \pi_1, \pi_1 \\ \pi_2, \pi_2 \\ \neg_{left}, \neg_{left} \\ \neg_{right}, \neg_{right} \\ \Leftarrow \ , \ \not\Rightarrow \\ \Rightarrow \ , \ \not\Leftarrow \\ \Leftrightarrow \ , \ \not\Leftrightarrow = \ \oplus \\ \nabla \ , \ \ \triangle \end{array}$$

Nota: $a \ \pi_1 \ b \stackrel{def}{=} a$
Nota: $a \ \neg_{left} \ b \stackrel{def}{=} \neg a$.

Proof: Per definizione abbiamo:

$$\circ \text{ dual } \bullet \Leftrightarrow \neg(a \circ b) = \neg a \bullet \neg b \Leftrightarrow a \circ b = \neg(\neg a \bullet \neg b) \quad (1)$$

Quindi conoscendo $\bullet$ possiamo ricavare $\circ$.

$$\neg(a \Leftarrow b) = \neg(\neg a \Rightarrow \neg b) = (\neg a \not\Rightarrow \neg b)$$
$$\neg(a \Rightarrow b) = \neg(\neg a \Leftarrow \neg b) = (\neg a \not\Leftarrow \neg b)$$
$$\neg(a \Leftrightarrow b) = \neg(\neg a \Leftrightarrow \neg b) = \neg a \not\Leftrightarrow \neg b$$
$$\neg(\neg a \triangle \neg b) = \neg(\neg(\neg a \wedge \neg b)) = \neg a \wedge \neg b = \neg(a \vee b) = a \nabla b \underbrace{\Rightarrow}_{\text{negando}} \neg(a \nabla b) = \neg a \triangle \neg b$$

□

Definition 4.17. Definiamo induttivamente la seguente funzione, chiamata *operatore duale*:

$$\cdot^{\delta} : \text{Form} \longrightarrow \text{Form}$$
$$A^{\delta} = A \ \ \forall A \in \text{Var}$$
$$\mathbf{t}^{\delta} = \mathbf{f}$$
$$\mathbf{f}^{\delta} = \mathbf{t}$$
$$(\neg X)^{\delta} = \neg X^{\delta} \ \ \forall X \in \text{Form}$$
$$(X \circ Y)^{\delta} = X^{\delta} \bullet Y^{\delta} \ \ \forall X, Y \in \text{Form} \ \forall \circ \in \text{Bin},$$
dove $\bullet$ e' il connettivo binario duale di $\circ$ (vedi [4.16,pg.12])

In pratica, $\cdot^\delta$ scambia tra loro $\mathbf{t}, \mathbf{f}$ e tutti i $\circ$ con i loro $\bullet$.
Definiamo l'operatore di negazione:

$$\begin{aligned}
&\overline{\cdot} : \text{Form} \longrightarrow \text{Form}\\
&\overline{A} = \neg A \ \ \forall A \in \text{Var}\\
&\overline{\mathbf{t}} = \mathbf{t}\\
&\overline{\mathbf{f}} = \mathbf{f}\\
&\overline{(\neg X)} = \neg\overline{X} \ \ \forall X \in \text{Form}\\
&\overline{(X \circ Y)} = \overline{X} \circ \overline{Y} \ \ \forall X, Y \in \text{Form}\ \forall \circ \in \text{Bin}
\end{aligned}$$

In pratica $\overline{\cdot}$ scambia ogni lettera proposizionale P con $\neg P$.

Proposition 4.18. *Sia* $X \in \text{Form}$

$$\models X \Leftrightarrow \models \overline{X}$$

Proof: Basta osservare che

$$\overline{v}(Y) = v(\overline{Y})$$

e' la valutazione v con i valori negati sulle lettere proposizionali, piu' precisamente, se $v = v_f$ allora $\overline{v} = v_{\neg f}$.
Inoltre e' chiaro che $\overline{\overline{v}} = v$:

$$\overline{v} = v_{\neg f} \Rightarrow \overline{\overline{v}} = v_{\neg\neg f} = v_f = v$$

Quindi,

$$\models X \Leftrightarrow \forall v \in \mathcal{V}\ v \models X \underbrace{\Rightarrow}_{\overline{v} \in \mathcal{V}} \overline{v} \models X \Leftrightarrow \overline{v}(X) = \mathbf{t} \Leftrightarrow v(\overline{X}) = \mathbf{t} \Leftrightarrow v \models \overline{X}$$

$$\forall v \in \mathcal{V}\ v \models \overline{X}$$

□

Proposition 4.19. *Sia* $v \in \mathcal{V}$, $X \in \text{Form}$, *allora*

$$v(\neg\overline{X}) = v(X^\delta)$$

Proof: Per induzione strutturata su Form:
$v(\neg\overline{P}) = v(\neg\neg P) = v(P) = v(P^\delta) \ \ \forall P \in \text{Var}$
$v(\neg\overline{\mathbf{t}}) = v(\neg\mathbf{t}) = v(\mathbf{f}) = v(\mathbf{t}^\delta)$
$v(\neg\overline{\mathbf{f}}) = v(\neg\mathbf{f}) = v(\mathbf{t}) = v(\mathbf{f}^\delta)$
$v(\neg\overline{(\neg X)}) = v(\neg\neg\overline{X}) = \neg v(\neg\overline{X}) \underbrace{=}_{\text{Hp indutt.}} \neg v(X^\delta) = v(\neg X^\delta) = v((\neg X)^\delta)$
$v(\neg\overline{(A \circ B)}) = v(\neg(\overline{A} \circ \overline{B})) \underbrace{=}_{\text{duale}} v(\neg\overline{A} \bullet \neg\overline{B})) = v(\neg\overline{A}) \bullet v(\neg\overline{B}) \underbrace{=}_{\text{Hp indutt}} v(A^\delta) \bullet v(B^\delta) = v(A^\delta \bullet B^\delta) = v((A \circ B)^\delta)$

□

Proposition 4.20. *Sia* $X, Y \in \text{Form}$

$$\models \underline{(X \Rightarrow Y)} \Rightarrow \models \underline{(Y^\delta \Rightarrow X^\delta)}$$

Proof:

$$\models \underline{(X \Rightarrow Y)} \underbrace{\Rightarrow}_{[4.18,\text{pg.}13]} \models \overline{X \Rightarrow Y} \Leftrightarrow v(\overline{X \Rightarrow Y}) = \mathbf{t} \ \ \forall v \in \mathcal{V} \ \ (1)$$

$$v(Y^\delta \Rightarrow X^\delta) = (v(Y^\delta) \Rightarrow v(X^\delta)) \underbrace{=}_{[4.19,\text{pg.}13]} (v(\neg\overline{Y}) \Rightarrow v(\neg\overline{X})) =$$

$$= (\neg v(\overline{Y}) \Rightarrow \neg v(\overline{X})) = (v(\overline{Y}) \Leftarrow v(\overline{X})) = v(\overline{X} \Rightarrow \overline{Y}) = v(\overline{(X \Rightarrow Y)}) \underbrace{=}_{(1)} \mathbf{t}$$

□

Definition 4.21. Sia $X, Y \in \text{Form}, \ P \in \text{Var}$. Definiamo

$$X(P|Y)$$

come quella formula ottenuta da X sostituendo la lettera P con la formula Y. Se la lettera P non e' presente in X, allora $X(P|Y) = X$.
Formalmente definiamo l'operatore di sostituzione:

$$(P|Y) : \text{Form} \longrightarrow \text{Form}$$

$$A(P|Y) = \begin{cases} Y & A = P \\ A & A \neq P \end{cases} \ \forall A \in \text{Var}$$

$$\mathbf{t}(P|Y) = \mathbf{t}$$

$$\mathbf{f}(P|Y) = \mathbf{f}$$

$$(\neg X)(P|Y) = \neg X(P|Y) \ \ \forall X \in \text{Form}$$

$$(X \circ Y)(P|Y) = X(P|Y) \circ Y(P|Y) \ \ \forall X, Y \in \text{Form} \ \forall \circ \in \text{Bin}$$

Proposition 4.22. *Sia* $v \in \mathcal{V}, \ Y, Z \in \text{Form}, \ P \in \text{Var}$. *Allora si ha*

$$v(Y) = v(Z) \Rightarrow v(X(P|Y)) = v(X(P|Z)) \ \ \forall X \in \text{Form}$$

Proof: Supponiamo che $v(Y) = v(Z)$ e procediamo per induzione su X:
$X = \mathbf{t} \Rightarrow X(P|Y) = \mathbf{t} = X(P|Z) \Rightarrow v(X(P|Y)) = v(X(P|Z))$

$X = \mathbf{f} \Rightarrow \ \ldots$ analogamente

$$X = P \Rightarrow X(P|Y) = Y, \ X(P|Z) = Z \underbrace{\Rightarrow}_{v(Y)=v(Z)} v(X(P|Y)) = v(X(P|Z))$$

$$X \in \text{Var} \setminus \{P\} \Rightarrow X(P|Y) = X = X(P|Z) \Rightarrow v(X(P|Y)) = v(X(P|Z))$$

$$X \in \text{Form}, \ v(\underline{\neg X}(P|Y)) \underbrace{=}_{\text{def}} v(\neg X(P|Y)) = \neg v(X(P|Y)) \underbrace{=}_{\text{Hp indutt.}} \neg v(X(P|Z)) \underbrace{=}_{\text{come prima}} v(\underline{\neg X}(P|Z))$$

$$A, B \in \text{Form}, \ v(\underline{A \circ B}(P|Y)) = v(A(P|Y)) \circ v(B(P|Y)) \underbrace{=}_{\text{Hp indutt.}} v(A(P|Z)) \circ v(B(P|Z)) = v(\underline{A \circ B}(P|Z))$$

□

Proposition 4.23.

$$\models \underline{Y \Leftrightarrow Z} \ \Rightarrow \ \models \underline{X(P|Y) \Leftrightarrow X(P|Z)}$$

Proof: Sia $v \in \mathcal{V}$
$\models \underline{Y \Leftrightarrow Z} \Rightarrow v(\underline{Y \Leftrightarrow Z}) = \mathbf{t} \ \Leftrightarrow \ (v(Y) \Leftrightarrow v(Z)) = \mathbf{t} \ \Leftrightarrow \ v(Y) = v(Z) = \mathbf{t} \ \ (1)$

$$(1) \underbrace{\Rightarrow}_{[4.22,\text{pg.}14]} v(X(P|Y)) = v(X(P|Z)) \Leftrightarrow (v(X(P|Y)) \Leftrightarrow v(X(P|Z))) = \mathbf{t} \Leftrightarrow v(\underline{X(P|Y) \Leftrightarrow X(P|Z)}) = \mathbf{t}$$

□

4.1 Forme normali

Definition 4.24. Diremo che una formula $X \in \text{Form}$ e' in forma normale negativa (FNN) se vale la seguente:

$$\underline{\neg Y} \in \mathcal{P}_{\text{form}}(X) \Rightarrow Y \in \text{Var}$$

ovvero, quando $\neg$ si trova solo davanti a lettere proposizionali.
Ogni formula si puo' convertire in FNN, utilizzando la seguente funzione induttiva:

$$\begin{aligned}
&\text{FNN}(P) = P \ \ \forall P \in \text{AT} \\
&\text{FNN}(\neg Y) = \begin{cases} \text{FNN}(X) & \text{se } Y = \neg X \\ \neg\,\text{FNN}(A) \bullet \neg\,\text{FNN}(B) & \text{se } Y = \neg(A \circ B) \end{cases} \\
&\text{FNN}(A \circ B) = \text{FNN}(A) \circ \text{FNN}(B)
\end{aligned}$$

Definition 4.25. I connettivi binari primari sono 10 e sono:

$$\text{Bin}_{\text{I}} \stackrel{\text{def}}{=} \{\ \wedge\ ,\ \vee\ ,\ \Rightarrow\ ,\ \Leftarrow\ ,\ \vartriangle\ ,\ \ \triangledown\ ,\ \not\Leftarrow,\ \not\Rightarrow\}$$

I connettivi secondari sono

$$\text{Bin}_{\text{II}} \stackrel{\text{def}}{=} \Leftrightarrow\ ,\ \not\Leftrightarrow$$

Si puo' verificare facilmente che utilizzando un $\circ \in \text{Bin}_{\text{I}}$ e $\neg$ (ad esempio $\neg$, $\vee$) e' possibile ricavare tutti gli altri connettivi.

Possiamo suddividere le formule del tipo $A \circ B$, $\neg A \circ B$, con $\circ \in \text{Bin}_{\text{I}}$ in due categorie: le α-formula e le β-formula.

$$\begin{aligned}
\alpha \in \alpha\text{-formula} &\stackrel{\text{def}}{\Leftrightarrow} \begin{cases} \alpha \in \left\{ \underline{X \circ Y},\ \ \underline{\neg(X \circ Y)} \mid X, Y \in \text{Form},\ \ \circ \in \text{Bin}_{\text{I}} \right\} \\ \models \underline{\alpha \Leftrightarrow \alpha_1 \wedge \alpha_2},\ \text{con } \alpha_1 \in \{X,\ \neg X\}\,,\ \ \alpha_2 \in \{Y,\ \ \neg Y\} \end{cases} \\
\beta \in \beta\text{-formula} &\stackrel{\text{def}}{\Leftrightarrow} \begin{cases} \beta \in \left\{ \underline{X \circ Y},\ \ \underline{\neg(X \circ Y)} \mid X, Y \in \text{Form},\ \ \circ \in \text{Bin}_{\text{I}} \right\} \\ \models \underline{\beta \Leftrightarrow \beta_1 \vee \beta_2},\ \text{con } \beta_1 \in \{X,\ \neg X\}\,,\ \ \beta_2 \in \{Y,\ \ \neg Y\} \end{cases}
\end{aligned}$$

α_1, α_2 si chiamano *componenti* di α
In altre parole, una α-formula e' una formula che si puo' scrivere equivalentemente come $\alpha_1 \wedge \alpha_2$. Invece, una β-formula si puo' scrivere equivalentemente come $\beta_1 \vee \beta_2$.

Definition 4.26. Sia FormI l'insieme induttivo delle formule definito analogamente a quanto fatto in [4.1,pg.7], ma usando Bin_{I} al posto di Bin.
Poiche' stiamo usando i connettivi primari, non avremo alcuna perdita di espressivita'. Inoltre, intenderemo i connettivi Bin_{II} definiti tramite Bin_{I}, ad esempio, la nostra scrittura

$$\underline{A \Leftrightarrow B}$$

sara' solo una abbreviazione per la formula

$$\underline{(A \Leftarrow B) \wedge (A \Rightarrow B)}$$

Definition 4.27. Ecco la lista delle α e β formule minimali (le piu' corte formule scritte in base alle loro componenti):

- α-formula

$$\begin{array}{c|cc} X \wedge Y & X & Y \\ \neg(X \vee Y) & \neg X & \neg Y \\ \neg(X \Rightarrow Y) & X & \neg Y \\ \neg(X \Leftarrow Y) & \neg X & Y \\ \neg(X \vartriangle Y) & X & Y \\ X \triangledown Y & \neg X & \neg Y \\ X \not\Rightarrow Y & X & \neg Y \\ X \not\Leftarrow Y & \neg X & Y \end{array}$$

  Definiamo Form_α come l'insieme delle α-formule presenti in questa tabella (prima colonna), con $X, Y \in \mathrm{FormI}$.

- β-formula

Poniamo

$$\begin{array}{c|cc} \neg(X \wedge Y) & \neg X & \neg Y \\ X \vee Y & X & Y \\ X \Rightarrow Y & \neg X & Y \\ X \Leftarrow Y & X & \neg Y \\ X \vartriangle Y) & \neg X & \neg Y \\ \neg(X \triangledown Y) & X & Y \\ \neg(X \not\Rightarrow Y) & \neg X & Y \\ \neg(X \not\Leftarrow Y) & X & \neg Y \end{array}$$

  Definiamo Form_β come l'insieme delle β-formule presenti in questa tabella (prima colonna), con $X, Y \in \mathrm{FormI}$.

Proposition 4.28. *Sia $\alpha \in \mathrm{Form}_\alpha,\ \beta \in \mathrm{Form}_\beta$ e siano $\alpha_1, \alpha_2, \beta_1, \beta_2$ le rispettive componenti, allora*

$$v(\alpha) = v(\alpha_1) \wedge v(\alpha_2)$$
$$v(\beta) = v(\beta_1) \vee v(\beta_2)$$

Theorem 4.29. *α, β-Induzione.*
Adesso presentiamo una tecnica di induzione strutturale che si basa sulle α, β-formule.
Sia $Q \subseteq \mathrm{FormI}$. Se valgono le seguenti:

1. $\mathrm{AT}, \neg\mathrm{AT} \subseteq Q$, *dove* $\neg\mathrm{AT} = \{\neg P \mid P \in \mathrm{AT}\}$
2. $z \in Q \Rightarrow \neg\neg z \in Q$
3. $\alpha \in \mathrm{Form}_\alpha,\ \alpha_1, \alpha_2 \in Q \Rightarrow \alpha \in Q$
4. $\beta \in \mathrm{Form}_\beta,\ \beta_1, \beta_2 \in Q \Rightarrow \beta \in Q$

allora

$$Q = \mathrm{FormI}$$

Proof: Sia

$$Q' = \{X \mid X, \neg X \in Q\} \subseteq Q \subseteq \mathrm{FormI}$$

Dimostriamo per induzione strutturale su FormI che $Q' = \mathrm{FormI}$.

- Per la 1., il caso base e' immediato:
$$P \in \mathrm{AT} \subseteq Q \underbrace{\Rightarrow}_{\neg\mathrm{AT}\subseteq Q} \neg P \in Q$$
$$\Rightarrow P \in Q'$$
- Supponiamo $z \in Q'$ e dimostriamo $\neg z \in Q'$:
$$z \in Q' \Rightarrow z,\ \neg z \in Q \underbrace{\Rightarrow}_{2.} \neg\neg z \in Q$$
$$\neg z \in Q,\ \ \neg\neg z \in Q \Rightarrow \neg z \in Q'$$
- Supponiamo $x, y \in Q'$, $\circ \in \mathrm{Bin_I}$ e dimostriamo $x \circ y \in Q'$.
$\circ \in \mathrm{Bin_I} \underbrace{\Rightarrow}_{[4.27,\text{pg.}15]} x \circ y \in \mathrm{Form}_\alpha \cup \mathrm{Form}_\beta$, con componenti $c_1 \in \{x,\ \neg x\}$, $c_2 \in \{y,\ \neg y\}$
$x, y \in Q' \Rightarrow c_1, c_2 \in Q \underbrace{\Rightarrow}_{3.,\ 4.} x \circ y \in Q$
analogamente $\neg(x \circ y) \in Q$
$\Rightarrow x \circ y \in Q'$

□

Definition 4.30. Sia $\mathrm{FormI}_{\alpha\beta}$ l'insieme induttivo cosi' definito:

$\mathcal{U} = (\mathrm{AT} \cup \neg\mathrm{AT} \cup \mathrm{Bin_I} \cup \{(,), \neg\})^\infty$
$B = \mathrm{AT} \cup \neg\mathrm{AT}$
$\mathcal{K} = \{\mathrm{nneg}\} \cup \{\mathrm{bin}_\alpha,\ \mathrm{bin}_\beta\}$
Dove nneg e' la funzione $\mathrm{nneg} : \mathcal{U} \longrightarrow \mathcal{U}$, $\mathrm{nneg}(P) = \neg\neg P$
bin_α e' la relazione:

$\mathrm{bin}_\alpha \subseteq \mathcal{U}^2 \times \mathcal{U}$, $A \in \mathrm{bin}_\alpha(\alpha_1, \alpha_2) \overset{\text{def}}{\Leftrightarrow} A$ e' una α-formula della tabella [4.27,pg.15] con componenti α_1, α_2
bin_β e' la relazione:

$\mathrm{bin}_\beta \subseteq \mathcal{U}^2 \times \mathcal{U}$, $A \in \mathrm{bin}_\beta(\beta_1, \beta_2) \overset{\text{def}}{\Leftrightarrow} A$ e' una β-formula della tabella [4.27,pg.15] con componenti β_1, β_2

In altri termini, $\mathrm{FormI}_{\alpha\beta}$ e' l'insieme delle formule a connettivi primari, costruite tramite $\neg\neg$ e le alpha formule delle tabelle [4.27,pg.15].

Proposition 4.31.

$$\mathrm{FormI}_{\alpha\beta} = \mathrm{FormI}$$

Di conseguenza, il thm [4.29,pg.16] si puo' riottenere applicando l'induzione strutturale (vedi [3.11,pg.5]) all'insieme $\mathrm{FormI}_{\alpha\beta}$.

Proof: Chiaramente $\mathrm{FormI}_{\alpha\beta} \subseteq \mathrm{FormI}$. A questo punto, per dimostrare l'uguaglianza basta usare la $\alpha\beta$-induzione (vedi [4.29,pg.16]). □

Proposition 4.32. *La def. [4.30,pg.17] e la prop [4.31,pg.17] servono per costruire induttivamente* FormI *usando le α e β formule. Si puo' dimostrare che questa costruzione e' libera. Per questo motivo possiamo sfruttare la prop [3.16,pg.7] e costruire senza problemi funzioni induttive su* $\mathrm{FormI}_{\alpha\beta}$ *(vedi esempio [4.33,pg.18]).*

Proof: PROOF SKETCH:Si definisce l'insieme

$$L = \{X \in \text{FormI}_{\alpha\beta} \mid \exists! \text{ costruzione per } X\}$$

e si dimostra con l'$\alpha\beta$-induzione che $L = \text{FormI}_{\alpha\beta}$ Nota: da questa dimostrazione si capisce perche' e' necessario usare la funzione $\text{nneg}(P) = \neg\neg P$, infatti, se usassimo la vecchia funzione $\text{neg}(P) = \neg P$ potremmo costruire la formula $\neg(X \vee Y)$, con $X, Y \in \text{Var}$, in due modi:

$$\neg(X \vee Y) = \text{neg}(X \vee Y) = \text{neg}(\text{bin}_\beta(X, Y))$$
$$\neg(X \vee Y) = \text{bin}_\alpha(\neg X, \neg Y)$$

□

Example 4.33. Definiamo indutttivamente, su $\text{FormI}_{\alpha\beta}$, la funzione di *rango*:

$$r(\mathbf{t}) = r(\mathbf{f}) = r(P) = r(\neg P) = 0 \ \ \forall P \in \text{AT}$$
$$r(\neg\mathbf{t}) = r(\neg\mathbf{f}) = 1$$
$$r(\neg\neg z) = r(z) + 1 \ \ \forall z \in \text{FormI}$$
$$r(\alpha) = r(\alpha_1) + r(\alpha_2) + 1 \ \ \forall \alpha \in \text{Form}_\alpha$$
$$r(\beta) = r(\beta_1) + r(\beta_2) + 1 \ \ \forall \beta \in \text{Form}_\alpha$$

Per adesso, possiamo pensare a $r(X)$ come il numero di "semplificazioni" che possiamo effettuare su X.

4.1.1 CNF e DNF

Il nostro scopo sara' quello di trovare una codifica opportuna delle formule $\text{FormI}_{\alpha\beta}$, che rende facile la loro analisi e computazione, in particolare capire se sono o non sono tautologie.

Definition 4.34. Allarghiamo l'insieme delle formule $\text{FormI}_{\alpha\beta}$ introducendo in B le parentesi $[,], \langle, \rangle$, e per ogni $n \in \mathbb{N}$ i costruttori

$$\text{disgiunzione}_n(X_1, \ldots, X_n) \stackrel{\text{def}}{=} [X_1, \ldots, X_n]$$
$$\text{congiunzione}_n(X_1, \ldots, X_n) \stackrel{\text{def}}{=} \langle X_1, \ldots, X_n \rangle$$

Infine, allarghiamo la definizione delle valutazioni considerando queste nuove formule:

$$v([X_1, \ldots, X_n]) = \begin{cases} \mathbf{t} & \exists i : \ v(X_i) = \mathbf{t} \\ F & \text{altrimenti} \end{cases}$$
$$v(\langle X_1, \ldots, X_n \rangle) = \begin{cases} \mathbf{t} & \forall i \ v(X_i) = \mathbf{t} \\ F & \text{altrimenti} \end{cases}$$

Allarghiamo inoltre la definizione di rango (vedi [4.33,pg.18]):

$$r(\langle X_1, \ldots, X_n \rangle) = r([X_1, \ldots, X_n]) = \sum_{i=1}^{n} r(X_i)$$

Definiamo l'insieme dei *letterali*:

$$\text{Lit} \stackrel{\text{def}}{=} \text{AT} \cup \neg\text{Var} = \text{Var} \cup \neg\text{Var} \cup TF$$

Definiamo le *disgiunzioni letterali*

$\underline{[X_1, \ldots, X_n]}$ e' una disgiunzione letterale (o clausola) $\overset{\text{def}}{\Leftrightarrow} \underline{[X_1, \ldots, X_n]} \in [\text{Lit}] \overset{\text{def}}{=} \left\{ \underline{[X_1, \ldots, X_n]} \mid X_i \in \text{Lit} \ \ \forall i \right\}$

Analogamente le *congiunzioni letterali*:

$$\langle \text{Lit} \rangle \overset{\text{def}}{=} \left\{ \underline{\langle X_1, \ldots, X_n \rangle} \mid X_i \in \text{Lit} \ \ \forall i \right\}$$

Possiamo infine definire le *Conjunctive Normal Form*:

$$\text{CNF} \overset{\text{def}}{=} \left\{ \underline{\langle X_1, \ldots, X_n \rangle} \mid X_i \in [\text{Lit}] \right\}$$

e le *Disjunctive Normal Form*:

$$\text{DNF} \overset{\text{def}}{=} \left\{ \underline{[X_1, \ldots, X_n]} \mid X_i \in \langle \text{Lit} \rangle \right\}$$

In altre parole, le CNF sono le formule del tipo $\langle [], [], \ldots, [] \rangle$.

Nota finale: alcune volte sara' utile considerare una disgiunzione come un insieme di formule e una CNF come un insieme di insiemi di formule. (stesso discorso per le congiunzioni e le DNF).

Remark 4.1. $r(\text{CNF}) = r(\text{DNF}) = \{0\}$

Proposition 4.35.

$$\begin{aligned}
&v([]) = \mathbf{f} \ \ \forall v \in \mathcal{V} \\
&v(\langle\rangle) = \mathbf{t}, \ \ \forall v \in \mathcal{V} \\
&\models \underline{\langle X \rangle \Leftrightarrow [X] \Leftrightarrow X} \\
&\models \underline{[X_1, \ldots, X_n] \Leftrightarrow X_1 \vee \ \ldots \ \vee X_n} \\
&\models \underline{\langle X_1, \ldots, X_n \rangle \Leftrightarrow X_1 \wedge \ \ldots \ \wedge X_n}
\end{aligned}$$

Proposition 4.36.

$$\begin{aligned}
&\models \underline{\langle [X_1, \ldots, X_n, (A \wedge B)] \rangle \Leftrightarrow \langle [X_1, \ldots, X_n, A], [X_1, \ldots, X_n, B] \rangle} \\
&\models \underline{[\langle X_1, \ldots, X_n, (A \vee B) \rangle] \Leftrightarrow [\langle X_1, \ldots, X_n, A \rangle, \langle X_1, \ldots, X_n, B \rangle]}
\end{aligned}$$

Proof: Sfruttiamo le proprieta' distributive di $\vee$, $\wedge$:
Sia $T = [X_1, \ldots, X_n]$
$\langle [X_1, \ldots, X_n, A], [X_1, \ldots, X_n, B] \rangle \Leftrightarrow \langle [T, A], [T, B] \rangle \Leftrightarrow (T \vee A) \wedge (T \vee B) \Leftrightarrow T \vee (A \wedge B) \Leftrightarrow [T, (A \wedge B)] \Leftrightarrow$
$\Leftrightarrow \langle [T, (A \wedge B)] \rangle \Leftrightarrow \langle [X_1, \ldots, X_n, (A \wedge B)] \rangle$
Analogamente per l'altro caso. □

Proposition 4.37. *Presentiamo un algoritmo di conversione di una formula $X \in \text{FormI}_{\alpha\beta}$ in una formula $Y \in \text{CNF}$.*
Per la proprieta' di commutativita', possiamo rappresentare una CNF $\langle [], [], \ldots, [] \rangle$ usando gli

insiemi $\{\{\},\{\},\ldots,\{\}\}$. *L'algoritmo e':*

$S := \{\{X\}\}$

While $\exists D \in S : \exists N \in D :\ N \backslash \mathrm{Lit} \neq \emptyset$, *ovvero fino a quando abbiamo un non letterale in* S

$S := S \backslash D$

$D := D \backslash N$

$D'' := \emptyset$

Se $N = \underline{\neg\mathbf{t}}$:

$D' := D \cup \{\mathbf{f}\}$ *ovvero, sostituiamo* $\neg\mathbf{t}$ *con* $\mathbf{f}$

Se $N = \underline{\neg\mathbf{f}}$:

$D' := D \cup \{\mathbf{t}\}$

Se $N = \underline{\neg\neg Z}$:

$D' := D \cup \{Z\}$

Se $N = \underline{\beta}$:

$D' := D \cup \{\beta_1, \beta_2\}$ *ovvero, spezziamo* β *nelle sue componenti*

Se $N = \underline{\alpha}$: (1)

$D' := D \cup \{\alpha_1\}$

$D'' := D \cup \{\alpha_2\}$

$S := S \cup \{D'\}$

Se $D'' \neq \emptyset$

$S := S \cup \{D''\}$

Nota: la (1) *serve per trasformare una formula del tipo*

$$\langle [X_1, \ldots, X_n, \alpha] \rangle$$

in

$$\langle [X_1, \ldots, X_n, \alpha_1], [X_1, \ldots, X_n, \alpha_2] \rangle$$

Ecco un esempio di esecuzione:

$\langle [(P \supset (Q \supset R)) \supset ((P \supset Q) \supset (P \supset R))] \rangle$

$\langle [\neg(P \supset (Q \supset R)), ((P \supset Q) \supset (P \supset R))] \rangle$

$\langle [\neg(P \supset (Q \supset R)), \neg(P \supset Q), (P \supset R)] \rangle$

$\langle [\neg(P \supset (Q \supset R)), \neg(P \supset Q), \neg P, R] \rangle$

$\langle [P, \neg(P \supset Q), \neg P, R], [\neg(Q \supset R), \neg(P \supset Q), \neg P, R] \rangle$

$\langle [P, P, \neg P, R], [P, \neg Q, \neg P, R], [\neg(Q \supset R), \neg(P \supset Q), \neg P, R] \rangle$

$\langle [P, P, \neg P, R], [P, \neg Q, \neg P, R], [Q, \neg(P \supset Q), \neg P, R],$
$[\neg R, \neg(P \supset Q), \neg P, R] \rangle$

$\langle [P, P, \neg P, R], [P, \neg Q, \neg P, R], [Q, P, \neg P, R], [Q, \neg Q, \neg P, R],$
$[\neg R, \neg(P \supset Q), \neg P, R] \rangle$

$\langle [P, P, \neg P, R], [P, \neg Q, \neg P, R], [Q, P, \neg P, R], [Q, \neg Q, \neg P, R],$
$[\neg R, P, \neg P, R], [\neg R, \neg Q, \neg P, R] \rangle$

Proof:

⟨1⟩1. Dimostriamo che la proposizione:

$P(S, X)$ = $'' \models \underline{S \Leftrightarrow X}$, S e' una congiunzione di disgiunzioni, cioe' e' del tipo $\langle [], [], \ldots, [] \rangle''$ e' un *invariante* del loop while, ovvero se P e' vera prima del while, allora lo sara' alla fine di ogni ciclo.

Prima del loop $S = \langle [X] \rangle$ e quindi la tesi e' vera.

Nel loop stiamo sostituendo sottoformule di S con formule equivalenti, e quindi per il thm di sostituzione (vedi [4.23,pg.14]) la tautologia $\models \underline{S \Leftrightarrow X}$ viene preservata. In particolare, nella (1) preserviamo l'equivalenza grazie alla prop [4.36,pg.19]. Inoltre, proprio per costruzione abbiamo sempre una congiunzione di disgiunzioni.

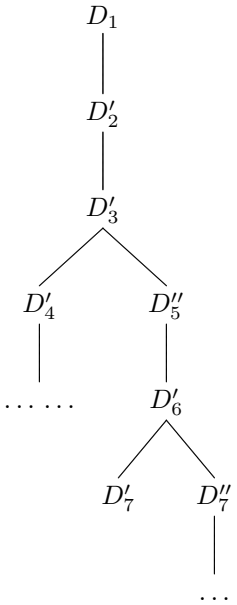
Quindi, poiche' $P(S, X)$ e' invariante, per il principio di *induzione per i loop while* sara' vera anche quando il while termina.

Il while termina quando ci sono solo letterali e poiche' $P(S, X)$, S sara' una congiunzione di disgiunzioni solo con letterali, ovvero una CNF.

Resta da dimostrare che il while termina.

Osserviamo che per la [4.1,pg.19], solo quando il while termina si ha $r(S) = 0$.

Supponiamo per assurdo che non termini. Osserviamo che alla sostituzione $(k+1)$-esima, si ha $r(S_{k+1}) < r(S_k)$ (per come e' definito $r()$ sulle disgiunzioni). Inoltre, possiamo rappresentare ogni sostituzione con il seguente albero:



Nota: i D' e D'' nascono con la sostituzione (1). Un solo D' nasce con tutte le altre sostituzioni.

Poiche' il while non termina, l'albero e' infinito, ma a diramazione finita. Quindi, per il lemma di Konig (vedi [2.2,pg.1]), esistera' un ramo infinito:

$$D_1, D_2, \ldots$$

e per quanto detto prima avremo:

$$r(D_1) > r(D_2) > \ldots$$

ma questo e' assurdo contro $r(D_i) \in \mathbb{N}$.

□

4.2 Tableaux

Per dimostrare che X e' una tautologia utilizzeremo l'algoritmo che converte $\neg X$ in una DNF (vedi [4.37,pg.19]), ma ci fermeremo non appena ci accorgiamo che una congiunzione contiene $Y, \neg Y$ (oppure contiene $\mathbf{f}$). In altre parole, quando la formula diventa del tipo $[\langle \ldots \rangle, \ \ldots, \ \langle \ldots, Y, \neg Y, \ldots \rangle, \ \ldots, \ \langle \ldots \rangle]$ l'algoritmo elimina $\langle \ldots, Y, \neg \mathbf{t} \rangle$ perche' ha trovato la contraddizione $Y \wedge \neg Y$. Quando la formula diventa la disgiunzione vuota $[]$, l'algoritmo si arresta: $\neg X$ e' falsa, e quindi X e' vera.

Questa tecnica si puo' codificare usando degli alberi $\mathcal{T}(X)$ che chiameremo *tableaux* di X.

Definition 4.38. Sia dato l'insieme di formule $X = \{X_1, \ldots, X_n\} \subseteq \mathrm{Form}\mathrm{I}_{\alpha\beta}$. Definiamo per induzione $\mathcal{T}(X)$, l'insieme dei tableaux costruiti su X: $\mathcal{T}(X)$ e' il piu' piccolo insieme tale che:

- L'albero formato dal solo ramo $X_1, X_2, \ldots, X_n$

  e' un tableaux, ovvero sta' in $\mathcal{T}(X)$

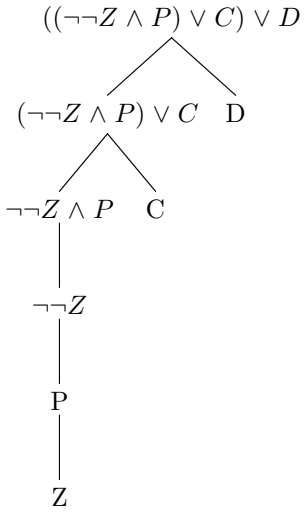
- Se $T \in \mathcal{T}(X)$ allora $T^* \in \mathcal{T}(X)$, dove T^* e' la copia di T a cui e' stata applicata una delle seguenti regole di espansione[3]:

  $T^* := T$
  Sia N un nodo di un ramo terminato r di T^*
  Sia M l'ultimo nodo di r

  1. **Se** $N = \underline{\neg \mathbf{t}}$:
     $r := r, \ \mathbf{f}$ aggiungiamo al ramo r il nodo $\mathbf{f}$
  2. **Se** $N = \underline{\neg \mathbf{f}}$:
     $r := r, \ \mathbf{t}$
  3. **Se** $N = \underline{\neg\neg Z}$:
     $r := r, \ Z$
  4. **Se** $N = \underline{\alpha}$:
     $r := r, \alpha_1, \alpha_2$
  5. **Se** $N = \underline{\beta}$: (1)
     $r := r, [\beta_1, \beta_2]$ ovvero, aggiungiamo ad M i figli β_1, β_2

Example 4.39. Il seguente albero e' un tableaux per la formula $((\neg\neg Z \wedge P) \vee C) \vee D$:

[3] sono le stesse regole usate dall'algoritmo DNF (vedi [4.37,pg.19])



Definition 4.40. Sia $T \in \mathcal{T}(X)$ e $r \in \text{ramo}_t(T)$,

$$r \text{ e' chiuso da } Y \in \text{FormI}_{\alpha\beta} \overset{\text{def}}{\Leftrightarrow} (Y, \neg Y \in \text{nodo}(r)) \text{ oppure } \mathbf{f} \in \text{nodo}(r)$$
$$r \text{ e' chiuso } \overset{\text{def}}{\Leftrightarrow} \exists Y \in \text{FormI}_{\alpha\beta} : \ r \text{ e' chiuso da } Y$$
$$r \text{ e' atomicamente chiuso } \overset{\text{def}}{\Leftrightarrow} \exists P \in \text{Var} : \ \ r \text{ e' chiuso da } P$$
$$T \text{ e' (atomicamente) chiuso } \overset{\text{def}}{\Leftrightarrow} \forall r \in \text{ramo}_t(T) \ \ r \text{ e' (atomicamente) chiuso}$$
$$\vdash X \overset{\text{def}}{\Leftrightarrow} \exists T \in \mathcal{T}(\neg X) : \ \ T \text{ e' chiuso}$$

In pratica, un ramo chiuso e' una congiunzione in cui abbiamo trovato una contraddizione.

$$r \text{ e' soddisfacibile } \overset{\text{def}}{\Leftrightarrow} \exists v \in \mathcal{V} : \ \ v(x) = T \ \ \forall x \in \text{nodo}(r)$$
$$T \text{ e' soddisfacibile } \overset{\text{def}}{\Leftrightarrow} \exists r \in \text{ramo}_t(T) : \ r \text{ e' soddisfacibile}$$

Lemma 4.41.

$$T \in \mathcal{T}(X) \ \textit{e' soddisfacibile} \Rightarrow T^* \ \textit{e' soddisfacibile}$$

dove T^* e' l'albero ottenuto da T usando le regole di espansione (vedi [4.38,pg.22]).

Proof:

$$T \text{ sodd.} \ \Rightarrow \exists r \in \text{ramo}_t(T), \ \exists v \in \mathcal{V} : \ v(r) = T$$

Se $r \in \text{ramo}(T^*)$, non abbiamo nulla da dimostrare.
Supponiamo allora che $r \notin \text{ramo}(T^*)$. Poiche' T^* e' una copia di T modificata, quest'ultimo caso si puo' verificare solo se T^* e' stato ottenenuto da T sostituendo r con r', tramite le regole di espansione (vedi [4.38,pg.22]). Esaminiamo i vari casi.
Case: r' e' stato ottenuto con la regola 1.
Quindi,

$$\exists N \in r : \ N = \neg\mathbf{t}$$
$$r' = r, \mathbf{f}$$

Allora,

$$v(r) = \mathbf{t} \Rightarrow v(N) = \mathbf{t} \Leftrightarrow \neg v(\mathbf{t}) = \mathbf{t} \Leftrightarrow \mathbf{f} = \mathbf{t}$$

assurdo. Quindi questo caso non si verifica.

CASE: r' e' stato ottenuto con la regola 2.

Quindi,

$$\exists N \in r : \ N = \neg\mathbf{f}$$
$$r' = r, \mathbf{t}$$

Allora,

$$v(r) = \mathbf{t}, \ v(\mathbf{t}) = \mathbf{t} \Rightarrow v(r') = \mathbf{t}$$

quindi r' e' soddisfatto.

CASE: r' e' stato ottenuto con la regola 3.

Quindi,

$$\exists N \in r : \ N = \neg\neg Z$$
$$r' = r, Z$$

Allora,

$$v(r) = \mathbf{t} \Rightarrow v(N) = \mathbf{t} \Leftrightarrow V(Z) = \mathbf{t}$$
$$v(r) = v(Z) = \mathbf{t} \Rightarrow v(r') = \mathbf{t}$$

quindi r' e' soddisfatto.

CASE: r' e' stato ottenuto con la regola 4.

Quindi,

$$\exists N \in r : \ N = \alpha$$
$$r' = r, \alpha_1, \alpha_2$$

Allora,

$$v(r) = \mathbf{t} \Rightarrow v(N) = \mathbf{t} \Leftrightarrow (v(\alpha_1) \wedge v(\alpha_2)) = \mathbf{t} \Leftrightarrow v(\alpha_1) = v(\alpha_2) = \mathbf{t}$$
$$v(r) = v(\alpha_1) = v(\alpha_2) = \mathbf{t} \Rightarrow v(r') = \mathbf{t}$$

quindi r' e' soddisfatto.

CASE: r' e' stato ottenuto con la regola 5.

Quindi,

$$\exists N \in r : \ N = \beta$$
$$r'_1 = r, \beta_1 \text{ oppure } r'_2 = r, \beta_2$$

In questo caso, la regola ha prodotto due nuovi rami in $\mathbf{t}^*$.

$$v(r) = \mathbf{t} \Rightarrow v(N) = \mathbf{t} \Leftrightarrow (v(\beta_1) \vee v(\beta_2)) = \mathbf{t} \Leftrightarrow v(\beta_1) = \mathbf{t} \vee v(\beta_2) = \mathbf{t}$$

In un caso, o nell'altro almeno uno tra r'_1, r'_2 e' soddisfatto.

Possiamo cosi' concludere, che in ogni caso possibile, esiste almeno un ramo soddisfatto in T^*. □

Theorem 4.42. *di correttezza. Sia* $X \in \mathrm{Form\,I}_{\alpha\beta}$,

$$\vdash X \Rightarrow \models X$$

Questo teorema garantisce la validita' del metodo dei Tableaux: e' sufficiente trovare un tableaux chiuso per $\neg X$ *per poter affermare che* X *e' una tautologia.*

Proof: Procediamo per assurdo: $\exists v \in \mathcal{V} : \ v(X) = \mathbf{f}$, ovvero

$$v(\neg X) = \mathbf{t}$$

Quindi il tableaux base $\{\neg X\}$, quello da cui si costruiscono tutti gli altri tableaux, e' per definizione soddisfacibile (il suo unico ramo e' soddisfacibile). Allora, per il lemma [4.41,pg.23] ogni $T^* \in \mathcal{T}(\neg X)$ e' soddisfacibile.

Per Hp $\exists T \in \mathcal{T}(\neg X)$ chiuso. Quindi $\forall r \in \text{ramo}_t(T)$: r e' chiuso. In particolare, il ramo r' che rende soddisfacibile T e' chiuso. Quindi

$$r' \text{ chiuso } \Rightarrow Y, \neg Y \in \text{nodo}(r') \text{ oppure } \mathbf{f} \in r' \underbrace{\Rightarrow}_{r' \text{ soddisfacibile}} v(Y) = v(\neg Y) \text{ oppure } v(\mathbf{f}) = \mathbf{t}$$

In entrambi i casi si ha un assurdo. □

Definition 4.43. Un Tableaux $T \in \mathcal{T}(X)$ e' ristretto $\overset{\text{def}}{\Leftrightarrow}$

$\forall r \in \text{ramo}(T)$ $\forall n \in \text{nodo}(r)$ n e' stato usato una volta sola per ottenere qualche altro $m \in r$

Remark 4.2. Il lemma [4.41,pg.23] e il thm [4.42,pg.24] valgono anche se usiamo solo tableaux ristretti, infatti la dimostrazione del lemma rimane invariata.

4.3 Risoluzioni

Le *risoluzioni* sono analoghe ai tableaux. Nei tableaux usavamo la DNF, mentre nelle risoluzioni espandiamo la formula con la CNF ($\langle[]\rangle$). La codifica avviene tramite liste e non tramite alberi. Al posto della chiusura, abbiamo una regola che permette di rimuovere pezzi di formula inutili. Quando arriviamo alla formula [] possiamo dire che $\neg X$ e' falsa.

Definition 4.44. Sia dato l'insieme di formule $X = \{X_1, \ldots, X_n\} \subseteq \text{FormI}_{\alpha\beta}$. Definiamo per induzione $\mathscr{R}(X)$, l'insieme delle risoluzioni costruite su X: $\mathscr{R}(X)$ e' il piu' piccolo insieme tale che:

- La lista non ordinata formata dalla seguenti disgiunzioni:

$$\begin{array}{l}[X_1] \\ [X_2] \\ \vdots \\ [X_n]\end{array}$$

  e' una risoluzione, ovvero sta' in $\mathscr{R}(X)$

- Se $R \in \mathscr{R}(X)$ allora $R^* \in \mathscr{R}(X)$, dove R^* e' la copia di R a cui e' stata applicata una delle

seguenti regole:

$R^* := R$

Sia N un elemento della lista R^*

Sia M un elemento della disgiunzione N

1. **Se** $M = \underline{\neg \mathbf{t}}$:
   $R^* := R^*,\ N(M|\mathbf{f})$ ovvero,
   aggiungiamo alla lista R^* la disgiunzione $N(M|\mathbf{f})$ che e' ottenuta da N sostituendo l'elemento M con la formula $\mathbf{f}$
2. **Se** $M = \underline{\neg \mathbf{f}}$:
   $R^* := R^*,\ N(M|\mathbf{t})$
3. **Se** $M = \underline{\neg\neg Z}$:
   $R^* := R^*,\ N(M|Z)$
4. **Se** $M = \underline{\alpha}$:
   $R^* := R^*, N(M|\alpha_1), N(M|\alpha_2)$
5. **Se** $M = \underline{\beta}$:
   $R^* := R^*, N(M|\beta_1, \beta_2)$
6. **Se** $\exists N' \in R^* :\ N' \neq N,\ \exists M' \in N'$ t.c. $\exists \underline{X} \in M \wedge \underline{\neg X} \in M'$:
   $R^* := R^* \setminus \{M, M'\}, M(X|) \cup M'(X|)$
7. **Se** $M = \underline{\mathbf{f}}$:
   $R^* := R^*, N(M|)$ ovvero, togliamo la $\mathbf{f}$ dalla disgiunzione N

Chiameremo le $1. - 6.$ regole di risoluzione.
Le $1. - 5.$ sono chiamate *regole di espansione* per risoluzioni.
Le regole $6. - 7.$ sono chiamate *regole risolutive*. La 6. dice che se troviamo due disgiunzioni di tipo $[Y_1, \ldots, Y_i, X, Y_{i+2}, \ldots, Y_n], [Z_1, \ldots, Z_j, \neg X, Z_{j+2}, \ldots, Z_m]$ allora le possiamo sostiture con l'unica disgiunzione $[Y_1, \ldots, Y_i, Y_{i+2}, \ldots, Y_n, Z_1, \ldots, Z_i, Z_{i+2}, Z_m]$

Example 4.45. La seguente lista e' una risoluzione per la formula $((\neg\neg Z \wedge P) \vee C) \vee \neg Z$:

$$
\begin{aligned}
&[((\neg\neg Z \wedge P) \vee C) \vee \neg Z]\\
&[(\neg\neg Z \wedge P) \vee C, \neg Z]\\
&[\neg\neg Z \wedge P, C, \neg Z]\\
&[\neg\neg Z, C, \neg Z]\\
&[P, C, \neg Z]\\
&[Z, C, \neg Z]\\
&[P, C, \neg Z]
\end{aligned}
$$

Definition 4.46. Sia $R \in \mathscr{R}(X)$,

$$
\begin{aligned}
&R \text{ e' chiusa } \overset{\text{def}}{\Leftrightarrow} \exists N \in R :\ \ N = []\\
&\vdash X \overset{\text{def}}{\Leftrightarrow} \exists R \in \mathscr{R}(\neg X) :\ \ R \text{ e' chiuso}
\end{aligned}
$$

In pratica, una risoluzione e' chiusa quando un suo elemento e' [], e quindi l'intera risoluzione, che e' una congiunzione, e' falsa.

$$R \text{ e' soddisfacibile } \overset{\text{def}}{\Leftrightarrow} \exists v \in \mathcal{V} : \ v(N) = \mathbf{t} \ \ \forall N \in R$$

Lemma 4.47.

$$R \in \mathscr{R}(X) \ \textit{e' soddisfacibile} \Rightarrow R^* \ \textit{e' soddisfacibile}$$

dove R^* e' l'albero ottenuto da R usando le regole descritte in vedi [4.44,pg.25].

Proof:

$$R \text{ sodd.} \ \Rightarrow \exists v \in \mathcal{V} : \ v(N) = \mathbf{t} \ \ \forall N \in R$$

Dobbiamo dimostrare che

$$v(N') = \mathbf{t} \ \ \forall N' \in R^*$$

In linea di principio, la dimostrazione procede cosi': sia $N = [Y_1, \ldots, Y_n]$ e sia $N' = [Y_1, \ldots, Y_i', \ldots, Y_n]$ ottenuta da N sostituendo Y_i con Y_i',

$$v(N') = \mathbf{t} \Rightarrow \exists j : \ v(Y_j) = \mathbf{t}$$

$$i \neq j \Rightarrow v(N') = \mathbf{t}$$

Se invece $i = j$ si dimostra che $v(Y_i') = \mathbf{t}$ e quindi $v(N') = \mathbf{t}$

Esaminiamo in dettaglio solo alcuni casi.

Case: N' e' stato ottenuto con la regola 5.

Quindi,

$$N = [Y_1, \ldots, Y_i = \beta, \ldots, Y_n]$$
$$N' = [Y_1, \ldots, \beta_1, \beta_2, \ldots, Y_n]$$
$$v(\beta) = \mathbf{t} \Rightarrow v(\beta_1) = \mathbf{t} \vee v(\beta_2) = \mathbf{t} \Rightarrow v(N') = \mathbf{t}$$

Case: r' e' stato ottenuto con la regola 4.

Quindi,

$$N = [Y_1, \ldots, Y_i = \alpha, \ldots, Y_n]$$
$$N' = [Y_1, \ldots, \alpha_1, \ldots, Y_n] \in R^*$$
$$N'' = [Y_1, \ldots, \alpha_2, \ldots, Y_n] \in R^*$$
$$v(\alpha) = \mathbf{t} \Rightarrow v(\alpha_1) = \mathbf{t} \wedge v(\alpha_2) = \mathbf{t} \Rightarrow v(N') = v(N'') = \mathbf{t}$$

Case: r' e' stato ottenuto con la regola 6.

Questo vuol dire che in R avevamo due disgiunzioni

$$N_1 = [Y_1, \ldots, Y_n, X], \ N_2 = [Z_1, \ldots, Z_m, \neg X]$$

e in R^* abbiamo

$$N' = [Y_1, \ldots, Y_n, Z_1, \ldots, Z_m]$$

Per Hp $v(N_1) = v(N_2) = \mathbf{t}$

Case: $v(X) = \mathbf{t}$

$$v(X) = \mathbf{t} \Rightarrow v(\neg X) = \mathbf{f}$$
$$\begin{cases} v(\neg X) = \mathbf{f} \\ v(N_2) = \mathbf{t} \end{cases} \Rightarrow \exists 1 \leq j \leq m : \ v(Z_j) = \mathbf{t} \Rightarrow v(N') = \mathbf{t}$$

Case: $v(X) = \mathbf{f}$

$$\begin{cases} v(X) = \mathbf{f} \\ v(N_1) = \mathbf{t} \end{cases} \Rightarrow \exists 1 \leq j \leq n : \ v(Y_j) = \mathbf{t} \Rightarrow v(N') = \mathbf{t}$$

In un caso, o nell'altro tutti gli elementi di R^* sono soddisfatti. □

Theorem 4.48. *di correttezza. Sia* $X \in \mathrm{FormI}_{\alpha\beta}$,

$$\vdash X \Rightarrow \models X$$

Proof: Procediamo per assurdo: $\exists v \in \mathcal{V} : \ v(X) = \mathbf{f}$, ovvero
$$v(\neg X) = \mathbf{t}$$
Quindi la risoluzione base $[\neg X]$, quella da cui si costruiscono tutte le altre risoluzioni, e' per definizione soddisfacibile. Allora, per il lemma [4.47,pg.27] ogni $R^* \in \mathscr{R}(\neg X)$ e' soddisfacibile. Per Hp $\exists R \in \mathscr{R}(\neg X)$ chiuso. Quindi $\underline{[]} \in R$. Ma R deve essere soddisfacibile: $\forall N \in R \ v(N) = \mathbf{t}$, in particolare $v([]) = \mathbf{t}$. E questo e' assurdo per definizione di v. □

4.4 Completezza

In questa sezione dimostreremo

$$\vdash X \Rightarrow \models X$$

sia per i Tableaux che per le Risoluzioni. In altre parole, se abbiamo una tautologia nella nostra logica proposizionale, allora i tableaux e le risoluzioni troveranno, prima o poi, una dimostrazione per esssa. Questo non e' vero per qualsiasi logica. Fatto questo, grazie ai teoremi di correttezza [4.42,pg.24], [4.48,pg.28], potremmo dire:

$$\models X \Leftrightarrow \vdash X$$

Definition 4.49. Sia $H \subseteq \mathrm{FormI}_{\alpha\beta}$, H e' un insieme di *Hintikka* $\overset{\text{def}}{\Leftrightarrow}$

1. $P \in \mathrm{Var} \Rightarrow (P \notin H \ \text{ oppure } \ \neg P \notin H)$
2. $\neg\mathbf{t}, \ \mathbf{f} \notin H$
3. $\neg\neg Z \in H \Rightarrow Z \in H$
4. $\alpha \in H \Rightarrow \alpha_1, \alpha_2 \in H$
5. $\beta \in H \Rightarrow \beta_1 \in H$ oppure $\beta_2 \in H$

Le 1., 2. sono le condizioni di *consistenza*. La 3., 4., 5. sono le condizioni di saturazione all'ingiu'.

Proposition 4.50.

$$H \ \textit{Hintikka} \ \Rightarrow H \ \textit{e' soddisfacibile, ovvero} \ \exists v \in \mathcal{V} : \ \ v(H) = \mathbf{t}$$

Proof: Proof sketch:Costruiamo una valutazione che soddisfa H.
Per far cio' bastera' assegnare $\mathbf{t}$ ad ogni lettera in H. Gli assiomi di H garantiranno che tutte le formule costruite da quelle lettere sono $\mathbf{t}$ Sia data la seguente funzione:

$$f : \mathrm{Var} \longrightarrow \{\mathbf{t}, \mathbf{f}\}$$

$$f(P) = \begin{cases} \mathbf{t} & P \in H \\ \mathbf{f} & P \notin H \end{cases}$$

Per la prop [4.4,pg.9] si ha

$$\exists! v \in \mathcal{V} : \ \ \forall p \in \mathrm{Var} \ \ v(p) = f(p)$$

Sia $X \in H$. Dimostriamo per induzione sul rango (vedi [4.33,pg.18]) che $v(X) = \mathbf{t}$, ovvero

$$\forall n \in \mathbb{N} \ \ \forall X \in H : \ r(X) = n \ \ v(X) = \mathbf{t}$$

CASE: $n = 0$
  CASE: $X = P \in \text{Var}$

$$v(P) = f(P) \underbrace{=}_{P=X \in H} \mathbf{t}$$

  CASE: $X = \neg P \in \neg\text{Var}$
    CASE: $P \in H$
      Questo caso non si verifica, perche' gli assiomi di Hintikka richiedono:

$$P \in H \Rightarrow \neg P \notin H$$

    CASE: $P \notin H$

$$v(P) = f(P) \underbrace{=}_{P \notin H} \mathbf{f}$$
$$v(\neg P) = \neg v(P) = \mathbf{t}$$

  CASE: $X = \mathbf{t}$

$$v(X) = \mathbf{t}$$

  CASE: $X = \mathbf{f}$
    Questo caso e' assurdo perche' per gli assiomi di Hintikka $\mathbf{f} \notin H$.
CASE: $n = 1,\ X = \neg\mathbf{t},\ X = \neg\mathbf{f}$
  Come prima.
CASE: $m < n \Rightarrow n$
  CASE: $X = \neg\neg Z$

$$X \in H \underbrace{\Rightarrow}_{[4.49,\text{pg.}28]} Z \in H\ (1)$$
$$v(X) = \neg\neg v(Z) \underbrace{=}_{(1),\ \text{Hp induttiva}} \neg\neg\mathbf{t} = \mathbf{t}$$

  CASE: $X = \alpha$

$$X \in H \Rightarrow \alpha_1, \alpha_2 \in H$$

    e quindi applichiamo l'Hp induttiva su α_1, α_2:

$$v(\alpha_1) = v(\alpha_2) = \mathbf{t}$$
$$v(X) = v(\alpha) = (v(\alpha_1) \wedge v(\alpha_2)) = \mathbf{t}$$

  CASE: $X = \beta$
    Analogamente a prima.

□

Definition 4.51. Sia $C \subseteq \mathcal{P}(\text{FormI}_{\alpha\beta})$. C e' una *proprieta' di consistenza* $\overset{\text{def}}{\Leftrightarrow} \forall S \in C$

1. $P \in \text{Var} \Rightarrow P \notin S$ oppure $\neg P \notin S$
2. $\mathbf{f},\ \neg\mathbf{t} \notin S$
3. $\neg\neg Z \in S \Rightarrow S \cup \{Z\} \in C$
4. $\alpha \in S \Rightarrow S \cup \{\alpha_1, \alpha_2\} \in C$
5. $\beta \in S \Rightarrow S \cup \{\beta_1\}$ oppure $S \cup \{\beta_2\} \in C$

Theorem 4.52. *di completezza per i Tableaux ristretti.*

$$\models X \Rightarrow \vdash X$$

Dove con $\vdash X$ *intendiamo che* $\exists T \in \mathcal{T}(\neg X)$ *chiuso e ristretto.*

Proof: Procediamo per assurdo.
I tableaux ristretti per $\neg X$ esistono, quindi supponendo per assurdo che $\not\vdash X$ si ha

$$T \text{ tableaux ristretto di } \neg X \;\Rightarrow T \text{ non chiuso per } \neg X$$

Quindi, qualsiasi tableaux che costruiamo a partire da $\neg X$ non verra' mai chiuso.
Poiche' i tableaux che costruiamo sono ristretti, sono anche finiti.
Quindi ogni T sara' un albero finito ma non chiuso, ovvero $\exists r \in \text{ramo}\,(T): \quad r$ non chiuso
E' facile vedere che

$$\{n \mid n \in \text{nodo}\,(r)\}$$

e' un insieme di Hintikka.
Quindi per il lemma [4.50,pg.28], r e' soddisfacibile:

$$\exists v \in \mathcal{V}: \quad v(r) = \mathbf{t} \Leftrightarrow \forall n \in \text{nodo}\,(r) \quad v(n) = \mathbf{t}$$

In particolare, poiche' r e' un ramo di $T \in \mathcal{T}(\neg X)$, abbiamo

$$v(\neg X) = \mathbf{t} \Rightarrow v(X) = \mathbf{f}$$

E questo e' assurdo contro $\models X$. □

Proposition 4.53.

$$\begin{cases} S \in C \\ |S| < \infty \end{cases} \Rightarrow S \textit{ e' soddisfacibile}$$

Proof: Definiamo il seguente algoritmo:

$S' := S$
While $S \neq \emptyset$
  Sia $X \in S$
  $S := S \setminus \{X\}$
  Se $X \in \text{Var} \cup \neg\text{Var} \cup \{\mathbf{t}, \neg\mathbf{f}\}$:
    continua
  Se $X := \neg\neg Z$:
    $S' := S' \cup \{Z\}$
    $S := S \cup \{Z\}$
  Se $X := \alpha$:
    $S' := S' \cup \{\alpha_1, \alpha_2\}$
    $S := S \cup \{\alpha_1, \alpha_2\}$
  Se $X := \beta$:
    Sia $b \in \{\beta_1, \beta_2\}: \; S' \cup \{b\} \in C$
    $S' := S' \cup \{b\}$
    $S := S \cup \{b\}$
Ritorna S'

In pratica, togliamo una formula da S, la processiamo e troviamo delle sottoformule "buone", che riaggiungiamo in S per poi processarle in futuro. In S' teniamo tutto. Pian piano S si svuota e l'algoritmo finisce.

1. Poiche' S e' finito, ragionando sul rango delle formule che aggiungiamo in S e usando il lemma di Konig cosi' come abbiamo fatto nella dimostrazione di [4.37,pg.19], concludiamo che l'algoritmo termina. (In poche parole, togliamo da S una formula e aggiungiamo formule con rango piu' piccolo, quindi a un certo punto ci fermiamo).
2. $S \subseteq S'$ e' vera ad ogni iterazione del while.
3. $S' \in C$ ad ogni iterazione del while (e' un invariante del loop while). Ragioniamo per induzione: inizialmente $S' = S \in C$. Sia S' prima del loop e $\overline{S}'$ quello ottenuto dopo il loop. Analizziamo solo un if, gli altri sono analoghi:

   Per Hp induttiva $S' \in C$

   $X \in S \subseteq S' \Rightarrow X \in S'$

   $X = \beta \in S', \;\; S' \in C \Rightarrow S' \cup \{\beta_1\}$ oppure $S' \cup \{\beta_2\} \in C$

   Per come e' costruito l'algoritmo:

   $S' \cup \{\beta_1\} \in C, \;\; S' \cup \{\beta_2\} \notin C \Rightarrow \overline{S}' = S' \cup \{\beta_1\} \in C$

   $S' \cup \{\beta_2\} \in C, \;\; S' \cup \{\beta_1\} \notin C \Rightarrow \overline{S}' = S' \cup \{\beta_2\} \in C$

   $S' \cup \{\beta_2\} \in C, \;\; S' \cup \{\beta_1\} \in C \Rightarrow \overline{S}'$ e' uno dei due, ma in ogni caso $\overline{S}' \in C$
4. S', quello finale restituito dall'algoritmo, e' un insieme di Hintikka.

Possiamo cosi' concludere che

$$S' \text{ hintikka} \underbrace{\Rightarrow}_{\text{[4.50,pg.28]}} S' \text{ soddisfacibile} \underbrace{\Rightarrow}_{S \subseteq S'} S \text{ soddisfacibile}$$

□

Theorem 4.54. *di completezza per tableaux qualsiasi.*

$$\models X \Rightarrow \vdash X$$

Proof: Supponiamo per assurdo che $\nexists T \in \mathcal{T}(\neg X) : \; T$ chiuso.
Poniamo

$$C = \{W \subseteq \text{FormI}_{\alpha\beta} \mid |W| < \infty, \;\; \nexists T \in \mathcal{T}(W) : \; T \text{ chiuso}\}$$

Quindi $\{\neg X\} \in C$.

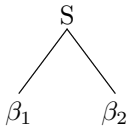
⟨1⟩1. C e' una proprieta' di consistenza

Sia $S = \{s_1, \ldots, s_n\} \in C$.

1. Non e' vero che $P, \neg P \in S$, altrimenti il tableaux formato dal solo ramo:

   $$\begin{array}{l} s_1 \\ s_2 \\ \vdots \\ s_n \\ P \\ \neg P \end{array}$$

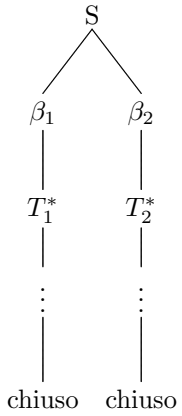
   sarebbe un tableaux chiuso per S
2. Analogamente, $\mathbf{f}$, $\neg\mathbf{t}$ non possono stare in S
3. Supponiamo che $\beta \in S$ e supponiamo per assurdo che $S_1 = S \cup \{\beta_1\}$, $S_2 = S \cup \{\beta_2\} \notin C$, ovvero che esistono due tableaux T_1, T_2 per S_1, S_2 chiusi.
   Costruiamo un tableaux T per S applicando una volta la regola di espansione a β:



T e' formato dai due rami

$$r_1 = S, \beta_1$$
$$r_2 = S, \beta_2$$

Il ramo r_1 e' esattamente l'inizio del tableaux T_1. Quindi, applicando le stesse regole usate per ottenere T_1 da $S \cup \{\beta_1\}$, possiamo espandere il ramo r_1, ottenendo un albero che coincide con T_1. E quindi questo albero e' chiuso.
Applichiamo la stessa procedura a r_2
Il tableaux cosi' ottenuto da T e' chiuso, in quanto formato da due tableaux chiusi. Graficamente:



Quindi $S \notin C$. Assurdo.

4. Si procede analogamente per gli altri casi.

$\langle 1 \rangle 2$. Q.E.D.

$$\neg X \in C \underbrace{\Rightarrow}_{[4.53,\text{pg.}30]} \neg X \text{ soddisfacibile } \Rightarrow \exists v \in \mathcal{V} : \ v(\neg X) = \mathbf{t} \Leftrightarrow v(X) = \mathbf{f}$$

e questo e' assurdo contro

$$\models X$$

□

Passiamo adesso alle risoluzioni.

Definition 4.55. Sia S un insieme di disgiunzioni del tipo $[*, \ldots, *]$.
Sia $D = (D_1, \ldots, D_n)$ una lista di disgiunzioni del tipo $[*, \ldots, *]$.

$D \in \mathcal{D}(S) \overset{\text{def}}{\Leftrightarrow} D$ e' una derivazione da $S \overset{\text{def}}{\Leftrightarrow}$

$\overset{\text{def}}{\Leftrightarrow} \forall i \ \ D_i \in S$ oppure si puo' costruire D_i applicando le regole [4.44,pg.25] ad alcuni D_j con $j < i$

d e' derivabile da $S \overset{\text{def}}{\Leftrightarrow} \exists D \in \mathcal{D}(S) : \ \ d$ e' l'ultimo elemento di D

Remark 4.3. Una $D \in \mathcal{D}(\{[X_1], \ldots, [X_n]\})$ si puo' considerare come una risoluzione si $\{X_1, \ldots, X_n\}$ (Nota[4])

[4] in realta' non e' esattamente una risoluzione perche' $[X_i]$ puo' essere ripetuto arbitrariamente in D. Nelle risoluzioni non abbiamo una regola che permette di copiare un elemento dell'insieme di partenza

Le derivazioni sono quindi piu' generali delle risoluzioni, in quanto partono da un qualsiasi insieme di disgiunzioni.

Definition 4.56. Sia $d = [X_1, \ldots, X_n]$ una disgiunzione. Diremo che le due disgiunzioni

$$[X_1, \ldots, X_n, Y], \quad [X_1, \ldots, X_n]$$

sono degli Y-ingrandimenti di d. La prima la chiamereo Y-ingrandimento effettivo. La seconda la chiameremo Y-ingrandimento non effettivo.
Sia S un insieme di disgiunzioni. S' e' un Y-ingrandimento di $S \overset{\text{def}}{\Leftrightarrow}$

$$\exists f : S \longrightarrow S' \text{ biezione } t.c. \qquad f(d) \text{ e' un } Y\text{-ingrandimento di } d$$

In pratica sostituiamo tutte le disgiunzioni di S con degli Y-ingrandimenti e otteniamo S'. Poniamo:

$$[S]_Y \overset{\text{def}}{=} \{\text{tutti gli } Y \text{ ingrandimenti di } S\}$$

Lemma 4.57. *Sia S un insieme di disgiunzioni. $S' \in [S]_Y$.*

$$d \text{ e' derivabile da } S \Rightarrow \exists d' \in [d]_Y : \quad d' \text{ derivabile da } S'$$

Proof: Sia $D = (D_1, \ldots, D_n) \in \mathcal{D}(S)$ la derivazione da cui deriviamo d, cioe' $d = D_n$. Procediamo per induzione su n.
Case: $n = 1$
D e' una derivazione. Siamo quindi forzati ad avere $d = D_1 \in S$.
Allora, poiche' $S' \in [S]_Y$, la disgiunzione $d = D_1$ comparira' in S' come un ingrandimento D'_1 che potra' essere effettivo oppure no.
In ogni caso ponendo $d' = D'_1$, abbiamo la tesi.
Case: $m < n \Rightarrow n$
Se $d = D_n \in S$, basta ragionare come prima. Altrimenti, se $d \notin S$, vuol dire che abbiamo applicato una regola di risoluzione ad al piu' due disgiunzioni D_{i_1}, D_{i_2} per ottenere d (vedi [4.44,pg.25]). Dobbiamo esaminare tutti i possibili casi, ma noi ne analizzeremo solo alcuni.
Case: $d = [\ldots, Z, \ldots,]$ ottenuto da $D_i = [\ldots, \neg\neg Z, \ldots,]$, ovvero $d = D_i(\neg\neg Z|Z)$

$$i < n \underbrace{\Rightarrow}_{\text{Hp induttiva}} \exists D'_i \in [D_i]_Y : \quad D'_i \text{ derivabile } da S'$$

Sia $(\overline{D}_1, \ldots, \overline{D}_m) \in \mathcal{D}(S')$, $\overline{D}_m = D'_i$, la derivazione da cui ricaviamo D'_i.
Poniamo $\overline{D}_{m+1} = \overline{D}_m(\neg\neg Z|Z) = D'_i(\neg\neg Z|Z)$, ovvero sostituiamo $\neg\neg Z$ con Z. Allora, $(\overline{D}_1, \ldots, \overline{D}_{m+1})$ e' la derivazione cercata per d. Infatti, la sostituzione di $\neg\neg Z$ e' proprio una regola di risoluzione e inoltre $\overline{D}_{m+1} \in [d]_Y$, infatti,
Case: $D'_i = D_i$ (ingrandimento non effettivo)

$$\overline{D}_{m+1} = D'_i(\neg\neg Z|Z) = D_i(\neg\neg Z|Z) = d \in [d]_Y$$

Case: $D'_i = [X_1, \ldots, \neg\neg Z, \ldots, X_n, Y]$ con $D_i = [X_1, \ldots, \neg\neg Z, \ldots, X_n]$ (ingrandimenti effettivo)

$$\overline{D}_{m+1} = D'_i(\neg\neg Z|Z) = [X_1, \ldots, Z, \ldots, X_n, Y] \in [d]_Y$$

Analogamente si dimostrano gli altri casi.

□

Theorem 4.58. *di completezza per le Risoluzioni*

$$\models X \Rightarrow \vdash X$$

Proof: Procediamo per assurdo: $\nexists R \in \mathscr{R}(\neg X):$ R e' chiusa Sia

$$C = \{W \subseteq \mathrm{Form}\mathrm{I}_{\alpha\beta} \mid |W| < \infty, \ \exists R \in \mathscr{R}(W): \text{ chiusa } \}$$

Quindi $\neg X \in C$.

$\langle 1 \rangle 1$. C e' una proprieta' di consistenza

Sia $S = \{s_1, \ldots, s_n\} \in C$.

1. Non e' vero che $P, \neg P \in S$, altrimenti se consideriamo la risoluzione formata da:

$$\begin{array}{l} [s_1] \\ \vdots \\ [P] \\ [\neg P] \\ \vdots \\ [s_n] \end{array}$$

   e applichiamo la regola risolutiva 6. a $[P], [\neg P]$, possiamo aggiungere [] alla lista e chiudere cosi' la risoluzione.

   Questo e' assurdo contro $S \in C$.
2. Analogamente, $\mathbf{f}$, $\neg T$ non possono stare in S
3. Supponiamo che $\beta \in S$ e supponiamo per assurdo che $S_1 = S \cup \{\beta_1\}$, $S_2 = S \cup \{\beta_2\} \notin C$, ovvero che esistono due risoluzioni $R_1 = (D_1, \ldots, D_n)$, $R_2 = (E_1, \ldots, E_m)$ per S_1, S_2 chiuse, cioe' $D_n = E_m = []$

   $$\beta \in S \Rightarrow S = \{X_1, \ldots, X_k, \beta\}$$

   Sia R una risoluzione per S in cui applichiamo una volta la regola di espansione a β, ovvero R e'

$$\begin{array}{l} [X_1] \\ [X_2] \\ \vdots \\ [X_k] \\ [\beta] \\ [\beta_1, \beta_2] \end{array}$$

Sia $\overline{R}_1 = (D_1, \ldots, D_{k+2})$ il primo pezzo di R_1:

$$\begin{array}{l} [X_1] \\ [X_2] \\ \vdots \\ [X_k] \\ [\beta] \\ [\beta_1] \end{array}$$

R e' un β_2-ingradimento effettivo di $\overline{R}_1$.
Poiche' R_1 e' chiusa, [] e' derivabile da $\overline{R}_1$, cioe' R e' del tipo

$$\begin{array}{l} [X_1] \\ [X_2] \\ \vdots \\ [X_k] \\ [\beta] \\ [\beta_1] \\ \vdots \\ [] \end{array}$$

Allora

$$\begin{cases} R \in \left[\overline{R}_1\right]_{\beta_2} \\ [] \text{ derivabile da } \overline{R}_1 \end{cases} \underbrace{\Rightarrow}_{[4.57,\text{pg.}33]} \exists \text{ un } \beta_2\text{-ingrandimento di [] che e' derivabile da R}$$

L'ingrandimento di [] puo' essere [] stesso oppure $[\beta_2]$.
Case: []
Sia $R' = (Y_1, \ldots, Y_h)$ la derivazione di [] in $\mathcal{D}(R)$. Aggiungendo a R gli elementi di R' che non sono gia' presenti in R, otteniamo la risoluzione R^* del tipo:

$$\begin{array}{l} [X_1] \\ [X_2] \\ \vdots \\ [X_k] \\ [\beta] \\ [\beta_1, \beta_2] \\ Y_{i_1} \\ \vdots \\ Y_{i_j} \\ Y_h = [] \end{array}$$

(nota: $Y_h = []$ lo aggiungiamo perche' non era contenuto in R, in quanto $S \in C$).
La risoluzione R^* e' quindi una risoluzione chiusa per S. Assurdo contro $S \in C$.
Case: $[\beta_2]$

Sia $R' = (Y_1, \ldots, Y_h)$ la derivazione di $[\beta_2]$ in $\mathcal{D}(R)$. R' e' del tipo:

$$
\begin{array}{l}
[X_1] \\
[X_2] \\
\vdots \\
[X_k] \\
[\beta] \\
[\beta_1, \beta_2] \\
\vdots \\
[\beta_2]
\end{array}
$$

La risoluzione chiusa $R_2 = (E_1, \ldots, E_m)$ di $S_2 = S \cup \{\beta_2\}$ e' del tipo

$$
\begin{array}{l}
[X_1] \\
[X_2] \\
\vdots \\
[X_k] \\
[\beta] \\
[\beta_2] \\
E_{k+3} \\
\vdots \\
E_m = []
\end{array}
$$

Quindi possiamo pensare R_2 come la derivazione di [] dalla lista $\overline{R}_2 = (E_1, \ldots, E_{k+2})$ che e' del tipo:

$$
\begin{array}{l}
[X_1] \\
[X_2] \\
\vdots \\
[X_k] \\
[\beta] \\
[\beta_2]
\end{array}
$$

Quest'ultime disgiunzioni sono tutte contenute in R'. Quindi applicando, in R', le stesse regole di risoluzione che abbiamo usato per derivare [] da $\overline{R}_2$ riusciamo ad ottenere la

risoluzione R'^*:

$$
\begin{array}{l}
[X_1] \\
[X_2] \\
\vdots \\
[X_k] \\
[\beta] \\
[\beta_1, \beta_2] \\
\vdots \\
[\beta_2] \\
E_{k+3} \\
\vdots \\
E_m = []
\end{array}
$$

e quindi abbiamo ottenuto una risoluzione chiusa per S. Assurdo.

4. Si procede analogamente per gli altri casi.

□

Theorem 4.59. *di esistenza del modello.*
Sia C una proprieta' di consistenza, allora

$$S \in C \Rightarrow S \text{ e' soddisfacibile}$$

Questo teorema e' una generalizzazione del thm [4.53,pg.30].

Proof: Analizziamo solo il caso S infinito, poiche' l'altro e' gia' stato dimostrato in [4.53,pg.30].

Siano dati S_1, S_2 t.c.

$$S_2 \in C,\ S_1 \subseteq S_2,\ \ S_1 \text{ finito}$$

Utilizziamo una versione modificata dell'algoritmo visto nella dimostrazione di [4.53,pg.30]. Il

nostro scopo e' quello di applicare l'algoritmo ad ogni elemento di S in un ordine ben preciso.

SATURA(S_1, S_2) :
- $S_1' := S_1$
- $S_2' := S_2$
- **While** $S_1 \neq \emptyset$
  - **Sia** $X \in S_1$
  - $S_1 := S_1 \setminus \{X\}$
  - **Se** $X \in \text{Var} \cup \neg\text{Var} \cup \{\mathbf{t}, \neg\mathbf{f}\}$:
    - **continua**
  - **Se** $X := \neg\neg Z$:
    - $S_1' := S_1' \cup \{Z\}$
    - $S_2' := S_2' \cup \{Z\}$
    - $S_1 := S_1 \cup \{Z\}$
  - **Se** $X := \alpha$:
    - $S_1' := S_1' \cup \{\alpha_1, \alpha_2\}$
    - $S_2' := S_2' \cup \{\alpha_1, \alpha_2\}$
    - $S_1 := S_1 \cup \{\alpha_1, \alpha_2\}$
  - **Se** $X := \beta$:
    - **Sia** $b \in \{\beta_1, \beta_2\}$: $S_2' \cup \{b\} \in C$
    - $S_1' := S_1' \cup \{b\}$
    - $S_2' := S_2' \cup \{b\}$
    - $S_1 := S_1 \cup \{b\}$
- $R_1 := S_1'$
- $R_2 := S_2'$
- **Ritorna** R_1, R_2

In pratica, stavolta, S_1 e' l'insieme che fa da contatore e che man mano si svuota. S_1' e' l'insieme che colleziona tutte le sottoformule che processiamo. S_2' e' l'equivalente di S_1', ma piu' "forte", nel senso che appartiene sempre a C.
I loop-invarianti di questo algoritmo sono infatti:

1. $S_2' \in C$
2. $S_1 \subseteq S_1' \subseteq S_2'$

che si dimostrano per induzione come abbiamo fatto in [4.53,pg.30]. In particolare abbiamo

$$S_1 \subseteq R_1 \subseteq R_2$$
$$R_2 \in C \quad (3.1)$$
$$S_2 \subseteq R_2 \quad (3)$$

Poiche' S_1 e' finito, l'algoritmo terminera' (vedi sempre [4.53,pg.30]).
Inoltre, si vede anche che R_1 e' un insieme di Hintikka, infatti, (analizziamo solo alcuni casi):

$$P, \neg P \in R_1 \subseteq R_2 \text{ assurdo contro } R_2 \in C$$

Se $\neg\neg Z \in R_1$, considerando l'S_1 di partenza, abbiamo due casi:
CASE: $\neg\neg Z \in S_1$

L'algoritmo termina, quindi processa tutte le formule contenute in S_1, in particolare a un certo punto arrivera' nel caso in cui $X = \neg\neg Z$. Quindi verra' applicata la regola: $S_1' := S_1' \cup \{Z\}$.

$$Z \in S_1' \subseteq R_1$$

Quindi $Z \in R_1$

CASE: $\neg\neg Z \notin S_1$

In questo caso, $\neg\neg Z$ e' stato aggiunto in S_1' durante la valutazione di una qualche formula $X \in S_1$.

In ogni caso, cosi' come e' stato aggiunto in S_1' viene aggiunto in S_1, e quindi ritorniamo al caso precedente, con la sola differenza che stavolta S_1 non e' quello di partenza.

Ricordiamo che la logica proposizionale in cui stiamo operando e' $\text{FormI}_{\alpha\beta}$ (vedi [4.30,pg.17]).

$S \subseteq \text{FormI}_{\alpha\beta} \subseteq \mathcal{U} = (\text{AT} \cup \neg\text{AT} \cup \text{Bin}_\text{I} \cup \{(,),\neg\})^\infty$

$$\begin{cases} \text{AT} = \{\mathbf{t}, \mathbf{f}\} \cup \{p_1, p_2, \dots,\} \text{ e' numerabile} \\ \text{Bin}_\text{I} \text{ e' finito} \end{cases} \Rightarrow \text{AT} \cup \neg\text{AT} \cup \text{Bin}_\text{I} \cup \{(,),\neg\} \text{ e' numerabile} \quad (1)$$

Sia $A = \text{AT} \cup \neg\text{AT} \cup \text{Bin}_\text{I} \cup \{(,),\neg\}$

$$\begin{cases} \mathcal{U} = \bigcup_{i \in \mathbb{N}^*} A^i \\ A \text{ numerabile} \end{cases} \underbrace{\Rightarrow}_{\text{Unione numerabile di numerabili e' numerabile}} \mathcal{U} \text{ e' numerabile } \Rightarrow$$

$\Rightarrow \text{FormI}_{\alpha\beta}$ e' numerabile $\Rightarrow S$ e' numerabile

Poiche' S e' numerabile, possiamo scrivere

$$S = \{X_0, X_1, X_2, \dots,\}$$

Definiamo la seguente sequenza:

$$T_0 := S$$

$$T_i := R_2, \text{ dove } R_2 \text{ e' ottenuto da } \text{SATURA}(\{X_i\}, T_{i-1})$$

Sia R_2 ottenuto da $\text{SATURA}(\{X_{i+1}\}, T_i)$, ovvero $R_2 = T_{i+1}$.

Per la (3) abbiamo

$$\{X_{i+1}\}, T_i \subseteq R_2 = T_{i+1}$$

$$T_i \subseteq T_{i+1} \ \forall i \quad (4)$$

⟨1⟩1. Dim. per induzione che $\{X_i\} \subseteq S \subseteq T_i$

CASE: $i = 0$

$$\{X_0\} \subseteq \{X_0, X_1, \dots,\} = S = T_0$$

CASE: $i \Rightarrow i + 1$

Si ha

$$\{X_{i+1}\} \subseteq \{X_0, X_1, \dots,\} = S \underbrace{\subseteq}_{\text{Hp induttiva}} T_i \underbrace{\subseteq}_{(4)} T_{i+1}$$

Per la (3.1) abbiamo anche che

$$T_i \in C \ \forall i \in \mathbb{N}$$

Sia $\mathcal{T} = \bigcup_{i \in \mathbb{N}} T_i$.

⟨1⟩2. Dim. $\mathcal{T}$ e' un insieme di Hintikka

⟨2⟩1. $\mathbf{f} \notin \mathcal{T}$

Se per assurdo $\mathbf{f} \in \mathcal{T}$, si avrebbe:

$$\exists i : \mathbf{f} \in T_i$$

e questo e' assurdo contro $T_i \in C$.

⟨2⟩2. $\neg\mathcal{T} \notin \mathcal{T}$

Analogamente a prima.

⟨2⟩3. $P \in \text{Var} \Rightarrow (P \notin \mathcal{T} \text{ oppure } \neg P \notin \mathcal{T})$

Supponiamo $P \in \text{Var}$.
Se per assurdo $P, \neg P \in \mathcal{T}$, si avrebbe:

$$\exists i : \ P \in T_i, \ \ \exists j : \ \neg P \in T_j$$

supponiamo $j > i$, quindi

$$T_i \subseteq T_j \Rightarrow P, \ \neg P \in T_j$$

e questo e' assurdo contro $T_j \in C$.

$\langle 2 \rangle 4$. $\neg\neg Z \in \mathcal{T} \Rightarrow Z \in \mathcal{T}$

$$\neg\neg Z \in \mathcal{T} \Rightarrow \exists i : \ \neg\neg Z \in T_i$$

Dobbiamo distinguere due casi:
Case: $\neg\neg Z \in T_0 = S$

$$\neg\neg Z \in \{X_1, X_2, \dots, \} \Rightarrow \exists i : \ \neg\neg Z = X_i$$
$$X_i \underbrace{\in}_{\text{passo } \langle 1 \rangle 1} T_i$$
$$T_i = \text{SATURA}(\{X_i\}, T_{i-1})$$

L'algoritmo SATURA, avra' esaminato il caso $X = X_i$, e quindi avra' usato la regola

$$S'_2 := S'_2 \cup \{Z\}$$

Poiche' $S'_2 \subseteq R_2 = T_i$, abbiamo

$$Z \in T_i \subseteq \mathcal{T}$$

Case: $\neg\neg Z \notin T_0$
Sia $i_0 = \min\{i \mid \neg\neg Z \in T_i\}$, quindi

$$\neg\neg Z \notin T_{i_0 - 1} \ \ (5)$$
$$\neg\neg Z \in T_{i_0} = \text{SATURA}(X_{i_0}, T_{i_0 - 1})$$
$$(5), \ \ X_{i_0} \in S = T_0 \Rightarrow \neg\neg Z \neq X_{i_0} \ \ (5.1)$$

Dalla (5.1) vediamo che l'unico modo per cui $\neg\neg Z$ sia finito in T_{i_0} e' che sia stato aggiunto a S'_2 durante qualche **Se** . Ma allora, in quel momento, sara' stato anche aggiunto anche a S_1, cioe' l'algoritmo avra' eseguito la linea:

$$S_1 := S_1 \cup \{\neg\neg Z\}$$

Poiche' l'algoritmo termina e svuota S_1, prima o poi l'algoritmo sara' entrato nel caso $X = \neg\neg Z$. Quindi, sara' stata applicata la regola:

$$S'_2 := S'_2 \cup \{Z\}$$

In conclusione:

$$Z \in S'_2 \subseteq R_2 = T_{i_0} \subseteq \mathcal{T}$$

Sostanzialmente: se $\neg\neg Z$ e' una formula di S, prima o poi sara' processata e verra' aggiunto Z. Se $\neg\neg Z$ non e' in S, allora vuol dire che e' stata aggiunta in un certo punto come sottoformula, ma allora verra' prima o poi processata.

$\langle 2 \rangle 5$. $\alpha \in \mathcal{T} \Rightarrow \alpha_1, \alpha_2 \in \mathcal{T}$
Si procede analogamente a prima.

$\langle 2 \rangle 6$. $\beta \in \mathcal{T} \Rightarrow \beta_1 \in \mathcal{T} \vee \beta_2 \in \mathcal{T}$
Si procede analogamente a prima.

$\langle 2 \rangle 7$. Q.E.D.

$$\mathcal{T} \text{ e' di Hintikka } \underbrace{\Rightarrow}_{[4.50,\text{pg.}28]} \mathcal{T} \text{ soddisfacibile } \underbrace{\Rightarrow}_{S = T_0 \subseteq \mathcal{T}} S \text{ e' soddisfacibile}$$

□

4.5 Compattezza

Definition 4.60. Sia $S \subseteq \mathrm{FormI}_{\alpha\beta}$.

$$S \text{ e' finitamente soddisfacibile } \overset{\text{def}}{\Leftrightarrow} \forall D \subseteq S : \ |D| < \infty \ \ D \text{ e' soddisfacibile}$$

Theorem 4.61. *di compattezza.*

$$S \textit{ finitamente soddisfacibile } \Leftrightarrow S \textit{ soddisfacibile}$$

Proof:
⟨1⟩1. Dim. $\Leftarrow$

$$S \text{ soddisfacibile } \Rightarrow \exists v \in \mathcal{V} : \ \ v(s) = \mathbf{t} \ \forall s \in S \Rightarrow v(s) = \mathbf{t} \ \ \forall s \in D$$

⟨1⟩2. Dim. $\Rightarrow$
Sia

$$C = \{W \subseteq \mathrm{FormI}_{\alpha\beta} \mid W \text{ finitamente soddisfacibile}\}$$

⟨2⟩1. C e' una proprieta' di consistenza
Sia $W \in C$
⟨3⟩1. $\mathbf{f} \notin W$
Altrimenti, $\{\mathbf{f}\}$ sarebbe soddisfacibile, il che' e' assurdo.
⟨3⟩2. $\neg\mathbf{t} \notin W$
Analogamente a prima
⟨3⟩3. $P \in \mathrm{Var} \Rightarrow \neg P \notin W \vee P \notin W$
Se per assurdo $\{P, \neg P\} \subseteq W$, si avrebbe che $\{P, \neg P\}$ e' soddisfacibile. Assurdo.
⟨3⟩4. $\neg\neg Z \in W \Rightarrow W \cup \{Z\} \in C$
Sia $D \subseteq W \cup \{Z\}$, finito. Dimostriamo che e' soddisfacibile.
Case: $Z \notin D$
In questo caso,

$$D \subseteq W \underbrace{\Rightarrow}_{W \in C} D \text{ e' soddisfacibile}$$

Case: $Z \in D$

$Z \notin D \Rightarrow D \setminus \{Z\} \subseteq W$
$(D \setminus \{Z\}) \cup \{\neg\neg Z\} \in W \Rightarrow (D \setminus \{Z\}) \cup \{\neg\neg Z\}$ soddisf. $\Leftrightarrow$
$\Leftrightarrow \exists v \in \mathcal{V} : \ v(D \setminus \{Z\}) = \mathbf{t}, \ v(\neg\neg Z) = \mathbf{t} \Rightarrow \forall d \in D \setminus \{Z\} : \ v(d) = \mathbf{t}, \ \ v(Z) = \mathbf{t} \Rightarrow$
$\Rightarrow \forall d \in D \ \ v(d) = \mathbf{t} \Leftrightarrow v(D) = \mathbf{t} \Leftrightarrow D$ soddisf.

⟨3⟩5. $\alpha \in W \Rightarrow W \cup \{\alpha_1, \alpha_2\} \in C$
Analogamente a prima.
⟨3⟩6. $\beta \in W \Rightarrow W \cup \{\beta_1\} \in C \vee W \cup \{\beta_2\} \in C$
Supponiamo per assurdo che $\exists D_1 \subseteq W \cup \{\beta_1\}, D_2 \subseteq W \cup \{\beta_2\}$ finiti e non soddisfacibili.
Se per assurdo $\beta_i \notin D_i$, allora

$$D_i \subseteq W \Rightarrow D_i \text{ soddisf.}$$

assurdo.
Quindi $\beta_i \in D_i$.

$(D_1 \setminus \{\beta_1\}) \cup (D_2 \setminus \{\beta_2\}) \cup \{\beta\} \subseteq W \Rightarrow$ e' soddisfacibile $\Rightarrow \exists v \in \mathcal{V} : \ v(\beta) = v(D_1 \setminus \{\beta_1\}) = v(D_2 \setminus \{\beta_2\}) = \mathbf{t}$
$v(\beta) = \mathbf{t} \Rightarrow v(\beta_1) = \mathbf{t} \vee v(\beta_2) = \mathbf{t}$
Case: $v(\beta_1) = \mathbf{t}$

In questo caso, tutto D_1 diventa soddisfacibile. Assurdo.
CASE: $v(\beta_1) = \mathbf{f},\ v(\beta_2) = \mathbf{t}$

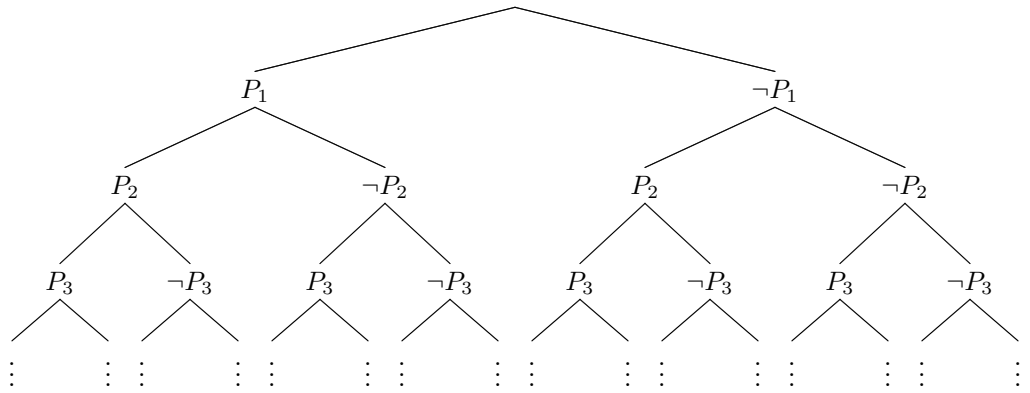
$$\begin{aligned} & v(D_2 \setminus \{\beta_2\}) = \mathbf{t} \\ & v(\beta_2) = \mathbf{t} \\ & \qquad \Rightarrow v(D_2) = \mathbf{t} \end{aligned}$$

D_2 soddisfacibile. Assurdo.

□

4.6 Risoluzioni ristrette

Definition 4.62. Siano $P_1, \ldots, P_n \in \text{Var}$ delle lettere proposizionali. Possiamo costruire il loro *albero semantico*:



Possiamo anche costruire l'albero semantico di tutte le lettere Var. In questo caso avremo un albero infinito, a ramificazione finita.

In maniera naturale, l'albero semantico permette di codificare tutte le possibili valutazioni booleane. Un ramo r dell'albero e' quindi interpretabile come una particolare valutazione booleana secondo la legge:

$$v(P) = \begin{cases} \mathbf{t} & P \in \text{nodo}\,(r) \\ \mathbf{f} & \neg P \in \text{nodo}\,(r) \end{cases}$$

Definition 4.63. Sia $R \in \mathscr{R}(X)$ una risoluzione Sia $C \subseteq [\text{Lit}]$ un insieme di disgiunzioni letterali (vedi [4.34,pg.18]).

C e' saturo per risoluzione $\overset{\text{def}}{\Leftrightarrow} \forall c_1, c_2 \in C \ \ c \in C$

, dove c e' la disgiunzione letterale ottenuta da c_1, c_2 con le regole risolutive (vedi [4.44,pg.25])

Sia $c \in C$ e siano $\{L_1, \ldots, L_k\} = c \cap \text{Lit} \setminus TF$ i letterali che compaiono in c, ma che non sono $\mathbf{t}, \mathbf{f}$. Sia A l'albero semantico formato sulle lettere $P_1, \ldots, P_k$ che compaiono in tutte gli elementi di C, ovvero sulle lettere dell'insieme

$$\bigcup_{c \in C} c \cap \text{Var}$$

Sia r un ramo dell'albero A e sia $N \in \text{nodo}(r)$ un suo nodo.

r contraddice $c \overset{\text{def}}{\Leftrightarrow} \overline{L_k} \in \text{nodo}(r) \quad \forall i = 1, \ldots, k$

dove $\overline{L_k}$ e' la negazione di L_k (vedi [4.17,pg.12]), con la seguente modifica: $\overline{\neg X} = X$

r e' C-chiuso $\overset{\text{def}}{\Leftrightarrow} \exists c \in C : \quad r$ contraddice c

N e' fallimentare per $C \overset{\text{def}}{\Leftrightarrow}$ il ramo $(S, \ldots, N)$ e' C-chiuso

dove S e' la radice di A

Se A e' infinito poniamo

$$r \in \text{ramo}(A) \text{ e' un ramo massimale } \overset{\text{def}}{\Leftrightarrow} r \text{ e' infinito}$$

Se A e' finito poniamo

$$r \in \text{ramo}(A) \text{ e' un ramo massimale } \overset{\text{def}}{\Leftrightarrow} r \text{ e' un ramo terminato (vedi [2,pg.1])}.$$

Sia $\text{ramo}_{\max}(A)$ l'insieme di tutti i rami massimali di A.

Proposition 4.64. *Sia* $C \subseteq [\text{Lit}]$,

$$\begin{cases} C \textit{ e' saturo per risoluzione} \\ C \textit{ insoddisfacibile} \end{cases} \Rightarrow [] \in C$$

Proof: Sia C' l'insieme di disgiunzioni letterali ottenute da quelle di C rimuovendo ogni occorrenza del simbolo $\mathbf{f}$. Ovvero, preso $[\ldots, \mathbf{f}, \ldots,] \in C$ e applicando la regola risolutiva 7. (vedi [4.44,pg.25]) abbiamo $[\ldots,] \in C'$.

⟨1⟩1. C' e' saturo

Sia $c \in C$ e sia c' ottenuto da c applicando la regola risolutiva 7. (vedi [4.44,pg.25]).

$$C \text{ saturo} \underbrace{\Rightarrow}_{[4.63,\text{pg.42}]} c' \in C$$

Poiche' gli elementi di C' sono appunto quelli ottenuti da C tramite la regola 7., si deve avere $C' \subseteq C$.

Dimostriamo che C' e' saturo.

Sia $c' \in C'$ e sia d' la disgiunzione ottenuta da c' tramite la regola 7.. Poiche' in C' non esistono disgiunzioni con il simbolo $\mathbf{f}$, la regola 7. lascia invariato c', cioe' $d' = c'$ e quindi $d' = c' \in C'$

Sia adesso $c'_1, c'_2 \in C'$ e sia d' la disgiunzione ottenuta da c'_1, c'_2 tramite la regola 6.. Abbiamo:

$$c'_1, c'_2 \in C' \subseteq C \underbrace{\Rightarrow}_{[4.63,\text{pg.42}]} d' \in C$$

$f \notin c'_1 \cup c'_2 \Rightarrow \mathbf{f} \notin d'$ ovvero, il simbolo $\mathbf{f}$ non puo' stare in $d' \Rightarrow d' = d'' \in C'$

dove d'' e' ottenuto da d' con la 7.

⟨1⟩2. C' e' insoddisfacibile

⟨2⟩1. Metodo diretto

$$C \text{ insoddisfacibile } \Rightarrow \exists c \in C : \quad v(c) = \mathbf{f} \quad \forall v \in \mathcal{V}$$

Poiche'
$$v([X_1, \ldots, X_n]) = \mathbf{t} \Rightarrow v([\mathbf{f}, X_1, \ldots, X_n]) = \mathbf{t} \quad (1)$$
si ha che
$$v(c') = \mathbf{f} \quad \forall v \in \mathcal{V}$$
dove $c' \in C'$ e' ottenuto da c con la 7.

$\langle 2 \rangle 2$. Metodo indiretto

Se per assurdo C' fosse soddisfacibile, si avrebbe
$$C' \text{ soddisf.} \Rightarrow \exists v \in \mathcal{V} : \quad v(C') = \mathbf{t} \Leftrightarrow v(c') = \mathbf{t} \quad \forall c' \in C'$$
Poiche'
$$v([X_1, \ldots, X_n]) = \mathbf{t} \Rightarrow v([\mathbf{f}, X_1, \ldots, X_n]) = \mathbf{t} \quad (1)$$
e poiche' C si puo' ottenere aggiungendo $\mathbf{f}$ ad alcune disgiunzioni di C', abbiamo che anche C e' soddisfacibile, ovvero:
$$\forall c \in C \; \exists c' = [X_1, \ldots, X_n] \in C' : \quad c = [\mathbf{f}, X_1, \ldots, X_n]$$
$$v(c') = \mathbf{t} \underbrace{\Rightarrow}_{(1)} v(c) = \mathbf{t}$$

$\langle 1 \rangle 3$. $[] \in C' \Leftrightarrow [] \in C$

$\langle 2 \rangle 1$. Dim. $\Rightarrow$

Basta osservare che $C' \subseteq C$

$\langle 2 \rangle 2$. Dim. $\Leftarrow$

Poiche' $[]$ si puo' ottenetere con la 7. da $[] \in C$ stesso, per defizione di C' si ha $[] \in C'$

Grazie ai passi precedenti, basta adesso dimostrare che $[] \in C'$.

Siano $P_1, \ldots, P_n$ tutte le lettere delle disgiunzioni letterali di C'. Sia A il relativo albero semantico e sia S la radice dell'albero (il nodo vuoto).

$\langle 1 \rangle 4$. S e' fallimentare per $C' \Rightarrow [] \in C'$

Osserviamo che il ramo (S) non contiene lettere.

S e' fallimentare per $C' \Rightarrow$ il ramo vuoto (S) e' C'-chiuso $\Rightarrow \exists c' \in C' :$ (S) contraddice c' (2)

Siano $Q_1, \ldots, Q_m$ le lettere di c'
$$(2) \Rightarrow \neg Q_1, \ldots, \neg Q_m \in (S) \underbrace{\Rightarrow}_{S \text{ non e' una lettera}} m = 0$$
ovvero c' non ha lettere, e quindi
$$c' = [] \in C'$$

$\langle 1 \rangle 5$. Dim. che S e' fallimentare per C'

Supponiamo per assurdo che la radice S non sia fallimentare.
$$C' \text{ insoddisfacibile } \Rightarrow \forall r \in \text{ramo}_{\max}(A) \;\; r \text{ e' C'-chiuso} \quad (3)$$
Infatti, se per assurdo $r \in \text{ramo}_{\max}(A)$ non fosse C'-chiuso, avremmo trovato una valutazione booleana (quella associata a r) che soddisfa tutti i $c' \in C'$, e quindi C' sarebbe soddisfacibile. Assurdo.

Preso $r = (r_1, r_2, \ldots,) \in \text{ramo}(A)$, sia r_i il primo nodo di r che rende il ramo $r^* = (r_1, \ldots, r_i)$ un ramo C-chiuso, ovvero $(r_1, \ldots, r_{i-1})$ non e' C-chiuso, mentre r^* lo e'. (Nota[5]).

Sia A^* l'albero t.c.
$$\text{ramo}(A^*) \stackrel{\text{def}}{=} \{r^* \mid r \in \text{ramo}(A)\}$$

Per ipotesi S non e' fallimentare, quindi A^* non e' formato dal solo ramo (S).

$\langle 2 \rangle 1$. A^* e' finito.

[5] Se $r = (r_1, r_2, \ldots,)$ e' un ramo C-chiuso, possiamo affermare che esiste il primo nodo di r che rende il ramo $r^* = (r_1, \ldots, r_i)$ un ramo C-chiuso. Infatti, le disgiunzioni letterali hanno solo un numero finito di lettere.

Se cosi' non fosse, per il lemma di Konig (vedi [2.2,pg.1]) esisterebbe un ramo $r^* \in \text{ramo}(A^*)$ infinito. Ma tutti i rami di A^*, per costruzione sono finiti. Assurdo.

$$A^* \text{ finito } \Rightarrow \text{ramo}_{\max}(A^*) = \text{ramo}_t(A^*)$$

ovvero i rami massimali sono quelli terminati.
Poiche' A^* e' finito, possiamo considerare l'insieme di tutti i suoi rami, non chiusi, piu' lunghi. Sia $r^* = (r_1, \ldots, r_{i-1})$ un tale ramo.

per costruzione A^* ha tutti i rami terminati chiusi, ovvero

$$r \in \text{ramo}_t(A^*) \Leftrightarrow r \text{ e' chiuso} \quad (4)$$

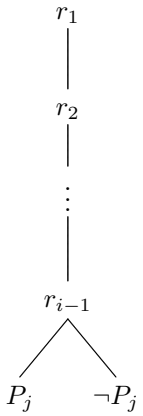
$$r^* \text{ non chiuso } \underbrace{\Rightarrow}_{(4)} r^* \notin \text{ramo}_t(A^*) \Rightarrow r_{i-1} \text{ ha dei figli}$$

Siano L, R i figli di r_{i-1}. Poiche' r^* e' uno dei rami non chiusi piu' lunghi, i rami $r^*_L = (r_1, \ldots, r_{i-1}, L)$ e $r^*_R = (r_1, \ldots, r_{i-1}, R)$ devono essere necessariamente chiusi. Quindi L, R sono fallimentari.

$$S \text{ non fallimentare } \Rightarrow (S) \text{ non chiuso } \Rightarrow r^*_L,\ r^*_R \neq (S) \Rightarrow L, R \in \text{Var} \Rightarrow$$

$$\underbrace{\Rightarrow}_{\text{def. di albero semantico}} \quad L = P_j,\ R = \neg P_j \text{ oppure } L = \neg P_j,\ R = P_j$$

Supponiamo $L = P_j,\ R = \neg P_j$. Graficamente la situazione e' la seguente:



Siano $c_L,\ c_R \in C'$ due elementi di C' contraddetti da $r^*_L,\ r^*_R$.

$\langle 2 \rangle 2.\ \neg P_j \in c_L,\ \ P_j \notin c_L$

Se per assurdo $\neg P_j \notin c_L$,

$$r^*_L \text{ contraddice } c_L \Leftrightarrow \forall H \in c_L \cap \text{Lit} \setminus TF\ \ \overline{H} \in r^*_L$$

$$\exists H \in c_L \cap \text{Lit} \setminus TF :\ \ \overline{H} = P_j \ \Leftrightarrow\ \exists H \in c_L \cap \text{Lit} \setminus TF\ \ H = \neg P_j \Leftrightarrow \neg P_j \in c_L \quad (5)$$

$$\neg P_j \notin c_L,\ \ (5) \Rightarrow \forall H \in c_L \cap \text{Lit} \setminus TF\ \ \overline{H} \in r^* \Rightarrow r^* \text{ contraddice } c_L \Rightarrow r^* \text{ chiuso}$$

Assurdo. Quindi $\neg P_j \in c_L$.

Se per assurdo $P_j \in c_L$,

$$P_j \in C_L,\ \ r^*_L \text{ contraddice } C_L \Rightarrow \neg P_j \in r^*_L$$

assurdo contro $P_j \in r^*_L$: in un ramo di un albero semantico non ci puo' essere sia P che $\neg P$.

$\langle 2 \rangle 3.\ P_j \in c_R,\ \ \neg P_j \notin c_R$

Analogamente a prima.

Consideriamo la disgiunzione $c = (c_L \setminus \{\neg P_j\}) \cup (c_R \setminus \{P_j\})$. Per quanto detto si ha:

$$P_j,\ \neg P_j \notin c \Rightarrow \forall H \in c \cap \mathrm{Lit} \setminus TF\ \ \overline{H} \in r^* \Rightarrow r^* \text{ contraddice } c$$

Ma questo e' un assurdo contro r^* non C'-chiuso.

□

Definition 4.65. Sia R un insieme di disgiunzioni. R e' di *Robinson* $\overset{\text{def}}{\Leftrightarrow}$

1. $D \in R \setminus [\mathrm{Lit}] \Rightarrow D' \in R$
   dove D' e' una qualsiasi disgiunzione ottenuta da D tramite una regola di espansione
2. $R \cap [\mathrm{Lit}]$ e' saturo per risoluzione
3. $[] \notin R$

La 1. equivale a dire che R e' saturo per espansione.

Theorem 4.66.

$$R \textit{ di Robinson } \Rightarrow R \textit{ soddisfacibile}$$

Proof: Sia $C = R \cap [\mathrm{Lit}]$.

⟨1⟩1. C e' saturo per risoluzione

Infatti, Presi $c_1, c_2 \in C$ $c \in C$, dove c e' la disgiunzione letterale ottenuta da c_1, c_2 con le regole risolutive (vedi [4.44,pg.25]), si ha

$$c_1, c_2 \in [\mathrm{Lit}] \Rightarrow c \in [\mathrm{Lit}] \cup \left\{\underline{[]}\right\}$$

$$C = R \cap [\mathrm{Lit}]\,,\ \ R \text{ saturo per risoluzione } \Rightarrow c \in R \underbrace{\Rightarrow}_{3.} c \neq []$$

$$[] \notin C,\ \ C \text{ saturo } \underbrace{\Rightarrow}_{[4.64,\mathrm{pg}.43]} \begin{array}{c} c \in R \cap [\mathrm{Lit}] = C \\ C \text{ soddisfacibile} \end{array} \Rightarrow \exists v \in \mathcal{V} :\ \ v(C) = \mathbf{t}$$

⟨1⟩2. v soddisfa R

Procediamo per induzione sul rango delle disgiunzioni. Ovvero,

$$R_i \overset{\text{def}}{=} \{d \in R \mid r(d) = i\}$$

dimostriamo che $v(R_i) = \mathbf{t}\ \ \forall i \in \mathbb{N}$.

Case: R_0

$$d \in R_0 \Rightarrow d \in [\mathrm{Lit}] \Rightarrow d \in C \Rightarrow v(d) = \mathbf{t}$$

Case: $v(R_m) = \mathbf{t},\ m < n \Rightarrow v(R_n) = \mathbf{t}$

Sia $d \in R_n$. Applicando una regola di espansione su d, otteniamo delle disgiunzioni d', d'' con rango inferiore. Per Hp induttiva, v soddisfa d', d''. Allora, d e' pure soddisfatta da v, infatti le regole di espansione garantiscono che

$$d \Leftrightarrow d' \wedge d''$$

□

Definition 4.67. Una risoluzione $R \in \mathscr{R}(X)$ e' ristretta $\Leftrightarrow$ le regole di espansione e la regola risolutiva 7. (vedi [4.44,pg.25]) vengono applicate al piu' una sola volta ad ogni disgiunzione e la regola risolutiva 6. viene applicata al piu' una volta ad ogni coppia di disgiunzioni. Inoltre, le regole di espansione devono essere applicate, il piu' possibile[6] , prima di quelle risolutive.

[6] "il piu' possibile" vuol dire che dobbiamo applicare le regole fino a quando non e' piu' possibile applicarle. A quel punto ci fermiamo.

Theorem 4.68. *di completezza per risoluzioni ristrette.*

$$\models X \Rightarrow \vdash X$$

Proof: Considero la risoluzione iniziale

$$[\neg X]$$

e applico ad essa il piu' possibile le regole di espansione, mantenendo la ristrettezza.
Questo procedimento avra' termine, perche' applicare le regole di espansione a $[\neg X]$ equivale a convertire $\neg X$ in una CNF (vedi [4.37,pg.19]).
Ottengo quindi la risoluzione finale S, che e' satura per espansione. Sia $C = S \cap [\text{Lit}]$. Adesso applico a C le regole risolutive, mantenendo la ristrettezza. Poiche' C e' finito e poiche' stiamo usando la ristrettezza, anche questo procedimento avra' termine (basta applicare il lemma di Konig (vedi [2.2,pg.1]) come gia' fatto nella dimostrazione di [4.37,pg.19] (ma utilizzando la lunghezza della formule piuttosto che il rango).
Ottengo cosi' l'insieme C', che e' saturo per risoluzione.
Sia $R = S \cup C'$. R e' una risoluzione per $\neg X$. Supponiamo per assurdo che $\nvdash X$, allora

$$\nvdash X \Rightarrow [] \notin R$$

Quindi

S saturo per espansione $\Rightarrow R$ saturo per espansione

C' saturo per risoluzione $\Rightarrow C' = R \cap [\text{Lit}]$ saturo per risoluzione

$[] \notin R$

$\Rightarrow R$ e' di Robinson $\underbrace{\Rightarrow}_{[4.66,\text{pg.}46]}$ R e' soddisfacibile $\Rightarrow \neg X$ e' soddisfacibile

Assurdo contro $\models X$. □

4.7 Conseguenza proposizionale

Theorem 4.69.

$$S \models X \Leftrightarrow \exists S_0 \subseteq S \textit{ finito t.c. } S_0 \models X$$

Proof: Ricordiamo che

$$S \models X \Leftrightarrow (\exists v \in \mathcal{V} : \ v(S) = \mathbf{t} \Rightarrow v(X) = \mathbf{t})$$

⟨1⟩1. Dim. $\Leftarrow$

$$v(S) = \mathbf{t} \Rightarrow v(S_0) = \mathbf{t} \Rightarrow v(X) = \mathbf{t}$$

⟨1⟩2. Dim. $\Rightarrow$
Supponiamo per assurdo che

$$\forall S_0 \subseteq S \text{ finito} \quad S_0 \not\models X \quad (1)$$

⟨2⟩1. $S \cup \{\neg X\}$ e' finitamente soddisfacibile.
Sia $D \subseteq S \cup \{\neg X\}$ finito. Sia $S_0 = D \setminus \{\neg X\} \subseteq S$.
D finito, $S_0 \subseteq S \underbrace{\Rightarrow}_{(1)} S_0 \not\models X \Rightarrow \exists v \in \mathcal{V} : \ v(S_0) = \mathbf{t}, \ v(\neg X) = \mathbf{f} \Rightarrow v(X) = \mathbf{t} \Rightarrow v(D) = \mathbf{t}$

$S \cup \{\neg X\}$ e' finitamente soddisfacibile. $\underbrace{\Rightarrow}_{[4.61,\text{pg.}41]}$ $S \cup \{\neg X\}$ soddisf. $\Rightarrow v(S) = \mathbf{t}, \ v(X) = \mathbf{f}$

assurdo.

□

4.8 Sistemi di Hilbert

Definition 4.70. Un sistema di Hilbert e' formato da un insieme $\mathcal{A} \subseteq \mathrm{FormI}_{\alpha\beta}$ di assiomi, e da un insieme di regole di inferenza, ad esempio il modus ponens (m.p.).

Una dimostrazione di $X \in \mathrm{FormI}_{\alpha\beta}$ a partire da $S \subseteq \mathrm{FormI}_{\alpha\beta}$ e' sequenza $X_1, \ldots, X_n$ di formule tali che per ogni i si abbia

$X_i \in \mathcal{A} \cup S$ oppure X_i e' ottenibile da $X_1, \ldots, X_{i-1}$ con l'uso di qualche regola di inferenza

e inoltre, $X_n = X$.

Una sequenza $X_1, \ldots, X_n$ si chiama *derivazione* da $S \subseteq \mathrm{FormI}_{\alpha\beta}$ se e' una dimostrazione di X_n a partire da S.

Se esiste una dimostrazione per la formula X a partire da S, scriveremo

$$S \vdash X$$

Se $S = \emptyset$, scriveremo

$$\vdash X$$

Gli assiomi che utilizzeremo sono i seguenti:

$$\begin{aligned}
\mathcal{A}_1 &= \left\{ \underline{X \Rightarrow (Y \Rightarrow X)} \mid X, Y \in \mathrm{FormI}_{\alpha\beta} \right\} \\
\mathcal{A}_2 &= \left\{ \underline{(X \Rightarrow (Y \Rightarrow Z)) \Rightarrow ((X \Rightarrow Y) \Rightarrow (X \Rightarrow Z))} \mid X, Y, Z \in \mathrm{FormI}_{\alpha\beta} \right\} \\
\mathcal{A}_3 &= \{ \underline{\mathbf{f} \Rightarrow X} \mid X \in \mathrm{FormI}_{\alpha\beta} \} \\
\mathcal{A}_4 &= \{ \underline{X \Rightarrow \mathbf{t}} \mid X \in \mathrm{FormI}_{\alpha\beta} \} \\
\mathcal{A}_5 &= \{ \underline{\neg\neg X \Rightarrow X} \mid X \in \mathrm{FormI}_{\alpha\beta} \} \\
\mathcal{A}_6 &= \left\{ \underline{X \Rightarrow (\neg X \Rightarrow Y)} \mid X, Y \in \mathrm{FormI}_{\alpha\beta} \right\} \\
\mathcal{A}_7 &= \left\{ \underline{\alpha \Rightarrow \alpha_1} \mid \alpha \in \mathrm{Form}_\alpha,\ \alpha_1 \text{ prima componente di } \alpha \right\} \\
\mathcal{A}_8 &= \left\{ \underline{\alpha \Rightarrow \alpha_2} \mid \alpha \in \mathrm{Form}_\alpha,\ \alpha_2 \text{ seconda componente di } \alpha \right\} \\
\mathcal{A}_9 &= \left\{ \underline{(\beta_1 \Rightarrow X) \Rightarrow ((\beta_2 \Rightarrow X) \Rightarrow (\beta \Rightarrow X))} \mid \beta \in \mathrm{Form}_\beta,\ \beta_1, \beta_2 \text{ componenti di } \beta \right\}
\end{aligned}$$

$$\mathcal{A} \stackrel{\mathrm{def}}{=} \bigcup_{i=1,\ldots,9} \mathcal{A}_i$$

Nota: tutti gli assiomi che utilizziamo sono tautologie!

L'unica regola di inferenza che useremo e' il *Modus Ponens*:

$$S \cup \{ \underline{X \Rightarrow Y}, X \} \vdash Y$$

ovvero, diremo che la seguente sequenza e' valida, e che la 3. e' stata ottenuta dalla 1., 2. con il m.p.

1. $X \Rightarrow Y$
2. X
3. Y

Proposition 4.71. *Data* $P \in \mathrm{FormI}_{\alpha\beta}$,

$$\vdash \underline{P \Rightarrow P}$$

Proof: Ecco la sequenza cercata:

1. $(P \Rightarrow ((P \Rightarrow P) \Rightarrow P)) \Rightarrow ((P \Rightarrow (P \Rightarrow P)) \Rightarrow (P \Rightarrow P))$ $\in \mathcal{A}_2$, con $X := P$, $Y := \underline{P \Rightarrow P}$, $Z := P$
2. $P \Rightarrow ((P \Rightarrow P) \Rightarrow P)$ $\in \mathcal{A}_1$, con $X := P$, $Y := \underline{P \Rightarrow P}$
3. $(P \Rightarrow (P \Rightarrow P)) \Rightarrow (P \Rightarrow P)$ *m.p.* da $1., 2.$
4. $(P \Rightarrow (P \Rightarrow P))$ $\in \mathcal{A}_1$, con $X := Y := P$
5. $P \Rightarrow P$ *m.p.* da $3., 4.$

□

Proposition 4.72. *Data* $X \in \mathrm{FormI}_{\alpha\beta}$,

$$\vdash X \Leftrightarrow S \vdash X \quad \forall S \subseteq \mathrm{FormI}_{\alpha\beta}$$

Proof:

⟨1⟩1. Dim. $\Rightarrow$

$\vdash X$ vuol dire che esiste una dimostrazione di X del tipo

$$X_1$$
$$\vdots$$
$$X_n = X$$

con $X_i \in \mathcal{A}$ oppure con X_i ottenuto dai precedenti $X_1, \ldots, X_{i-1}$ con il m.p.

In particolare possiamo dire che

$$X_i \in \mathcal{A} \Rightarrow X_i \in \mathcal{A} \cup S$$

e quindi possiamo considerare $X_1, \ldots, X_n$ come una dimostrazione di X a partire da S.

⟨1⟩2. Dim. $\Leftarrow$

Basta scegliere $S = \emptyset$.

□

Theorem 4.73. *di Deduzione di Tarski-Herbrand.*
Utilizzando un sistema di hilbert tale che $\mathcal{A}_1, \mathcal{A}_2 \subseteq \mathcal{A}$, *si ha*

$$S \vdash \underline{X \Rightarrow Y} \Leftrightarrow S \cup \{X\} \vdash Y$$

Nota: questo teorema e' l'analogo del teorema [4.12,pg.11]. Qui, pero', stiamo operando solamente sulla sintassi.
Nota: in questo teorema non stiamo facendo uso degli assiomi $\mathcal{A}_3, \ldots, \mathcal{A}_n$. *Richiediamo solo gli assiomi* $\mathcal{A}_1, \mathcal{A}_2$.

In particolare, con $S = \emptyset$ *otteniamo:*

$$\vdash \underline{X \Rightarrow Y} \Leftrightarrow \{X\} \vdash Y$$

Proof:

⟨1⟩1. Dim. $\Rightarrow$

Poiche' $S \vdash \underline{X \Rightarrow Y}$, siamo riusciti a trovare una dimostrazione da S del tipo:

$$X_1$$
$$X_2$$
$$\vdots$$
$$X_n = \underline{X \Rightarrow Y}$$

Se aggiungiamo X alla fine:

$$\begin{array}{l} X_1 \\ X_2 \\ \vdots \\ X_n = \underline{X \Rightarrow Y} \\ X \end{array}$$

otteniamo una derivazione da $S \cup \{X\}$. Infine, applicando m.p. alle ultime due otteniamo la dimostrazione cercata:

$$\begin{array}{l} X_1 \\ X_2 \\ \vdots \\ X_n = \underline{X \Rightarrow Y} \\ X \\ Y \end{array}$$

$\langle 1 \rangle 2$. Dim. $\Leftarrow$

Poiche' $S \cup \{X\} \vdash Y$, siamo riusciti a trovare una dimostrazione da $S \cup \{X\}$ del tipo:

$$\begin{array}{l} Z_1 \\ Z_2 \\ \vdots \\ Z_n = \underline{Y} \end{array}$$

$\langle 2 \rangle 1$. Per ogni $1 \leq i \leq n$, dimostriamo che

$$S \vdash \underline{X \Rightarrow Z_i}$$

Procediamo per induzione su i.

Case: $i = 1$

Necessariamente si deve avere $Z_1 \in \mathcal{A} \cup S \cup \{X\}$, e quindi rientriamo nel caso $Z_i \in \mathcal{A} \cup S$ o nel caso $Z_i = X$ che analizziamo qui di seguito. (Nota: questi due casi sono dimostrati senza far uso dell'ipotesi induttiva).

Case: $j < i \Rightarrow i$

Abbiamo tre possibili casi.

Case: $Z_i \in \mathcal{A} \cup S$

Adoperiamo la seguente dimostrazione a partire da S:

1. Z_i $\in \mathcal{A} \cup S$
2. $(Z_i \Rightarrow (X \Rightarrow Z_i))$ $\in \mathcal{A}_1$
3. $X \Rightarrow Z_i$ m.p. da 1.,2.

Case: $Z_i = X$

Per la prop [4.71,pg.49], conosciamo gia' la dimostrazione di

$$X \Rightarrow X$$

Che possiamo anche scrivere come

$$X \Rightarrow Z_i$$

Nota: la dimostrazione di $X \Rightarrow X$ parte da $\emptyset$ e quindi si puo' considerare come una dimostrazione che parte da S (vedi [4.72,pg.49])

Case: Z_i ottenuto da Z_h, Z_j con il m.p., dove $h, j < i$

Supponiamo che $Z_j = \underline{Z_h \Rightarrow Z_i}$. Nella derivazione $Z_1, \ldots, Z_n$ abbiamo quindi un pezzo

del tipo:

$$Z_j = \underline{Z_h \Rightarrow Z_i}$$
$$Z_h$$
$$Z_i$$

Per Hp induttiva riusciamo a dimostrare da S le seguenti formule:

$$X \Rightarrow Z_h$$
$$X \Rightarrow Z_j$$

ovvero, abbiamo una dimostrazione a partire da S del tipo:

$$\vdots$$
$$X \Rightarrow Z_h$$
$$\vdots$$
$$X \Rightarrow Z_j$$

da questa creiamo la seguente:

$\vdots$

1. $X \Rightarrow Z_h$

$\vdots$

2. $X \Rightarrow Z_j = \underline{Z_h \Rightarrow Z_i}$
3. $(X \Rightarrow (Z_h \Rightarrow Z_i)) \Rightarrow ((X \Rightarrow Z_h) \Rightarrow (X \Rightarrow Z_i)) \quad \in \mathcal{A}_2,$ con $X := X,\ Y := Z_h,\ Z := Z_i$
4. $(X \Rightarrow Z_h) \Rightarrow (X \Rightarrow Z_i)$ m.p. da 2.,3.

$X \Rightarrow Z_i$ m.p. da 4.,1.

Abbiamo cosi' trovato una dimostrazione di $X \Rightarrow Z_i$ a partire da S.

$\langle 2 \rangle 2$. Q.E.D.

Per il passo precedente riusciamo a costruire, a partire da S, la seguente dimostrazione:

$$X \Rightarrow Z_1$$
$$X \Rightarrow Z_2$$
$$\vdots$$
$$X \Rightarrow Z_n = Y$$

Siamo cosi' riusciti a dimostrare $X \Rightarrow Y$ a partire da S.

□

Proposition 4.74. *Data* $X \in \mathrm{Form}\mathrm{I}_{\alpha\beta}$,

$$\vdash \underline{(\neg X \Rightarrow X) \Rightarrow X}$$

Proof: La formula $\neg X \Rightarrow X$ e' una β, possiamo quindi usare l'assioma $\mathcal{A}_9$:

1. $(\neg\neg X \Rightarrow X) \Rightarrow ((X \Rightarrow X) \Rightarrow ((\neg X \Rightarrow X) \Rightarrow X))$ $\in \mathcal{A}_9$
2. $\neg\neg X \Rightarrow X$ $\in \mathcal{A}_5$
3. $(X \Rightarrow X) \Rightarrow ((\neg X \Rightarrow X) \Rightarrow X)$ m.p. da 1.,2.
4. $X \Rightarrow X$ dalla prop [4.71,pg.49]
5. $(\neg X \Rightarrow X) \Rightarrow X$ m.p da 3.,4.

□

Theorem 4.75. *di correttezza e completezza forte per i sistemi di Hilbert.*

$$S \vdash X \Leftrightarrow S \models X$$

Proof:
⟨1⟩1. Dim. $\Rightarrow$
Supponiamo $\exists v : \ v(S) = \mathbf{t}$ e dimostriamo che $v(X) = \mathbf{t}$.
Per Hp esiste la seguente dimostrazione a partire da S:

$$X_1$$
$$\vdots$$
$$X_n = X$$

Dimostriamo per induzione che $v(X_i) = \mathbf{t}$ per ogni i.
CASE: $i = 1$

$$X_1 \in \mathcal{A} \cup S$$

CASE: $X_1 \in \mathcal{A}$

$$v(X_1) = \mathbf{t}$$

perche' i nostri assiomi sono delle tautologie.
CASE: $X_1 \in S$

$$v(S) = \mathbf{t} \Rightarrow v(X_1) = \mathbf{t}$$

CASE: $j < i \Rightarrow i$
CASE: $X_i \in \mathcal{A} \cup S$
Analogamente a prima.
CASE: X_i ottenuto da X_h, X_j con il m.p, dove $h, j < i$
Supponiamo $X_j = \underline{X_h \Rightarrow X_i}$, allora nella dimostrazione $X_1, \ldots, X_n$ abbiamo

$$\vdots$$
$$X_j = \underline{X_h \Rightarrow X_i}$$
$$X_h$$
$$X_i$$

Per Hp induttiva $v(X_h) = v(X_j) = \mathbf{t}$,

$$v(X_j) \underbrace{=}_{\text{definizione di valutazione}} \left(\underbrace{v(X_h)}_{=\mathbf{t}} \Rightarrow v(X_i) \right) = \mathbf{t} \ \Rightarrow \ v(X_i) = \mathbf{t}$$

⟨1⟩2. Dim. $\Leftarrow$
Sia dato il seguente insieme:

$$C = \{W \subseteq \mathrm{Form}\mathrm{I}_{\alpha\beta} \mid W \nvdash X\}$$

Supponiamo per assurdo che $S \nvdash X$, ovvero $S \in C$.
⟨2⟩1. Dim. che $S \cup \{\neg X\} \in C$
Infatti, se per assurdo $S \cup \{\neg X\} \vdash X$ abbiamo

$$S \cup \{\neg X\} \vdash X \underbrace{\Leftrightarrow}_{[4.73,\text{pg.}49]} S \vdash \underline{\neg X \Rightarrow X}$$

abbiamo quindi la seguente dimostrazione di $\neg X \Rightarrow X$ a partire da S:

$$\begin{array}{l} X_1 \\ \vdots \\ X_n = \underline{\neg X \Rightarrow X} \end{array}$$

E per la prop [4.74,pg.51], possiamo estenderla cosi':

$$\begin{array}{ll} X_1 & \\ \vdots & \\ 1.\ X_n = \underline{\neg X \Rightarrow X} & \\ 2.\ (\neg X \Rightarrow X) \Rightarrow X & \text{per la [4.74,pg.51]} \\ 3.\ X & \text{m.p. da 1. e 2.} \end{array}$$

Abbiamo cosi' dimostrato X a partire da S. Assurdo contro $S \in C$.

$\langle 2 \rangle 2$. Dimostriamo che C e' una proprieta' di consistenza proposizionale (vedi [4.51,pg.29])

Sia $W \in C$ e dimostriamo che rispetta le condizioni [4.51,pg.29]

$\langle 3 \rangle 1$. $\mathbf{f} \notin W$

Se per assurdo, $\mathbf{f} \in W$, possiamo costruire la seguente dimostrazione a partire da W:

$$\begin{array}{ll} \mathbf{f} & \in W \\ \mathbf{f} \Rightarrow X & \in \mathcal{A}_3 \\ X & \text{m.p.} \end{array}$$

e quindi $W \vdash X$, assurdo contro $W \in C$

$\langle 3 \rangle 2$. $\neg \mathbf{t} \notin W$

Analogamente a prima, costruiamo la seguente dimostrazione di X:

$$\begin{array}{ll} 1.\ \neg \mathbf{t} & \in W \\ 2.\ \neg \mathbf{t} \Rightarrow \mathbf{t} & \in \mathcal{A}_4 \\ 3.\ \mathbf{t} & \text{m.p.} \\ 4.\ \mathbf{t} \Rightarrow (\neg \mathbf{t} \Rightarrow X) & \in \mathcal{A}_6 \\ 5.\ \neg \mathbf{t} \Rightarrow X & \text{m.p} \\ 6.\ X & \text{m.p. da 1.,5.} \end{array}$$

Quindi $W \vdash X$ assurdo.

$\langle 3 \rangle 3$. $P \in \text{Var} \Rightarrow P \notin W$ oppure $\neg P \notin W$

Supponiamo per assurdo che $P, \neg P \in W$, allora

$$\begin{array}{ll} P & \in W \\ \neg P & \in W \\ P \Rightarrow (\neg P \Rightarrow X) & \in \mathcal{A}_6 \\ \neg P \Rightarrow X & \text{m.p.} \\ X & \text{m.p.} \end{array}$$

Quindi $W \vdash X$ assurdo.

$\langle 3 \rangle 4$. $\neg\neg Z \in W \Rightarrow W \cup \{Z\} \in C$

Supponiamo per assurdo che $W \cup \{Z\} \vdash X$,

$$W \cup \{Z\} \vdash X \underbrace{\Leftrightarrow}_{\text{[4.73,pg.49]}} W \vdash \underline{Z \Rightarrow X}$$

Allora esiste una dimostrazione di $Z \Rightarrow X$ a partire da W:

$$\begin{array}{l} X_1 \\ \vdots \\ X_n = Z \Rightarrow X \end{array}$$

e la estendiamo cosi'

$$\begin{array}{lr} X_1 & \\ \vdots & \\ 1.\ X_n = Z \Rightarrow X & \\ 2.\ \neg\neg Z & \in W \\ 3.\ \neg\neg Z \Rightarrow Z & \in \mathcal{A}_5 \\ 4.\ Z & \text{m.p. da 2.,3.} \\ X & \text{m.p. da 1.,4.} \end{array}$$

Quindi $W \vdash X$. Assurdo.

$\langle 3 \rangle 5.\ \alpha \in W \Rightarrow W \cup \{\alpha_1, \alpha_2\} \in C$

Come prima:

$$W \cup \{\alpha_1, \alpha_2\} \vdash X \underbrace{\Leftrightarrow}_{\text{[4.73,pg.49]}} W \vdash \underline{\alpha_1 \Rightarrow (\alpha_2 \Rightarrow X)}$$

Quindi,

$$\begin{array}{l} X_1 \\ \vdots \\ X_n = \underline{\alpha_1 \Rightarrow (\alpha_2 \Rightarrow X)} \end{array}$$

che estendiamo con

$$\begin{array}{lr} X_1 & \\ \vdots & \\ X_n = \underline{\alpha_1 \Rightarrow (\alpha_2 \Rightarrow X)} & \\ \alpha & \in W \\ \alpha \Rightarrow \alpha_1 & \in \mathcal{A}_7 \\ \alpha \Rightarrow \alpha_2 & \in \mathcal{A}_8 \\ \alpha_1 & \text{m.p.} \\ \alpha_2 & \text{m.p.} \\ \alpha_2 \Rightarrow X & \text{m.p.} \\ X & \text{m.p.} \end{array}$$

Assurdo.

$\langle 3 \rangle 6.\ \beta \in W \Rightarrow W \cup \{\beta_1\} \in C$ oppure $W \cup \{\beta_2\} \in C$

Supponiamo per assurdo che

$$\begin{array}{l} W \cup \{\beta_1\} \vdash X \\ W \cup \{\beta_2\} \vdash X \end{array}$$

Per il thm [4.73,pg.49]

$$\begin{array}{l} W \vdash \beta_1 \Rightarrow X \\ W \vdash \beta_2 \Rightarrow X \end{array}$$

Quindi

X_1

$\vdots$

1. $\beta_1 \Rightarrow X$
2. $\beta_2 \Rightarrow X$
3. β $\beta \in W$
4. $(\beta_1 \Rightarrow X) \Rightarrow ((\beta_2 \Rightarrow X) \Rightarrow (\beta \Rightarrow X))$ $\in \mathcal{A}_9$
5. $(\beta_2 \Rightarrow X) \Rightarrow (\beta \Rightarrow X)$ m.p. da 1.,4.
6. $\beta \Rightarrow X$ m.p. da 2.,5.
7. X m.p. da 3.,6.

Assurdo.

$\langle 2 \rangle 3$. Q.E.D.

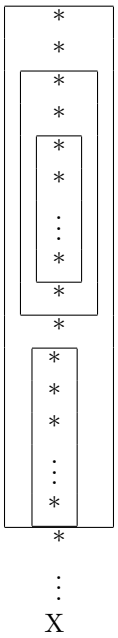
$$\begin{cases} C \text{ proprieta' di consistenza} \\ S \cup \{\neg X\} \in C \end{cases} \underbrace{\Rightarrow}_{[4.59,\text{pg.}37]} S \cup \{\neg X\} \text{ soddisfacibile } \Rightarrow \exists v \in \mathcal{V} : \ v(S) = \mathbf{t}, \ v(X) = \mathbf{f}$$

assurdo contro $S \models X$

□

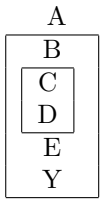
4.9 Sistema di deduzione naturale

Definition 4.76. Una dimostrazione di $X \in \text{FormI}_{\alpha\beta}$ in un sistema di deduzione naturale a partire da $S \subseteq \text{FormI}_{\alpha\beta}$ e' una sequenza di scatole annidate del tipo:



Dove ogni $*$ e' una qualsiasi formula di $\text{FormI}_{\alpha\beta}$. L'ultima formula della sequenza e' X.
Per costruire la sequenza, vengono rispettate le seguenti condizioni:

- Se Y e' formula della sequenza, allora si verifica una delle seguenti condizioni (che chiamiamo $C1, C2, C3$):

  1. $Y \in S$
  2. oppure, Y e' derivata tramite le regole di deduzione dalle formule attive. Le formule attive, sono tutte le formule della sequenza che stanno sopra Y e che non stanno in scatole gia' chiuse. Ad esempio, nella sequenza 

Le formule attive per Y sono A, B, E.
  3. oppure, Y e' la prima formula di una scatola. (In pratica, all'inizio di una scatola possiamo inserire qualsiasi formula).

Se esiste una dimostrazione per X a partire da S scriveremo

$$S \vdash X$$

Le regole di deduzione naturale sono le seguenti:

$D1.$ $\mathbf{f} \vdash X \quad \forall X \in \mathrm{FormI}_{\alpha\beta}$

$D2.$ $\vdash \mathbf{t}$

$D3.$ $X,\ \neg X \vdash \mathbf{f} \quad \forall X \in \mathrm{FormI}_{\alpha\beta}$

$D4.$ $\alpha \vdash \alpha_1, \alpha_2 \quad \forall \alpha \in \mathrm{Form}_{\alpha}$

$D5.$ $\alpha_1, \alpha_2 \vdash \alpha \quad \forall \alpha \in \mathrm{Form}_{\alpha}$

$D6.$ $\neg\beta_1,\ \beta \vdash \beta_2 \quad \forall \beta \in \mathrm{Form}_{\beta}$

$D7.$ $\neg\beta_2,\ \beta \vdash \beta_1 \quad \forall \beta \in \mathrm{Form}_{\beta}$

$D8.$ $\boxed{\begin{array}{c}\neg\beta_1 \\ \vdots \\ \beta_2\end{array}} \vdash \beta \ \ \forall \beta \in \mathrm{Form}_{\beta}$

$D9.$ $\boxed{\begin{array}{c}\neg\beta_2 \\ \vdots \\ \beta_1\end{array}} \vdash \beta \ \ \forall \beta \in \mathrm{Form}_{\beta}$

$D10.$ $\boxed{\begin{array}{c}\mathrm{X} \\ \vdots \\ \mathbf{f}\end{array}} \vdash \neg X \ \ \forall X \in \mathrm{FormI}_{\alpha\beta}$

$D11.$ $\boxed{\begin{array}{c}\neg X \\ \vdots \\ \mathbf{f}\end{array}} \vdash X \ \ \forall X \in \mathrm{FormI}_{\alpha\beta}$

La $D4$ viene chiamata α-eliminazione, la $D5$ viene chiamata α-introduzione. Analogamente, le $D6, D7, D8, D9$ sono chiamate β-eliminazione e β-introduzione.

Queste regole dicono che se a un certo punto della dimostrazione abbiamo una formula che rispetta un loro primo membro, allora possiamo aggiungere alla dimostrazione il relativo secondo membro, immediatamente dopo il primo membro. Ad esempio, $\begin{array}{|c|}\hline \neg\beta_1 \\ \vdots \\ \beta_2 \\ \hline \end{array}$ β

Proposition 4.77.

1. $\neg Y,\ \underline{X \Rightarrow Y} \vdash \neg X$ *[si chiama Modus Tolles*
2. $\neg\neg X,\ \underline{X \Rightarrow Y} \vdash Y$ *[si chiama Quasi Modus Ponens]*
3. $X,\ \underline{X \Rightarrow Y} \vdash Y$ *[modus ponens]*
4. $\neg\neg X \vdash X$
5. $X \vdash \neg\neg X$
6. $\begin{array}{|c|}\hline X \\ \vdots \\ Y \\ \hline \end{array} \vdash \underline{X \Rightarrow Y}$

Proof:
⟨1⟩1. Dim 4.

$$\begin{array}{rl} \neg\neg X & \text{C1} \\ \boxed{\begin{array}{c}\neg X \\ \mathbf{f}\end{array}} & \text{C3,D3} \\ \text{X} & \text{D11} \end{array}$$

⟨1⟩2. Dim 5.

$$\begin{array}{rl} \text{X} & \text{C1} \\ \boxed{\begin{array}{c}\neg X \\ \mathbf{f}\end{array}} & \text{C3, D3} \\ \neg\neg X & \text{D11} \end{array}$$

⟨1⟩3. Dim 1.

1. $\neg Y$ $C1$
2. $X \Rightarrow Y$ $C1$

Nota: $X \Rightarrow Y$ e' una β con componenti $\neg X, Y$

3. $\neg X$ $D7$ applicata a $1., 2.$

⟨1⟩4. Dim 3.
Come prima:

1. X $C1$
2. $\neg\neg X$ per il passo ⟨1⟩2
3. $\underbrace{X \Rightarrow Y}_{\beta_1=\neg X,\ \beta_2=Y}$ $C1$
4. Y $D6$ applicata a $2., 3.$

⟨1⟩5. Dim 2.

Come prima:

1. $\neg\neg X$ — $C1$
2. X — per il passo $\langle 1\rangle 1$
3. $X \Rightarrow Y$ — $C1$
4. Y — per il passo $\langle 1\rangle 4$ applicato a 2.,3.

$\langle 1\rangle 6$. Dim 6.

1. $\boxed{\begin{array}{c}\neg\neg X \\ \text{X (per il passo } \langle 1\rangle 1) \\ \text{Y (per Hp)}\end{array}}$
2. $\neg X \vee Y$ — $D8$

e la 2. e' proprio $X \Rightarrow Y$

Nota: nella scatola abbiamo potuto scrivere X, Y perche' per ipotesi abbiamo gia' una scatola valida del tipo $X, \ldots, Y$. Quindi, in quest'ultima dimostrazione e' come se avessimo ricopiato la scatola dell'ipotesi.

□

Theorem 4.78.

$$S \vdash X \Leftrightarrow S \models X$$

Proof:

$\langle 1\rangle 1$. Dim. $\Rightarrow$

Supponiamo che $v(S) = \mathbf{t}$ e dimostriamo che $v(X) = \mathbf{t}$.

Per Hp esiste la dimostrazione di X, che sara' una sequenza di formule $X_1, \ldots, X_n$ con $X_n = X$, opportunamente inscatolata.

Consideriamo tutte le assunzioni precedenti a X_i:

$A_i = \{$tutte le formule X_j poste ad inizio di una scatola in cui X_i e' presente, con $j \leq i\}$

Dimostriamo per induzione che $A_i \cup S \models X_i$.

Case: $i = 1$

Per le condizioni $C1, C2, C3$, abbiamo solo tre casi:

$X_1 \in S$

X_1 e' una assunzione

$X_1 = \mathbf{t}$ e' stata ricavata con la regola D_2

Nei primi due casi:

$$X_1 \in A_1 \cup S \Rightarrow A_1 \cup S \models X_1$$

Nell'ultimo:

$$v(X_1) = \mathbf{t} \ \ \forall v \in \mathcal{V}$$

Case: $m < n \Rightarrow n$

Se X_n e' una assunzione oppure $X_n \in S$, ragionando come prima abbiamo la tesi. Supponiamo quindi $X_n \notin S,\ X_n \notin A_n$.

Allora, X_n e' stata ricavata con la C_2, ovvero e' stata applicata qualche regola D_i.

Tutte le regole $D_1, D_3, D_4, D_5, D_6, D_7$ preservano la soddisfacibilita', ad esempio, la D_5:

$X_n = \alpha$

$\alpha_1 = X_h,\ \alpha_2 = X_j,\ \ h < j < n$

Hp induttiva $\Rightarrow S \cup A_h \models \alpha_1,\ \ S \cup A_j \models \alpha_2 \ \ (*)$

Per utilizzare la D_5 le α_1, α_2 devono essere formule attive per X_n, ovvero non devono stare in scatole chiuse in cui X_n non e' presente. In altre parole,

$$A_h, A_j \subseteq A_n$$

quindi
$v(S\cup A_n) = \mathbf{t} \underbrace{\Rightarrow}_{A_h, A_j \subseteq A_n} v(S\cup A_h) = v(S\cup A_j) = \mathbf{t} \underbrace{\Rightarrow}_{(*)} v(\alpha_1) = v(\alpha_2) = \mathbf{t} \Rightarrow v(\alpha) = \mathbf{t} \Leftrightarrow v(X_n) = \mathbf{t}$

Quindi $S\cup A_n \models X_n$

Analogamente si vede per gli altri $D_i,\ i \leq 7$.

Esaminiamo una D_i con una scatola, ad esempio la D_9. In questo caso $X_n = \beta$, e immediatamente prima di X_n, e' presente la scatola $\boxed{\begin{array}{c}\neg\beta_2\\ \vdots \\ \beta_1\end{array}}$

Con $\beta_1 = X_m$, $m = n-1$. Per Hp induttiva abbiamo:

$$S\cup A_m \models \beta_1 \quad (1)$$

Per definizione di A_m si ha

$$\neg\beta_2 \in A_m$$

Poiche' la scatola e' chiusa si ha

$$A_m \setminus \{\neg\beta_2\} = A_{m+1} = A_n \Leftrightarrow A_n \cup \{\neg\beta_2\} \supseteq A_m \quad (2)$$

Quindi si $v \in \mathcal{V}:\ \ v(A_n \cup S) = \mathbf{t}$,

Case: $v(\neg\beta_2) = \mathbf{t}$

$$v(A_n\cup S) = v(\neg\beta_2) = \mathbf{t} \Rightarrow v(A_n\cup S\cup\{\neg\beta_2\}) = \mathbf{t} \underbrace{\Rightarrow}_{(2)} v(A_m\cup S) = \mathbf{t} \underbrace{\Rightarrow}_{(1)} v(\beta_1) = \mathbf{t} \Rightarrow v(\beta) = \mathbf{t} \Leftrightarrow v(X_n) = \mathbf{t}$$

Case: $v(\neg\beta_2) = \mathbf{f}$

$$v(\neg\beta_2) = \mathbf{f} \Leftrightarrow v(\beta_2) = \mathbf{t} \Rightarrow v(\beta) = \mathbf{t}$$

In entrami i casi $v(X_n) = \mathbf{t}$, e quindi per l'arbitrarieta' di $v \in \mathcal{V}$ abbiamo dimostrato che

$$A_n \cup S \models X_n$$

Nota[7].

Analogamente si procede per i casi $D8, D10, D11$.

Poiche' alla fine della dim tutte le scatole sono chiuse, $A_n = \emptyset$. In conclusione

$$A_n = \emptyset \Rightarrow S \models X_n = X$$

$\langle 1\rangle 2$. Dim. $\Leftarrow$

Supponiamo per assurdo che $S \nvdash X$.

Sia

$$C = \{W \in \mathrm{Form}_{\mathrm{I}\alpha\beta} \mid W \nvdash X\}$$

$\langle 2\rangle 1$. Dim. $S\cup\{\neg X\} \in C$

Se per assurdo

$$S\cup\{\neg X\} \vdash X$$

sia avrebbe una dimostrazione del tipo $\begin{array}{c}\mathrm{A}\\ \vdots \\ \neg X \\ \vdots \\ \mathrm{X}\end{array}$ Allora, per la $D3$,, possiamo scrivere: $\begin{array}{c}\mathrm{A}\\ \vdots \\ \neg X \\ \vdots \\ \mathrm{X} \\ \mathbf{f}\end{array}$

Utilizzando questa sequenza con la $D11$, possiamo costruire la seguente dimostrazione $S \vdash X$:

[7] In pratica, quando abbiamo una scatola chiusa, non importa piu' quale valutazione usiamo per le sue assunzioni

$$\dfrac{\dfrac{\boxed{\begin{array}{c}\neg X\\ \text{A}\\ \vdots\\ \neg X\\ \vdots\\ \text{X}\\ \mathbf{f}\end{array}}}{\text{X}}}{}$$

Assurdo contro $S \nvdash X$.

$\langle 2 \rangle 2$. Dim. C e' una proprieta' di consistenza proposizionale

1. Se per assurdo $P, \neg P \in W$, allora possiamo costruire la seguente dimostrazione:

$$\begin{array}{ll} P & \\ \neg P & \\ \mathbf{f} & D3 \\ X & D1 \end{array}$$

   Quindi $W \vdash X$ assurdo.
2. Analogamente, si deve avere $\mathbf{f}, \neg \mathbf{t} \notin W$
3. Supponiamo $\neg\neg Z \in W$ e supponiamo per assurdo che $W \cup \{Z\} \notin C$. Allora, esiste la seguente dimostrazione $W \cup \{Z\} \vdash X$:

$$\begin{array}{c} Z \\ \vdots \\ X \end{array}$$

   Possiamo costruire la seguente:

$$\begin{array}{ll} \neg\neg Z & \\ Z & \text{prop [4.77,pg.57]} \\ X & \text{per la dim. } W \cup \{Z\} \vdash X \text{ di prima} \end{array}$$

   Quindi $W \vdash X$, assurdo.
4. Sia $\alpha \in W$ supponiamo per assurdo che $W \cup \{\alpha_1, \alpha_2\} \notin C$.
   Procedendo come prima e usando la $D4$ otteniamo l'assurdo.
5. Sia $\beta \in W$ supponiamo per assurdo che $W \cup \{\beta_1\} \notin C$, $W \cup \{\beta_2\} \notin C$.

   Abbiamo quindi due dimostrazioni: $\begin{array}{c}\beta_1\\ \vdots\\ \text{X}\end{array}$ e $\begin{array}{c}\beta_2\\ \vdots\\ \text{X}\end{array}$ Sfruttando queste due, costruiamo la seguente

$$\boxed{\begin{array}{c}\neg X\\ \boxed{\begin{array}{c}\beta_1\\ \text{X (perche' } W \cup \{\beta_1\} \vdash X)\\ \mathbf{f}\end{array}}\\ \neg\beta_1 \text{ (D10)}\\ \beta \ (\beta \in W)\\ \beta_2 \text{ (D6)}\\ \text{X (perche' } W \cup \{\beta_2\} \vdash X)\\ \mathbf{f} \text{ (D3 su } \neg X, X)\end{array}}$$

$$\text{X (D11)}$$

   Abbiamo cosi' dimostrato $W \vdash X$. Assurdo.

□

4.10 Procedura di Davis Putnam

Definition 4.79. Sia S un insieme di disgiunzioni letterali (clausole) e sia $L \in \text{Lit}$. Definiamo il seguente insieme:

$$S_L \stackrel{\text{def}}{=} \{C \setminus \{\overline{L}\} \mid C \in S,\ \overline{L} \in C,\ L \notin C\} \cup \{C \in S \mid \overline{L} \notin C,\ L \notin C\}$$

dove $\overline{L}$ e' la negazione di L (vedi [4.17,pg.12]), con la seguente modifica: $\overline{\neg X} = X$

In altre parole, S_L e' ricavato da S includendo tutte le clausole che non avevamo L ne' $\overline{L}$ e eliminando $\overline{L}$ da quelle che lo contenevano.

Il nostro obbiettivo sara' quello di considerare S come una CNF. Cosi' facendo, S_L si puo' interpretare come la CNF ottenuta da S ponendo $L = \mathbf{t}$ ed effettuando opportune semplificazioni.

Lemma 4.80. *Sia S un insieme di clausole, $L \in \text{Lit}$ e sia $\langle S \rangle$ la congiunzione generalizzata $\langle X_1, \ldots, X_n \rangle$ con $S = \{X_1, \ldots, X_n\}$. Si ha*

$$\models \underline{\langle S \rangle \wedge L \Leftrightarrow \langle S_L \rangle \wedge L}$$

Proof: Per la dimostrazione di questa tautologia facciamo ricorso alla prop [4.12,pg.11]: dimostreremo le seguenti:

$$\underline{\langle S \rangle \wedge L} \models \underline{\langle S_L \rangle \wedge L}$$
$$\underline{\langle S_L \rangle \wedge L} \models \underline{\langle S \rangle \wedge L}$$

$\langle 1 \rangle 1$. Dim. $\Rightarrow$

Il primo punto chiave e' osservare che vale la seguente:

$$(\overline{L} \vee X_1 \vee \ldots \vee X_n) \wedge L \Leftrightarrow (X_1 \vee \ldots \vee X_n) \wedge L$$

In pratica, se L e' vera, allora possiamo eliminare $\overline{L}$ dalla disgiunzione: $\overline{L}$ e' falsa e quindi e' inutile includerla nella disgiunzione.

Il secondo punto chiave e':

$$(L \vee X_1 \vee \ldots \vee X_n) \wedge (Y_1 \vee \ldots \vee Y_m) \wedge L \Leftrightarrow (Y_1 \vee \ldots \vee Y_m) \wedge L$$

In altre parole, se abbiamo una disgiunzione in cui un elemento e' vero, allora possiamo eliminare l'intera disgiunzione perche' sappiamo che e' tutta vera.

Procediamo in dettaglio. Sia v tale che

$$v \models \underline{\langle S \rangle \wedge L} \quad (1)$$

dimostriamo che

$$v \models \langle S_L \rangle \wedge L$$

Si ha

$$(1) \Rightarrow v(C) = v(L) = \mathbf{t} \ \ \forall C \in S \quad (1.1)$$

Sia $C = \{Y_1, \ldots, Y_n\} \in S : \ \exists i : \ \overline{L} = Y_i$

$v(L) = \mathbf{t} \Rightarrow v(\overline{L}) = \mathbf{f} \Leftrightarrow v(Y_i) = \mathbf{f}$

$(1.1) \Rightarrow v(C) = \mathbf{t} \Rightarrow \exists j : \ v(Y_j) = \mathbf{t}$

$\Rightarrow j \neq i$

$\Rightarrow v(C \setminus \{\overline{L}\}) = \mathbf{t}$ Nota: stiamo considerando C come disgiunzione (2)

Sia $C \in S: \ \overline{L} \notin C, \ L \notin C$

$$(1.1) \Rightarrow v(C) = \mathbf{t} \ \ (3)$$

Per la $(2), (3)$ e per la definizione di S_L abbiamo

$$\forall C' \in S_L \ \ v(C') = \mathbf{t}$$

e quindi la tesi.

$\langle 1 \rangle 2$. Dim. $\Leftarrow$

Analogamente.

□

Lemma 4.81. *Sia S un insieme di clausole e L un letterale*

$$S \text{ soddisfacibile } \Leftrightarrow S_L \text{ soddisfacibile } \vee S_{\overline{L}} \text{ soddisfacibile}$$

Proof:

$\langle 1 \rangle 1$. Dim. $\Rightarrow$

Supponiamo $\exists v \in \mathcal{V}: \ v(S) = \mathbf{t}$.

Nota: con $v(S) = \mathbf{t}$ intendiamo $v(\langle S \rangle) = \mathbf{t}$, ovvero $v(C) = \mathbf{t} \ \ \forall C \in S$.

Abbiamo due casi

Case: $v(L) = \mathbf{t}$

Allora,

$$v(S) = \mathbf{t}, \ v(L) = \mathbf{t} \underbrace{\Rightarrow}_{\text{[4.80,pg.61]}} v(S_L) = \mathbf{t}$$

quindi S_L e' soddisfacibile.

Case: $v(L) = \mathbf{f}$

Allora

$$v(L) = \mathbf{f} \Rightarrow v(\overline{L}) = \mathbf{t}$$

e quindi per il lemma [4.80,pg.61] si ha

$$v(S_{\overline{L}}) = \mathbf{t}$$

$\langle 1 \rangle 2$. Dim. $\Leftarrow$

Supponiamo S_L soddisfacibile: $\exists v: \ v(S_L) = \mathbf{t}$.

Case: $v(L) = \mathbf{t}$

Basta usare il lemma [4.80,pg.61]

Case: $v(L) = \mathbf{f}$

Poiche' in S_L non esistono clausole contenenti $L, \overline{L}$, la seguente valutazione soddisfera' ancora S_L:

$$v_L(P) = \begin{cases} v(P) & P \notin \{L, \overline{L}\} \\ \mathbf{t} & P = L \\ \mathbf{f} & P = \overline{L} \end{cases}$$

e in piu' soddisfera' anche L. Quindi, applicando ancora il lemma [4.80,pg.61] si ha la tesi.

□

Lemma 4.82. *Sia S un insieme di clausole e L un letterale. Supposto $\{L\} \in S$ (ovvero la clausola singola $\{L\}$ sta' in S), si ha*

$$S \text{ soddisf } \Leftrightarrow S_L \text{ soddisf}$$

In altre parole, se in una CNF abbiamo la clausola $[L]$, per soddisfare la CNF siamo costretti a porre $v(L) = \mathbf{t}$.

Proof:

$\langle 1\rangle 1$. Dim. $\Rightarrow$

$$S \text{ soddisf} \Rightarrow \exists v \in \mathcal{V} : \ v(S) = \mathbf{t} \Rightarrow v(\{L\}) = \mathbf{t} \Rightarrow v(L) = \mathbf{t}$$
$$v(S) = \mathbf{t},\ v(L) = \mathbf{t} \underbrace{\Rightarrow}_{[4.80,\text{pg.}61]} v(S_L) = \mathbf{t}$$

$\langle 1\rangle 2$. Dim. $\Leftarrow$

$$S_L \text{ soddisf} \underbrace{\Rightarrow}_{[4.81,\text{pg.}62]} S \text{ soddisf}$$

□

Lemma 4.83. *Sia S un insieme di clausole, L un letterale. Supposto che*

1. $\exists C \in S : \ L \in C$
2. $\forall C \in S \ \ \overline{L} \notin C$

si ha

$$S \ \textit{soddisf} \Leftrightarrow S_L \ \textit{soddisf}$$

Proof:
$\langle 1\rangle 1$. Dim. $\Rightarrow$
In questo caso, per definizione di S_L si ha

$$\forall C \in S \ \ \overline{L} \notin C \underbrace{\Rightarrow}_{\text{def. di } S_L} S_L = \left\{C \in S \mid \overline{L} \notin C,\ L \notin C\right\} \subseteq S$$

$$\begin{cases} S_L \subseteq S \\ S \text{ soddisf.} \end{cases} \Rightarrow S_L \text{ soddisf.}$$

$\langle 1\rangle 2$. Dim. $\Leftarrow$

$$S_L \text{ soddisf} \underbrace{\Rightarrow}_{[4.81,\text{pg.}62]} S \text{ soddisf}$$

□

Lemma 4.84. *Sia S un insieme di clausole. Supposto che*

$$\exists C, C' \in S : \ C' \supsetneq C$$

si ha

$$S \ \textit{soddisf} \Leftrightarrow S \setminus \{C'\} \ \textit{soddisf}$$

Proof:
$\langle 1\rangle 1$. Dim. $\Rightarrow$
Basta osservare che

$$S \setminus \{C'\} \subseteq S$$

$\langle 1\rangle 2$. Dim. $\Leftarrow$

$$C \neq C',\ C \in S \Rightarrow C \in S \setminus \{C'\}$$
$$v(S \setminus \{C'\}) = \mathbf{t} \underbrace{\Rightarrow}_{C \in S \setminus \{C'\}} v(C) = \mathbf{t} \underbrace{\Leftrightarrow}_{C \text{ e' una disgiunzione}} \exists X \in C : \ v(X) = \mathbf{t} \underbrace{\Rightarrow}_{X \in C \subseteq C',\ C' \text{ disgiunzione}} v(C') = \mathbf{t}$$

□

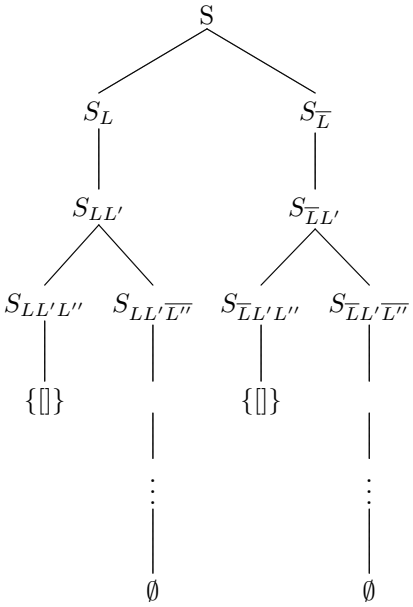
Definition 4.85. Procedura di Putnam per insiemi di clausole.
Sia S un insieme di clausole, con $|S| < \infty$.

1. Per ogni $C \in S$
   (a) eliminiamo C da S se $\mathbf{t} \in C$
   (b) eliminiamo C da S se $A, \neg A \in C$
   (c) eliminiamo tutti i simboli $\mathbf{f}$ da C
   (d) eliminiamo tutte le ripetizioni di letterali contenute in C (Nota[8])
2. Cancelliamo da S tutte le clausole ripetute.
3. Se $\exists C, C' \in S : \ C' \supsetneq C$ allora si sostituisce S con $S \setminus \{C'\}$
4. Per ogni letterale L contenuto in qualche clausola di S:
   (a) Se $\{L\} \in S$ si sostituisce S con S_L.
   (b) Se e' vero che

$$\exists C \in S : \ L \in C$$
$$\forall C \in S \ \ \overline{L} \notin C$$

   allora si sostituisce S con S_L.
   (c) Se i due casi di sopra non si sono verificati, si sostituisce S con $S_L \vee S_{\overline{L}}$. (regola di splitting).

Iterando la procedura si otteniene un albero del tipo



[8] in realta', se consideriamo C un insieme, non sarebbe necessario utilizzare questa regola, ma nelle implementazioni algoritmiche, si deve utilizzare

Chiaramente si crea una biforcazione quando si usa la regola dello splitting.
Un albero di Putnam per la formula X viene formato cosi':

1. $\neg X$ viene convertito in CNF (vedi [4.37,pg.19]) e S e' il relativo insieme di clausole
2. Viene creato l'albero di Putnam per S.

Diremo che $\vdash X$, quando esiste un albero di Putnam per $\neg X$ tale che tutte le foglie dell'albero (i nodi finali) sono insiemi $\{[]\}$, ovvero contengono solo la clausola vuota.

Proposition 4.86. *Un albero di Putnam e' finito (ovvero, la procedura di Putnam ha termine). Una foglia di un albero di Putnam e'* $\{[]\}$ *oppure* $\emptyset$ *(l'insieme vuoto).*

Proof:
⟨1⟩1. Dim. 2.
Supponiamo che l'albero sia finito.
Se l'albero e' finito, vuol dire che la procedura di Putnam ha avuto termine e quindi non abbiamo piu' potuto applicare alcuna regola.
Supponiamo per assurdo che una foglia S' non sia ne' $\{[]\}$ ne' $\emptyset$. Allora, poiche' non e' $\emptyset$, contiene una clausola C. Poiche'$S' \neq \{[]\}$, si ha che $C \neq []$.

$$C \neq [] \Rightarrow \exists L \in C \cap \text{Lit}$$

Possiamo cosi' applicare la regola 4. ad S' rispetto alla clausola L.
Questo e' assurdo contro l'ipotesi che l'albero e' finito.
⟨1⟩2. Dim. 1.
L'albero e' finito perche' la procedura di Putnam elimina un letterale ad ogni passo. E i letterali iniziali sono finiti.

□

Theorem 4.87.

$$\vdash X \Leftrightarrow \models X$$

Proof:
⟨1⟩1. Dim. $\Rightarrow$
Se per assurdo $\not\models X$ allora esiste $v \in \mathcal{V} :\ v(\neg X) = \mathbf{t}$.
Se S e' la CNF di $\neg X$, si ha sempre $v(S) = \mathbf{t}$.
Allora, creando l'albero su S e applicando i lemmi che abbiamo visto in precedenza ([4.84,pg.63], [4.81,pg.62], [4.82,pg.62], [4.81,pg.62]) si vede che tutti gli insiemi-nodi dell'albero sono soddisfacibili.
In particolare, poiche' $\vdash X$, le foglie dell'albero saranno del tipo $\{[]\}$ e quindi, essendo anch'esse soddisfacibili, si avra':

$$v([]) = \mathbf{t}$$

assurdo contro definizinoe di v (vedi [4.35,pg.19]).
⟨1⟩2. Dim. $\Leftarrow$
Supponiamo per assurdo che $\not\vdash X$, ovvero che qualsiasi albero di putnam per $\neg X$ non e' chiuso, ovvero esiste una foglia che non e' $\{[]\}$. Quindi, per la prop [4.86,pg.65] questa foglia deve essere l'insieme $\emptyset$.
Questa foglia e' soddisfacibile e quindi per i lemmi (che sono dei $\Leftrightarrow$) anche il padre della foglia lo e'. Cosi' risalendo, si arriva fino alla radice dell'albero, cioe' S che e' la CNF di $\neg X$

$$S \text{ soddisf.} \Rightarrow \neg X \text{ soddisf.}$$

assurdo contro $\models X$.

□

Remark 4.4. La procedura di Putnam e' usata efficientemente per risolvere i problemi SAT, ovvero trovare una valutazione v che soddisfi una formula X. In particolare, un ramo di un albero di putnam di X che termina in $\emptyset$ si puo' convertire in una valutazione che soddisfa X: scegliere S_L equivale a porre $L = \mathbf{t}$, scegliere $S_{\overline{L}}$ equivale a porre $L = \mathbf{f}$.

5 Logica del I ordine

Definition 5.1. L'universo $\mathcal{U}$ di un linguaggio della logica del I ordine e' composto da tutti gli elementi della logica logica proposizionale (variabili, connettivi, punteggiatura (vedi [4,pg.7])) con l'aggiunta di

1. $\mathcal{F}$, un insieme finito o numerabile di simboli, che verranno interpretati in seguito come funzioni.
   $\hat{\cdot}$, una mappa che associa ad ogni $f \in \mathcal{F}$ un numero $\hat{f} \in \mathbb{N}$, che verra' interpretato come l'arieta' della funzione f.
   L'insieme
   $$\mathcal{F}_0 \stackrel{\text{def}}{=} \left\{ f \in \mathcal{F} \mid \hat{f} = 0 \right\}$$
   e' l'insieme delle *costanti*.
   In generale poniamo:
   $$\mathcal{F}_i \stackrel{\text{def}}{=} \left\{ f \in \mathcal{F} \mid \hat{f} = i \right\}$$

2. $\mathcal{R}$, un insieme finito o numerabile di simboli, che verranno interpretati in seguito come relazioni.
   Per ogni $r \in \mathcal{R}$, $\hat{r}$ rappresenta l'arieta' della relazione r.
   Poniamo:
   $$\mathcal{R}_i \stackrel{\text{def}}{=} \{ r \in \mathcal{R} \mid \hat{r} = i \}$$

3. $\{\exists, \forall\}$, i simboli dei *quantificatori universali.*

Definiamo inoltre l'insieme dei *termini*: Term e' il piu' piccolo sottoinsieme di $\mathcal{U}$ tale che:

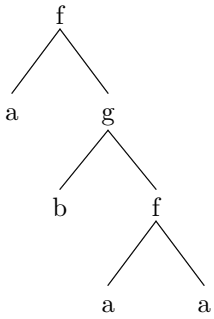
$$\text{Var} \cup \mathcal{F}_0 \subseteq \text{Term}$$
$$t_1, \ldots, t_n \in \text{Term}, \quad f \in \mathcal{F}_n \Rightarrow \underline{f(t_1, \ldots, t_n)} \in \text{Term}$$

Nota: $\underline{f(t_1, \ldots, t_n)}$ e' solo una stringa, ovvero una sequenza di simboli.
Un termine si dira' chiuso quando non contiene variabili, ma solo costanti e simboli funzionali.
Con CTerm indichiamo l'insieme di tutti i termini chiusi.
Con $\text{Var}(t)$ indichiamo l'insieme di tutte le variabili presenti nel termine t.

Example 5.2. Siano $a, b, c \in \text{Var} \cup \mathcal{F}_0$, allora la seguente stringa

$$\underline{f(a, g(b, f(a, a)))}$$

e' un termine.
Per ogni termine possiamo costruire il relativo *albero sintattico*.



Definition 5.3. L'insieme delle formule atomiche AT e' cosi' definito:

$$\mathrm{AT} = TF \cup \left\{ \underline{R(t_1, \ldots, t_n)} \mid t_1, \ldots, t_n \in \mathrm{Term}, \;\; R \in \mathcal{R}_n \right\}$$

L'insieme delle formule Form e' il piu' piccolo insieme tale che

$$\begin{aligned}
&\mathrm{AT} \subseteq \mathrm{Form} \\
&X \in \mathrm{Form} \Rightarrow \underline{\neg X} \in \mathrm{Form} \\
&X, Y \in \mathrm{Form}, \; \circ \in \mathrm{Bin} \Rightarrow \underline{X \circ Y} \in \mathrm{Form} \\
&x \in \mathrm{Var}, \; X \in \mathrm{Form} \Rightarrow \; \underline{(\forall x) \; X} \in \mathrm{Form} \\
&x \in \mathrm{Var}, \; X \in \mathrm{Form} \Rightarrow \; \underline{(\exists x) \; X} \in \mathrm{Form}
\end{aligned}$$

Diremo che x e' una *variabile legata* (o non libera) della formula X, se le uniche occorrenze di x in X sono in sottoformule del tipo $(\exists x)Y$, $(\forall x)Y$, con x presente in Y. Altrimenti, la variabile x sara' detta *libera*.

Con $\mathrm{Free}(X)$ indichiamo l'insieme di tutte le variabili libere di X, e infine poniamo $\mathrm{Bound}(X) = \mathrm{Var}(X) \backslash \mathrm{Free}(X)$.

Diremo che X e' una formula chiusa quando $\mathrm{Free}(X) = \emptyset$. Con CForm indichiamo l'insieme di tutte le formule chiuse. Le formule chiuse vengono anche chiamate *enunciati*.

Example 5.4. Supponiamo che R sia il simbolo della relazione di uguaglianza e che $t_1, t_2 \in \mathrm{Term}$. Allora, la seguente e' una formula atomica:

$$R(t_1, t_2)$$

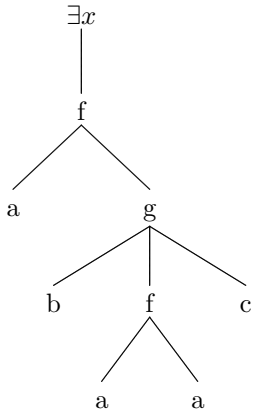
od anche:

$$\underline{= (t_1, t_2)}$$

o scritta in notazione usuale:

$$\underline{t_1 = t_2}$$

Anche per le formule possiamo costruire il relativo albero sintattico: $(\exists x) f(a, g(b, f(a, a), c))$ diventa



Definition 5.5. Una qualsiasi funzione $\sigma : \text{Var} \longrightarrow \text{Term}$ e' chiamata *sostituzione* di variabili. Utilizzeremo la seguente notazione:

$$x\sigma \stackrel{\text{def}}{=} \sigma(x)$$

Una sostituzione di termini $\sigma : \text{Term} \longrightarrow \text{Term}$ e' l'estensione di qualche sostituzione di variabili $\sigma : \text{Var} \longrightarrow \text{Term}$:

$$\begin{aligned}
&c\sigma = c \;\; \forall c \in \mathcal{F}_0 \\
&f(t_1, \ldots, t_n)\sigma = f(t_1\sigma, \ldots, t_n\sigma) \;\; \forall f \in \mathcal{F}_n, \;\; t_i \in \text{Term}
\end{aligned}$$

Una sostituzione di formule $\sigma : \text{Form} \longrightarrow \text{Form}$ e' l'estensione di qualche sostituzione di termini $\sigma : \text{Term} \longrightarrow \text{Term}$:

$$\begin{aligned}
&c\sigma = c \;\; \forall c \in \mathcal{F}_0 \\
&f(t_1, \ldots, t_n)\sigma = f(t_1\sigma, \ldots, t_n\sigma) \;\; \forall f \in \mathcal{F}_n, \;\; t_i \in \text{Term} \\
&\mathbf{t}\sigma = \mathbf{t}, \;\; \mathbf{f}\sigma = \mathbf{f} \\
&\underline{P(t_1, \ldots, t_n)} \in \text{AT} \Rightarrow \underline{P(t_1, \ldots, t_n)}\sigma = \underline{P(t_1\sigma, \ldots, t_n\sigma)} \\
&(\neg X)\sigma = \neg(X\sigma) \\
&(X \circ Y)\sigma = (X\sigma \circ Y\sigma) \\
&\underline{(\forall x)Y}\sigma = (\forall x)(Y\sigma_x) \\
&\underline{(\exists x)Y}\sigma = (\exists x)(Y\sigma_x) \\
&\text{dove } y\sigma_x = \begin{cases} y & y = x \\ y\sigma & y \neq x \end{cases}
\end{aligned}$$

Definiamo la composizione tra sostituzioni. Poiche' siamo interessati a cambiare le singole variabili di una formula, piuttosto che a cambiare una formula con un'altra, la composizione si basera' sulle variabili: $\sigma_1\sigma_2$ e' l'estensione della seguente sostituzione di variabili τ:

$$x\tau = (x\sigma_1)\sigma_2 \;\; \forall x \in \text{Var}$$

Sia $\sigma : S \longrightarrow S$ un qualsiasi tipo di sostituzione, con $\text{Var} \subseteq S$, poniamo:

$$\text{supp}(\sigma) \stackrel{\text{def}}{=} \{x \in \text{Var} \mid x \neq x\sigma\}$$

$\mathrm{supp}(\sigma)$ e' il *supporto* di σ.
Le sostituzioni che considereremo avranno tutte supporto finito. La notazione

$$\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$$

indichera' che σ e' l'estensione della seguente sostituzione di variabili τ:

$$x_i\tau = t_i \ \ \forall i = 1, \ldots, m$$
$$\mathrm{supp}(\sigma) = \mathrm{supp}(\tau) = \{x_1, \ldots, x_m\}$$

Proposition 5.6. *Siano* $\sigma_1, \sigma_2, \sigma_3 : \mathrm{Term} \longrightarrow \mathrm{Term}$ *tre sostituzioni di termini, e* $t \in \mathrm{Term}$*, allora*

0. $t\sigma_1 \in \mathrm{Term} \ \ \forall t \in \mathrm{Term}$
1. $x\sigma_1 = x\sigma_2 \ \ \forall x \in \mathrm{Var}(t) \Rightarrow t\sigma_1 = t\sigma_2$
2. $(t\sigma_1)\sigma_2 = t(\sigma_1\sigma_2) \ \ \forall t \in \mathrm{Term}$
3. $(\sigma_1\sigma_2)\sigma_3 = \sigma_1(\sigma_2\sigma_3)$

Siano $\tau_1, \tau_2, \tau_3 : \mathrm{Form} \longrightarrow \mathrm{Form}$ *tre sostituzioni di formule, allora*

4. $\exists X \in \mathrm{Form} : \ \ (X\tau_1)\tau_2 \neq X(\tau_1\tau_2)$
5. $(\tau_1\tau_2)\tau_3 = \tau_1(\tau_2\tau_3)$

La proprieta' 4. *mette in evidenza la differenza tra composizione di funzioni e composizione di sostituzioni.*

Proof: Per la 0., 1. e la 2., si procede per induzione strutturale sull'insieme dei termini Term.
La 4.: consideriamo questo controesempio:

$$X = \underline{(\forall y)R(x,y)}$$
$$\tau_1 = \{x \mapsto y\}$$
$$\tau_2 = \{y \mapsto c\}$$
$$\tau_1\tau_2 = \{x \mapsto c, y \mapsto c\}$$
$$X\tau_1 = \underline{(\forall y)R(y,y)}$$
$$(X\tau_1)\tau_2 = \underline{(\forall y)R(y,y)}\tau_2 = \underline{(\forall y)R(y,y)}$$
$$X(\tau_1\tau_2) = \underline{(\forall y)R(c,y)}$$
$$X(\tau_1\tau_2) \neq (X\tau_1)\tau_2$$

La 3.,5.: poiche' la composizione di sostituzioni e' una estensione di una sostituzione di varabili, basta verificare che

$$x(\sigma_1\sigma_2)\sigma_3 = x\sigma_1(\sigma_2\sigma_3) \ \ \forall x \in \mathrm{Var}$$

quindi, per definizione:

$$x(\sigma_1\sigma_2)\sigma_3 = (x(\sigma_1\sigma_2))\sigma_3 \underbrace{=}_{2.} ((x\sigma_1)\sigma_2)\sigma_3$$
$$x\sigma_1(\sigma_2\sigma_3) = (x\sigma_1)(\sigma_2\sigma_3) \underbrace{=}_{2.} ((x\sigma_1)\sigma_2)\sigma_3$$

□

Proposition 5.7. *Siano* $\sigma_1, \sigma_2 : \mathrm{Term} \longrightarrow \mathrm{Term}$ *due sostituzioni di termini e* $t \in \mathrm{Term}$*, allora*

$$t\sigma_1 = t\sigma_2 \Rightarrow x\sigma_1 = x\sigma_2 \ \ \forall x \in \mathrm{Var}(t)$$

Proof: Per induzione strutturale su Term.
Vediamo direttamente il passo induttivo:
$f(t_1,\dots,t_n)\sigma_1 = f(t_1,\dots,t_n)\sigma_2 \Leftrightarrow f(t_1\sigma_1,\dots,t_n\sigma_1) = f(t_1\sigma_2,\dots,t_n\sigma_2) \Leftrightarrow t_i\sigma_1 = t_i\sigma_2 \ \ \forall i \ \ (1)$
$x \in \mathrm{Var}(\underline{f(t_1,\dots,t_n)}) \Rightarrow \exists i : \ x \in \mathrm{Var}(t_i) \ \ (1.1)$
$(1),(1.1) \underbrace{\Rightarrow}_{\text{Hp induttiva}} x\sigma_1 = x\sigma_2$

□

Definition 5.8. Sia $\sigma : \mathrm{Form} \longrightarrow \mathrm{Form}$ una sostituzione di formule.
Definiamo induttivamente l'insieme $\mathrm{FreeF}(\sigma)$ delle formule che rendono σ una *sostituzione libera*:

$$\mathrm{AT} \subseteq \mathrm{FreeF}(\sigma)$$
$$\neg X \in \mathrm{Form}, \ \ X \in \mathrm{FreeF}(\sigma) \Rightarrow \neg X \in \mathrm{FreeF}(\sigma)$$
$$X \circ Y \in \mathrm{Form}, \ \ X, Y \in \mathrm{FreeF}(\sigma) \Rightarrow X \circ Y \in \mathrm{FreeF}(\sigma)$$
$$\begin{cases} Y \in \mathrm{FreeF}(\sigma_x) \\ x \notin \mathrm{Var}(y\sigma) \ \ \forall y \in \mathrm{Free}(Y) \setminus \{x\} \end{cases} \Rightarrow \underline{(\forall x)Y}, \ \ \underline{(\exists x)Y} \in \mathrm{FreeF}(\sigma)$$

In pratica, σ e' libera per X quando σ non introduce variabili legate.

Proposition 5.9. *Sia* $X \in \mathrm{Form}$ *ed* $\sigma, \tau : \mathrm{Form} \longrightarrow \mathrm{Form}$ *due sostituzioni di formule, allora*

$$\begin{cases} X \in \mathrm{FreeF}(\sigma) \\ X\sigma \in \mathrm{FreeF}(\tau) \end{cases} \Rightarrow (X\sigma)\tau = X(\sigma\tau)$$

Proof: Procediamo per induzione strutturale sull'insieme Form.
Per la [5.6,pg.69] sappiamo che

$$(t\sigma)\tau = t(\sigma\tau) \ \ \forall t \in \mathrm{Term} \ \ (1)$$

quindi per il caso base dell'induzione:
$(P(t_1,\dots,t_n)\sigma)\tau = P(t_1\sigma,\dots,t_n\sigma)\tau = P((t_1\sigma)\tau,\dots,(t_n\sigma)\tau) \underbrace{=}_{(1)} P(t_1(\sigma\tau),\dots,t_n(\sigma\tau)) = P(t_1,\dots,t_n)(\sigma\tau)$

Supponiamo che

$$X \circ Y \in \mathrm{FreeF}(\sigma)$$
$$\underline{X \circ Y}\sigma \in \mathrm{FreeF}(\tau)$$

e dimostriamo che

$$(\underline{X \circ Y}\sigma)\tau = \underline{X \circ Y}(\sigma\tau)$$

Abbiamo:

$$X \circ Y \in \mathrm{FreeF}(\sigma) \underbrace{\Rightarrow}_{\text{def. di FreeF}} X, Y \in \mathrm{FreeF}(\sigma)$$
$$X\sigma \circ Y\sigma \in \mathrm{FreeF}(\tau) \Rightarrow X\sigma, Y\sigma \in \mathrm{FreeF}(\tau)$$
$$\underbrace{\Rightarrow}_{\text{ipotesi induttiva}} (X\sigma)\tau = X(\sigma\tau), \ (Y\sigma)\tau = Y(\sigma\tau) \ \ (2)$$

Quindi:

$$(\underline{X \circ Y}\sigma)\tau = ((X\sigma) \circ (Y\sigma))\tau = ((X\sigma)\tau \circ (Y\sigma)\tau) \underbrace{=}_{(2)} X(\sigma\tau) \circ Y(\sigma\tau) = (X \circ Y)(\sigma\tau)$$

Analogamente si dimostra il caso di $\neg X$.
Adesso, supponiamo che

$$\underline{(\forall x)Y} \in \mathrm{FreeF}(\sigma)$$
$$\underline{(\forall x)Y}\sigma \in \mathrm{FreeF}(\tau)$$

e dimostriamo che

$$((\underline{\forall x})Y\sigma)\tau = \underline{(\forall x)}Y(\sigma\tau)$$

Abbiamo:

$$\underline{(\forall x)}Y \in \mathrm{FreeF}(\sigma) \underbrace{\Rightarrow}_{\text{def di FreeF}} \begin{cases} Y \in \mathrm{FreeF}(\sigma_x) & (3.1) \\ x \notin \mathrm{Var}(y\sigma) \ \ \forall y \in \mathrm{Free}(Y)\setminus\{x\} & (3.2) \end{cases}$$

$$\underline{(\forall x)}Y\sigma = \underline{(\forall x)(Y\sigma_x)} \in \mathrm{FreeF}(\tau) \Rightarrow \begin{cases} Y\sigma_x \in \mathrm{FreeF}(\tau_x) & (4.1) \\ x \notin \mathrm{Var}(y\tau) \ \ \forall y \in \mathrm{Free}(Y\sigma_x)\setminus\{x\} & (4.2) \end{cases}$$

$$(3.1),(4.1) \underbrace{\Rightarrow}_{\text{ipotesi induttiva}} (Y\sigma_x)\tau_x = Y(\sigma_x\tau_x) \underbrace{=}_{(3.2),(4.2)} Y(\sigma\tau)_x \quad (5)$$

Quindi:

$$((\underline{\forall x})Y\sigma)\tau = ((\underline{\forall x})(Y\sigma_x))\tau = (\underline{(\forall x)((Y\sigma_x)\tau_x)}) \underbrace{=}_{(5)} (\underline{(\forall x)(Y(\sigma\tau)_x)}) = (\underline{(\forall x)Y})(\sigma\tau)$$

Analogamente si vede per $\exists x$. □

Definition 5.10. Siano $t_1, t_2 \in \mathrm{Term}$ e sia $\sigma : \mathrm{Term} \longrightarrow \mathrm{Term}$ una sostituzione di termini.

$$\sigma \text{ e' un unificatore di } t_1, t_2 \overset{\text{def}}{\Leftrightarrow} t_1\sigma = t_2\sigma$$

Ad esempio, $\sigma = \{x \mapsto f(a,b)\}$ e' un unificatore dei termini $h(x)$, $h(f(a,b))$.
Poniamo: .

$$\Sigma \overset{\text{def}}{=} \{\sigma \mid \sigma : \mathrm{Term} \longrightarrow \mathrm{Term} \text{ sostituzione di termini}\}$$

$$\sigma\Sigma \overset{\text{def}}{=} \{\sigma\tau \mid \tau \in S\}$$

$$U(t_1,t_2) \overset{\text{def}}{=} \{\sigma \in \Sigma \mid \sigma \text{ unificatore di } t_1, t_2\}$$

$$\sigma \in \Sigma \text{ e' una } \textit{ridenominazione} \text{ su } V \overset{\text{def}}{\Leftrightarrow} \begin{cases} V \subseteq \mathrm{Var} \\ x\sigma \in \mathrm{Var} \ \ \forall x \in V \\ \sigma_{|V} \text{ e' iniettiva} \end{cases}$$

Sia $\sigma \in \Sigma$, poniamo:

$$\mathrm{range}(\sigma) \overset{\text{def}}{=} \bigcup_{x \in \mathrm{Var}} \mathrm{Var}(x\sigma)$$

Proposition 5.11.

$$\sigma \in U(t_1,t_2) \Rightarrow \sigma\Sigma \subseteq U(t_1,t_2)$$

Proof:

$$\sigma \in U(t_1,t_2) \Leftrightarrow t_1\sigma = t_2\sigma \Rightarrow (t_1\sigma)\tau = (t_2\sigma)\tau \underbrace{\Leftrightarrow}_{\text{[5.6,pg.69]}} t_1(\sigma\tau) = t_2(\sigma\tau) \Rightarrow \sigma\tau \in U(t_1,t_2)$$

□

Definition 5.12. Siano $t_1, t_2 \in \mathrm{Term}$ e sia $\sigma \in U(t_1,t_2)$

σ e' un unificatore massimale $t_1, t_2 \overset{\text{def}}{\Leftrightarrow} \sigma\Sigma = U(t_1,t_2) \Leftrightarrow \forall \rho \in U(t_1,t_2) \ \exists \tau \in \Sigma : \ \ \rho = \sigma\tau$

$U_{\max}(t_1,t_2) \overset{\text{def}}{=} \{\sigma \mid \sigma \text{ unificatore massimale di } t_1, t_2\}$

Proposition 5.13. *Siano* $t_1, t_2 \in \text{Term}$ *e sia* $\sigma, \tau \in U_{\max}(t_1, t_2)$*, allora*

$$\exists \rho \in \Sigma \textit{ ridenominazione su } \text{range}(\sigma) \textit{ tale che}$$
$$\tau = \sigma\rho$$

Proof: Dalla definizione, si ha

$$\sigma\Sigma = U(t_1, t_2) = \tau\Sigma \Rightarrow \begin{cases} \exists \rho_1 \in \Sigma : \quad \tau = \sigma\rho_1 \quad (1) \\ \exists \rho_2 \in \Sigma : \quad \sigma = \tau\rho_2 \end{cases} \Rightarrow \sigma = \sigma\rho_1\rho_2 \quad (2)$$

Per la (1), e' sufficiente dimostrare che ρ_1 e' una ridenominazione su range(σ).

$\langle 1 \rangle 1$. $x\rho_1 \in \text{Var} \quad \forall x \in \text{range}(\sigma)$

$$x \in \text{range}(\sigma) \Rightarrow \exists y : \quad x \in \text{Var}(y\sigma)$$
$$(2) \Rightarrow y\sigma = y\sigma\rho_1\rho_2 = ((y\sigma)\rho_1)\rho_2 \underbrace{=}_{[5.6,\text{pg.}69]} (y\sigma)(\rho_1\rho_2) \Leftrightarrow (y\sigma)\text{id} = (y\sigma)(\rho_1\rho_2) \quad (3)$$
$$\underbrace{\Rightarrow}_{[5.7,\text{pg.}69]} x\text{id} = x(\rho_1\rho_2) \Leftrightarrow x = (x\rho_1)\rho_2 \quad (4)$$

x e' una variabile del termine $y\sigma$. Se per assurdo $x\rho_1$ e' del tipo $f(t_1, \ldots, t_n)$, allora vuol dire che in $(y\sigma)\rho_1$ la variabile x e' stata sostituita con $f(\ldots,)$, quindi $(y\sigma)\rho_1$ ha una lunghezza maggiore di $y\sigma$:

$$l(y\sigma) < l((y\sigma)\rho_1) \leq l(((y\sigma)\rho_1)\rho_2) \underbrace{=}_{(3)} l(y\sigma)$$
$$l(y\sigma) < l(y\sigma)$$

assurdo.

Se per assurdo $x\rho_1$ e' una costante c, allora

$$(4) \Rightarrow x = c\rho_2 = c \Rightarrow x \text{ costante}$$

assurdo contro $x \in \text{range}(\sigma) \subseteq \text{Var}$.

Quindi, si deve avere

$$x\rho_1 \in \text{Var}$$

$\langle 1 \rangle 2$. Presi $x, y \in \text{range}(\sigma)$, si ha $x\rho_1 = y\rho_1 \Rightarrow x = y$

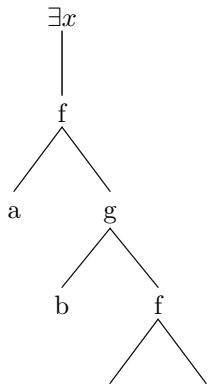
$$x \underbrace{=}_{(4)} (x\rho_1)\rho_2 \underbrace{=}_{x\rho = y\rho_1} (y\rho_1)\rho_2 \underbrace{=}_{(4)} y \Rightarrow x = y$$

In conclusione, ρ_1 e' una ridenominazione su range(σ). □

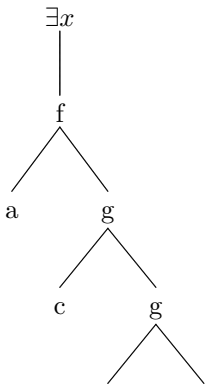
Definition 5.14. Siano $t_1, t_2 \in \text{Term}$ e siano T_1, T_2 i relativi alberi sintattici di t_1, t_2 (vedi [5.2,pg.66]). Siano d_1, d_2 due sottoalberi di T_1, T_2 (che possiamo pensare come due sottotermini di t_1, t_2). Sia $r_1 = (x_1, \ldots, x_m)$ il ramo di T_1, dove x_m e' la radice di d_1 e x_1 e' la radice di T_1 . Analogamente, sia $r_2 = (x'_1, \ldots, x'_n)$ il ramo che parte dalla radice x'_1 di T_2 e finisce in x'_n, radice di d_2.

$$(d_1, d_2) \text{ e' una } \textit{coppia di disaccordo} \text{ per } t \overset{\text{def}}{\Leftrightarrow} \begin{cases} x_n \neq x_m \\ (x_1, \ldots, x_{n-1}) = (x'_1, \ldots, x'_{m-1}) \end{cases}$$

Example 5.15.



$t_1 = (\exists x) f(a, g(b, f(a, a)))$,



$t_2 = (\exists x) f(a, g(c, g(b, c)))$,

Una coppia di disaccordo e' data dai due sottotermini $d_1 = f(a, a),\ d_2 = g(b, c)$.
Un'altra coppia e' $d_1 = b,\ d_2 = c$.

Theorem 5.16. *Algoritmo di unificazione.*
Siano $t_1, t_2 \in \text{Term}$. *Il seguente algoritmo restituisce* $\emptyset$ *se non e' possibile unificare* t_1, t_2, *altrimenti restituisce un* $\sigma \in U_{\max}(t_1, t_2)$*:*

$\sigma := \text{id}$
While $t_1\sigma \neq t_2\sigma$:
  Sia (d_1, d_2) *una coppia di disaccordo per* t_1, t_2
  Se $\{d_1, d_2\} \cap \text{Var} = \emptyset$: (1)
    $\sigma := \emptyset$
    break
  Sia $x \in \{d_1, d_2\} \cap \text{Var},\ t \in \{d_1, d_2\} \setminus \{x\}$
  Se $x \in \text{Var}(t)$: (2)
    $\sigma := \emptyset$
    break
  $\sigma := \sigma\{x \mapsto t\}$ *e' la composizione di* σ *con* $\{x \mapsto t\}$
σ

Proof: Supponiamo le condizioni (1),(2) non si verifichino, ovvero che l'algoritmo non restituisca $\emptyset$. Dimostriamo che l'algoritmo termina e che $\sigma \in U_{\max}(t_1, t_2)$.
⟨1⟩1. L'algoritmo termina
  Sia $V(\sigma) = \text{Var}(t_1\sigma) \cup \text{Var}(t_2\sigma)$.
  Ad ogni ciclo sostituiamo la variabile x con il termine t. Ma poiche' la (2) non si verifica, $x \notin \text{Var}(t)$. Quindi, ad ogni ciclo $V(\sigma)$ perde un elemento.
  Poiche' $V(\sigma)$ era inizialmente finito, l'algoritmo dovra' prima o poi terminare: ad un certo punto la condizione di terminazione del while ($t_1\sigma = t_2\sigma$) sara' verificata.
⟨1⟩2. $\sigma \in U_{\max}(t_1, t_2)$
  Sia $\tau \in U(t_1, t_2)$. Dimostriamo che $\exists \rho \in \Sigma :\ \tau = \sigma\rho$. In particolare, dimostriamo che $\tau = \sigma\tau$.

  ⟨2⟩1. $\tau = \sigma\tau$ e' un invariante del while
    Per induzione, caso base:
      $\sigma = \text{id}$:
        $\tau = \text{id}\tau = \tau$

Supponiamo $\tau = \sigma\tau$ e dimostriamo che $\tau = (\sigma\{x \mapsto t\})\tau$.

$$\tau \in U(t_1, t_2) \Rightarrow t_1\tau = t_2\tau \Rightarrow d_1\tau = d_2\tau \quad (3)$$

$$\text{Per ipotesi induttiva } \tau = \sigma\tau \quad (4)$$

Dimostriamo che $\{x \mapsto t\}\tau = \tau$. Per la [5.6,pg.69] basta dimostrare che

$$y\{x \mapsto t\}\tau = y\tau \quad \forall y \in \text{Var}$$

quindi,

$$y = x \Rightarrow y\{x \mapsto t\} = t \Rightarrow (y\{x \mapsto t\})\tau = t\tau \underbrace{=}_{(3),\ \{x,t\}=\{d_1,d_2\}} x\tau \underbrace{=}_{y=x} y\tau$$

$$y \neq x \Rightarrow y\{x \mapsto t\} = y \Rightarrow (y\{x \mapsto t\})\tau = y\tau$$

Infine,

$$(\sigma\{x \mapsto t\})\tau = \sigma(\{x \mapsto t\}\tau) = \sigma\tau \underbrace{=}_{(4)} \tau$$

Abbiamo cosi' dimostrato che

$$\tau = \sigma\tau$$

vale ad ogni ciclo.

□

Proposition 5.17.

$$U_{\max}(t_1, t_2) \neq \emptyset \Leftrightarrow \textit{L'algoritmo di unificazione restituisce un } \sigma \in U_{\max}(t_1, t_2)$$

Proof:
⟨1⟩1. Dim. ⇐

$$\sigma \in U_{\max}(t_1, t_2) \Rightarrow U_{\max}(t_1, t_2) \neq \emptyset$$

⟨1⟩2. Dim. ⇒
Se le condizioni (1), (2) dell'algoritmo [5.16,pg.73] non si verificano, allora, come gia' visto, si ha la tesi.
Se le (1), (2) si verificano, e' facile vedere che $U(t_1, t_2) = \emptyset$. Assurdo.

□

Proposition 5.18. *Unificazione concorrente*
Siano dati i termini $t_1, u_1, t_2, u_2, \ldots, t_n, u_n \in \text{Term}$. *Per trovare un elemento* σ *di*

$$\bigcap_{i=1,\ldots,n} U(t_i, u_i)$$

e' sufficiente applicare l'algoritmo [5.16,pg.73] ai due termini

$$f(t_1, \ldots, t_n), \quad f(u_1, \ldots, u_n)$$

dove $f \in \mathcal{F}$ *e' un qualsiasi simbolo funzionale, infatti,*

$$\bigcap_{i=1,\ldots,n} U(t_i, u_i) = U\left(\underline{f(t_1, \ldots, t_n)}, \underline{f(u_1, \ldots, u_n)}\right)$$

Proof: Basta osservare che

$$f(t_1, \ldots, t_n)\sigma = f(t_1\sigma, \ldots, t_n\sigma)$$

$$f(t_1\sigma, \ldots, t_n\sigma) = f(u_1\sigma, \ldots, u_n\sigma) \Leftrightarrow t_i\sigma = u_i\sigma \quad \forall i$$

□

Definition 5.19. Unificazione multipla

$$U(t_1,\dots,t_n)\stackrel{\text{def}}{=}\bigcap_{i\neq j}U(t_i,t_j)$$

Proposition 5.20. *Per trovare un elemento $\sigma\in U(t_1,\dots,t_n)$ e' sufficiente applicare l'algoritmo [5.16,pg.73] ai termini:*

$$f(t_1,\dots,t_{n-1}),\quad f(t_2,\dots,t_n)$$

Proof: Per la prop [5.18,pg.74] si ha

$$\bigcap_{i=1,\dots,n-1}U(t_i,t_{i+1})=U\left(\underline{f(t_1,\dots,t_{n-1})},\underline{f(t_2,\dots,t_n)}\right)$$

Inoltre,

$$\sigma\in\bigcap_{i=1,\dots,n-1}U(t_i,t_{i+1})\Leftrightarrow t_1\sigma=t_2\sigma,\ \ t_2\sigma=t_3\sigma,\dots\Leftrightarrow t_1\sigma=t_2\sigma=t_3\sigma=\cdots=t_n\sigma\Leftrightarrow\sigma\in\bigcap_{i\neq j}U(t_i,t_j)$$

□

5.1 Semantica

Nella logica proposizionale era sufficiente usare le valutazioni per dare un "significato" alle formule. Vediamo adesso come dare senso alle formule della logica del I ordine.

Definition 5.21. Un modello di un linguaggio $L=(\mathcal{F},\mathcal{R})$ della logica del I ordine e' una coppia $M=(D,I)$, dove D e' un insieme non vuoto, I e' una mappa tale che

$$\begin{aligned}
&I:\mathcal{F}\cup\mathcal{R}\longrightarrow D\cup\{\text{funzioni su D}\}\cup[\text{relazioni su D}]\\
&c^I\in D\ \ \forall c\in\mathcal{F}_0\\
&f\in\mathcal{F}_i\Rightarrow f^I\text{ e' una funzione }f^I:D^i\longrightarrow D\\
&R\in\mathcal{R}_i\Rightarrow R^I\subseteq D^i\text{ e' una relazione su }D
\end{aligned}$$

I e' chiamata *interpretazione* del linguaggio L.

Una qualsiasi funzione $A:\text{Var}\longrightarrow D$ e' chiamata *assegnamento* di variabili.
Sia $x\in\text{Var}$, definiamo l'insieme degli x-varianti di A:

$$A_x=\left\{B:\text{Var}\longrightarrow D\mid y^B=y^A\ \ \forall y\in\text{Var}\backslash\{x\}\right\}$$

In altre parole, un elemento di A_x e' una assegnamento che differisce da A in al piu' x.

Sia I una interpretazione e A un assegnamento, definiamo la valutazione (I,A) di un termine:

$$\begin{aligned}
&(I,A):\text{Term}\longrightarrow D\\
&c^{I,A}=c^I\ \ \forall c\in\mathcal{F}_0\\
&x^{I,A}=x^A\ \ \forall x\in\text{Var}\\
&(f(t_1,\dots,t_n))^{I,A}=f^I(t_1^{I,A},\dots,t_n^{I,A})\ \ \forall f\in\mathcal{F}_n
\end{aligned}$$

In altre parole, la valutazione "calcola" il valore dei termini.

Utilizzando le valutazioni dei termini, possiamo fornire un valore di verita' ad ogni formula $X \in \text{Form}$:

$$
\begin{aligned}
&v_{I,A} : \text{Form} \longrightarrow TF \\
&v_{I,A}(\mathbf{t}) = \mathbf{t}, \;\; v_{I,A}(\mathbf{f}) = \mathbf{f} \\
&v_{I,A}\left(\underline{R(t_1, \ldots, t_n)}\right) = \mathbf{t} \Leftrightarrow (t_1^{I,A}, \ldots, t_n^{I,A}) \in R^I, \quad \text{dove } R \in \mathcal{R}_n,\ t_1, \ldots, t_n \in \text{Term} \\
&v_{I,A}(\underline{\neg X}) = \neg v_{I,A}(X) \\
&v_{I,A}(\underline{X \circ Y}) = v_{I,A}(X) \circ v_{I,A}(Y) \\
&v_{I,A}(\underline{(\forall x)Y}) = \mathbf{t} \Leftrightarrow v_{I,B}(Y) = \mathbf{t} \;\; \forall B \in A_x \\
&v_{I,A}(\underline{(\exists x)Y}) = \mathbf{t} \Leftrightarrow \exists B \in A_x : \; v_{I,B}(Y) = \mathbf{t}
\end{aligned}
$$

Sia $M = (D, I)$ un modello del linguaggio L, sia $X \in \text{Form}$ e sia $S \subseteq \text{Form}$

X e' soddisfacibile in $M \overset{\text{def}}{\Leftrightarrow} M$ soddisfa $X \overset{\text{def}}{\Leftrightarrow} \exists A : \text{Var} \longrightarrow D$ assegnamento t.c. $v_{I,A}(X) = \mathbf{t}$

S e' soddisfacibile $\overset{\text{def}}{\Leftrightarrow} \exists A : \text{Var} \longrightarrow D$ assegnamento t.c. $v_{I,A}(Y) = \mathbf{t} \;\; \forall Y \in S$

X e' vera in $M \overset{\text{def}}{\Leftrightarrow} v_{I,A}(X) = \mathbf{t} \;\; \forall A : \text{Var} \longrightarrow D$ assegnamento

X e' valida in L $\overset{\text{def}}{\Leftrightarrow} X$ e' vera in ogni modello M di L

Poniamo le seguenti notazioni:

$$
\begin{aligned}
&M, A \models X \overset{\text{def}}{\Leftrightarrow} v_{I,A}(X) = \mathbf{t} \\
&M \models X \overset{\text{def}}{\Leftrightarrow} X \text{ e' vera in } M \\
&\models X \overset{\text{def}}{\Leftrightarrow} X \text{ e' valida} \\
&S \text{ e' soddisfacibile } \overset{\text{def}}{\Leftrightarrow} \exists M \text{ modello t.c. } S \text{ e' soddisfatto da } M
\end{aligned}
$$

Infine, poniamo

$$S \models X \overset{\text{def}}{\Leftrightarrow} (M \models S \Rightarrow M \models X)$$

Proposition 5.22. *Siano $Y, Z \in \text{Form}$, allora*

$$\models \underline{Y \Leftrightarrow Z} \Rightarrow \; (\models Y \Leftrightarrow \; \models Z)$$

Proposition 5.23. *Sia $Y \in \text{Form}$, si ha*

$$M \models \underline{(\forall x)Y} \Leftrightarrow M \models Y$$

Proof:
$M \models \underline{(\forall x)Y} \Rightarrow \forall A$ assegnamento $v_{I,A}(\underline{(\forall x)Y}) = \mathbf{t} \Leftrightarrow \forall A$ assegnamento $\forall B \in A_x v_{I,B}(Y) = \mathbf{t} \Leftrightarrow$
$\Leftrightarrow \forall B$ assegnamento $v_{I,B}(Y) = \mathbf{t} \Leftrightarrow M \models Y$

□

5.2 Termini chiusi

Proposition 5.24. *Sia A un assegnamento, σ una sostituzione qualsiasi di termini. Si ha*

$$t \in \mathrm{CTerm} \Rightarrow t^A = t\sigma = t$$

Ovvero i termini chiusi non vengono modificati dagli assegnamenti e dalle sostituzioni.

Proof: Intuitivamente: t, essendo chiuso non ha variabili, quindi A e σ non possono modificarlo. Si dimostra per induzione su Term. □

Lemma 5.25. *Sia $M = (D, I)$ un dominio, $x \in \mathrm{Var}$ e siano A, A' due assegnamenti tali che $x^A = x^{A'}$*

1. $t \in \mathrm{Term},\ \ \mathrm{Var}(t) \subseteq \{x\} \Rightarrow t^{I,A} = t^{I,A'}$
2. $Y \in \mathrm{Form},\ \ \mathrm{Free}(Y) \subseteq \{x\} \Rightarrow v_{I,A}(Y) = v_{I,A'}(Y)$

Proof: Si procede per induzione su Term e poi su Form.

$$t = c \in \mathcal{F}_0 \Rightarrow c^{I,A} = c^I = c^{I,A'}$$

$$t = y \in \mathrm{Var} \underbrace{\Rightarrow}_{\mathrm{Var}(t)\subseteq\{x\}} \Rightarrow t = x \Rightarrow x^{I,A} = x^A = x^{A'} = x^{I,A'}$$

$$t = f(t_1, \ldots, t_n) \in \mathrm{Term} \Rightarrow (f(t_1, \ldots, t_n))^{I,A} = f^I(t_i^{I,A} \mid i = 1, \ldots, n) =$$

$$\underbrace{=}_{\text{Hp induttiva}} f^I(t_i^{I,A'} \mid i = 1, \ldots, n) = (f(t_1, \ldots, t_n))^{I,A'}$$

Quindi per i termini funziona tutto. Passiamo alle formule

$$Y = \mathbf{t} \text{ oppure } Y = \mathbf{f} \Rightarrow v_{I,A}(Y) = Y = v_{I,A'}(Y)$$

$$Y = R(t_1, \ldots, t_n) \in \mathrm{AT}$$

$$\Rightarrow v_{I,A}(R(t_1, \ldots, t_n)) = \mathbf{t} \Leftrightarrow (t_1^{I,A}, \ldots, t_n^{I,A}) \in R^I$$

$$\mathrm{Free}(Y) \subseteq \{x\} \Rightarrow \mathrm{Var}(t_i) \subseteq \{x\} \ \ \forall i = 1, \ldots, n \underbrace{\Rightarrow}_{1.} t_i^{I,A} = t_i^{I,A'}$$

$$\Leftrightarrow (t_1^{I,A'}, \ldots, t_n^{I,A'}) \in R^I \Leftrightarrow v_{I,A'}(R(t_1, \ldots, t_n)) = \mathbf{t}$$

Per il passo induttivo dimostriamo solo la parte che coinvolge i quantificatori.

$$Y = \underline{(\forall x)Z} \in \mathrm{Form} \qquad v_{I,A}(\underline{(\forall x)Z}) = \mathbf{t} \Leftrightarrow \forall B \in A_x \ \ v_{I,B}(Z) = \mathbf{t}$$

$$\mathrm{Free}(Y) \subseteq \{x\} \Rightarrow \mathrm{Free}(Z) \subseteq \{x\} \underbrace{\Rightarrow}_{\text{Hp indutt.}} v_{I,B}(Z) = v_{I,B'}(Z)$$

$$\text{dove } B' \in A'_x : \ \ x^B = x^{B'}$$

$$\Leftrightarrow \forall B' \in A'_x \ \ v_{I,B'}(Z) = \mathbf{t} \Leftrightarrow v_{I,A'}(\underline{(\forall x)Z})$$

Analogamente per $(\exists x)$ □

Proposition 5.26. *Sia $M = (D, I)$ un dominio,*

$$t \in \mathrm{CTerm} \Rightarrow t^{I,A} = t^{I,A'} \ \ \forall A, A' \text{ assegnamenti}$$

$$Y \in \mathrm{CForm} \Rightarrow v_{I,A}(Y) = v_{I,A'}(Y) \ \ \forall A, A'$$

Per questo motivo, useremo la seguente notazione:

$$t^I \overset{def}{=} t^{I,A}$$

$$v_I(Y) \overset{def}{=} v_{I,A}(Y)$$

dove A e' una qualsiasi assegnamento.

Proof: Si procede per induzione su Term e poi su Form. Dimostriamo solo il passo che coinvolge i quantificatori.

$v_{I,A}(\underline{(\forall x)Z}) = \mathbf{t} \Leftrightarrow \forall B \in A_x \;\; v_{I,B}(Z) = \mathbf{t}$

$\underline{(\forall x)Z} \in \text{CForm} \Rightarrow \text{Free}(Z) \subseteq \{x\} \underbrace{\Rightarrow}_{[5.25,\text{pg.}77]} v_{I,B}(Z) = v_{I,B'}(Z),$ con $B' \in A'_x : \; x^B = x^{B'}$

$\Leftrightarrow \forall B' \in A'_x \;\; v_{I,B'}(Z) = \mathbf{t} \Leftrightarrow v_{I,A'}(\underline{(\forall x)Z}) = \mathbf{t}$

Analogamente si vede per $(\exists x)$:

$v_{I,A}(\underline{(\exists x)Z}) = \mathbf{t} \Leftrightarrow \exists B \in A_x : \;\; v_{I,B}(Z) = \mathbf{t}$

$\underline{(\exists x)Z} \in \text{CForm} \Rightarrow \text{Free}(Z) \subseteq \{x\} \underbrace{\Rightarrow}_{[5.25,\text{pg.}77]} v_{I,B}(Z) = v_{I,B'}(Z),$ con B' un qualsiasi elemento di $A'_x : \; x^B = x^{B'}$

$\Rightarrow v_{I,B'}(Z) = \mathbf{t} \Rightarrow v_{I,A'}(\underline{(\exists x)Z}) = \mathbf{t}$

Allo stesso modo si vede che

$$v_{I,A'}(\underline{(\exists x)Z}) = \mathbf{t} \Rightarrow v_{I,A}(\underline{(\exists x)Z}) = \mathbf{t}$$

□

Proposition 5.27. *Sia $Z \in \text{CForm}$, e $M = (I, D)$ un modello, allora*

$$v_I(Z) = \mathbf{t} \Leftrightarrow M \models Z \quad \textit{ovvero, } Z \textit{ e' vera in } M$$

Proof:

$$\mathbf{t} = v_I(Z) \underbrace{=}_{[5.26,\text{pg.}77]} v_{I,A}(Z) \;\; \forall A \text{ assegnamento} \Rightarrow M \models Z$$

□

Proposition 5.28. *Sia $Y \in \text{Form}$, $t \in \text{CTerm}$, $M = (D, I)$ un modello, A una assegnazione e $x \in \text{Var}$. Si ha*

$$x^A = t^I \Rightarrow v_{I,A}(Y) = v_{I,A}(Y\{x \mapsto t\})$$

Proof: Supponiamo $x^A = t^I$.

Si procede per induzione su Term dimostrando che

$$(s\{x \mapsto t\})^{I,A} = s^{I,A} \;\; \forall s \in \text{Term}$$

Poi si procede facilmente per induzione su Form dimostrando la tesi:

$$v_{I,A}(Y) = v_{I,A}(Y\{x \mapsto t\}) \;\; \forall Y \in \text{Form}$$

□

Definition 5.29. Sia $M = (D, I)$ un modello, diremo che M e' un *modello di Herbrand* $\Leftrightarrow$

1. $D = \text{CTerm}$
2. $t^I = t \;\; \forall t \in \text{CTerm}$

dove CTerm e' l'insieme di tutti i termini chiusi (termini privi di variabili).

Proposition 5.30. *Sia $M = (D, I)$ un modello di Herbrand, sia $A : \text{Var} \longrightarrow D$ un assegnamento, $t \in \text{Term}$, $Y \in \text{Form}$, allora*

1. *La funzione $(xA) = x^A$ e' una sostituzione*
2. $(tA)^I = t^{I,A}$
3. $v_{I,A}(Y) = v_I(YA)$

In altre parole, gli assegnamenti e le sostituzioni si equivalgono sotto l'interpretazione I del modello di Herbrand.

Proof:
⟨1⟩1. Dim 1.
Poiche' M e' di Herbrand si ha

$$A : \text{Var} \longrightarrow D = \text{CTerm} \subseteq \text{Term}$$

e quindi, per definizione, A e' anche una sostituzione
⟨1⟩2. Dim 2.
Poiche' M e' un modello di Herbrand, si ha

$$s^I = s \ \ \forall s \in \text{CTerm} \ \ (1)$$

Procediamo per induzione su Term,
Case: $t = c \in \mathcal{F}_0$

$$(cA)^I \underbrace{=}_{cA=c \text{ per la def di sostituzione estesa ai termini}} c^I = c^{I,A}$$

Case: $t = x \in \text{Var}$

$$xA \in D = \text{CTerm} \underbrace{\Rightarrow}_{(1)} (xA)^I = xA = x^A = x^{I,A}$$

Case: $t = f(t_1, \ldots, t_n)$

$$(f(t_1, \ldots, t_n)A)^I \underbrace{=}_{(1)} f(t_1A, \ldots, t_nA) \underbrace{=}_{\text{Hp induttiva}} f(t_1^{I,A}, \ldots, t_n^{I,A}) \underbrace{=}_{f \in \text{CTerm} = D,\ (1)} f^I(t_1^{I,A}, \ldots, t_n^{I,A}) =$$

$$= (f(t_1, \ldots, t_n))^{I,A}$$

⟨1⟩3. Dim 3.
Per induzione su Form,
$Y = \mathbf{t}$ oppure $Y = \mathbf{f} \Rightarrow v_{I,A}(Y) = Y = YA = v_{I,A'}(YA)$
$Y = R(t_1, \ldots, t_n) \in \text{AT}$

$$\Rightarrow v_{I,A}(R(t_1, \ldots, t_n)) = \mathbf{t} \Leftrightarrow (t_1^{I,A}, \ldots, t_n^{I,A}) \in R^I$$

$$t_i^{I,A} \underbrace{=}_{2.} (t_iA)^I$$

$$\Leftrightarrow ((t_1A)^I, \ldots, (t_nA)^I) \in R^I \Leftrightarrow v_{I,A}(R(t_1A, \ldots, t_nA)) = \mathbf{t} \Leftrightarrow v_{I,A}(R(t_1, \ldots, t_n)A) = \mathbf{t} \Leftrightarrow$$

$$\underbrace{\Leftrightarrow}_{R(t_1,\ldots,t_n)A \in \text{CTerm},\ [5.26,\text{pg.}77]} v_I(R(t_1, \ldots, t_n)A)$$

Per il passo induttivo dimostriamo solo la parte che coinvolge i quantificatori. Nota: usiamo l'induzione sul numero di costruttori della formula (vedi [3.14,pg.6]).
Sia $Z = (\forall x)Y \in \text{Form}$, dimostriamo che $v_{I,A}(Z) = v_I(ZA)$
⟨2⟩1. Dim. $v_{I,A}(Z) = \mathbf{t} \Rightarrow v_I(ZA) = \mathbf{t}$

$ZA = (\forall x) Y\sigma_x$, dove σ_x e' la seguente sostituzione: $\sigma_x(u) = \begin{cases} u & x = u \\ uA & x \neq u \end{cases}$ (1)

$\sigma_x \in A_x$, ovvero, σ_x e' un x-variante di A

$v_I(ZA) = \mathbf{t} \underbrace{\Leftrightarrow}_{\text{[5.27,pg.78]}} M \models Y\sigma_x \Leftrightarrow \forall C$ assegnamento $v_{I,C}(Y\sigma_x) = \mathbf{t}$ (2)

$v_{I,A}(Z) = \mathbf{t} \Leftrightarrow \forall B \in A_x \;\; v_{I,B}(Y) = \mathbf{t} \;\; \Leftrightarrow$

$v_{I,B}(Y) \underbrace{=}_{\text{Hp indutt.}} v_I(YB)$

$\Leftrightarrow \forall B \in A_x \;\; v_{I,B}(Y) = v_I(YB) = v_{I,C}(YB) \;\; \forall C$ assegn. $\underbrace{\Rightarrow}_{\text{per } B=\sigma_x} v_{I,C}(Y\sigma_x) = \mathbf{t} \;\; \forall C \underbrace{\Rightarrow}_{(2)} v_I(ZA) = \mathbf{t}$

$\langle 2 \rangle 2$. Dim. $v_I(ZA) = \mathbf{t} \Rightarrow v_{I,A}(Z) = \mathbf{t}$

(2) $\Rightarrow \forall C$ assegn. $v_{I,C}(Y\sigma_x) = \mathbf{t}$ (2′)

$v_{I,A}(Z) = \mathbf{t} \Leftrightarrow \forall B \in A_x \;\; v_{I,B}(Y) = \mathbf{t}$ (3)

Fissato $B \in A_x$ si ha $B = \{x \mapsto d\}\, A$ per qualche $d \in D$ vedi [5.31,pg.80]

$B = \{x \mapsto d\}\, A = \{x \mapsto d\}\, \sigma_x$ (4)

(2′) $\underbrace{\Rightarrow}_{\text{posto } C=\{x\mapsto d\}} v_{I,\{x\mapsto d\}}(Y\sigma_x) = \mathbf{t} \underbrace{\Leftrightarrow}_{\text{Hp indutt.}} v_{I,\{x\mapsto d\}\sigma_x}(Y) = \mathbf{t} \underbrace{\Leftrightarrow}_{(4)} v_{I,B}(Y) = \mathbf{t}$

Quindi la (3) e' vera.

Analogamente per $(\exists x)$,

□

Proposition 5.31. *Sia $M = (D, I)$ un modello di Herbrand e sia A un assegnamento, allora*

$$B \in A_x \Leftrightarrow \exists d \in D : \;\; B = \{x \mapsto d\}\, A$$

In particolare esiste una biezione tra A_x e D:

$$\tau : A_x \to D$$
$$B \mapsto x^B$$

la cui inversa e':

$$\tau^{-1} : D \longrightarrow A_x$$
$$d \mapsto \{x \mapsto d\}\, A$$

Proof: Sia $d \in D$. Poiche' $D = \text{CTerm}$ si ha

$$D = \text{CTerm} \underbrace{\Rightarrow}_{\text{[5.24,pg.77]}} dC = d \;\; \forall C \text{ assegnamento} \quad (1)$$

quindi

$\langle 1 \rangle 1$. Dim. $\Leftarrow$

$$x(\{x \mapsto d\}\, A) \underbrace{=}_{(1)} dA = d \Rightarrow \{x \mapsto d\}\, A \in A_x$$

$\langle 1 \rangle 2$. Dim. $\Rightarrow$

$$B \in A_x \Rightarrow x^B = d \in D, \text{ per qualche } d, \quad u^B = u^A \ \ \forall u \neq x$$
$$\underbrace{\Rightarrow}_{(1)} B = \{x \mapsto d\}\, A$$

E' facile vedere che τ e' una biezione. □

Proposition 5.32. *Sia $M = (D, I)$ un modello di Herbrand, $Y \in$ Form, allora*

1. $M \models \underline{(\forall x)Y} \Leftrightarrow \forall d \in D \ \ M \models Y\,\{x \mapsto d\}$
2. $\underline{(\exists x)Y} \in \text{CForm} \Rightarrow \Big(M \models \underline{(\exists x)Y} \Leftrightarrow \exists d \in D : \ \ M \models Y\,\{x \mapsto d\}\Big)$

Proof: Sia A un assegnamento,
⟨1⟩1. Dim. 1.
⟨2⟩1. Dim. ⇔

$$M \models \underline{(\forall x)Y} \Leftrightarrow \forall A \text{ assegn.} \quad v_{I,A}(\underline{(\forall x)Y}) = \mathbf{t} \Leftrightarrow \forall A \text{ assegn.} \quad \forall B \in A_x \ \ v_{I,B}(Y) = \mathbf{t} \Leftrightarrow$$
$$\underbrace{\Leftrightarrow}_{\text{[5.31,pg.80]}} \quad \forall A \text{ assegn.} \quad \forall d \in D \ \ v_{I,\{x\mapsto d\}A}(Y) = \mathbf{t} \ \ (1)$$
$$v_{I,\{x\mapsto d\}A}(Y) \underbrace{=}_{\text{3. di [5.30,pg.79]}} v_I(Y(\{x \mapsto d\}\, A)) \underbrace{=}_{\text{FreeF}(\{x\mapsto d\})=\text{FreeF}(A)=\text{Form, [5.9,pg.70]}} v_I((Y\,\{x \mapsto d\})A) =$$
$$\underbrace{=}_{\text{3. di [5.30,pg.79]}} v_{I,A}(Y\,\{x \mapsto d\}) \ \ (2)$$
$$(1) \underbrace{\Leftrightarrow}_{(2)} \Rightarrow \forall A \text{ assegn.} \quad \forall d \in D \ \ v_{I,A}(Y\,\{x \mapsto d\}) = \mathbf{t} \Leftrightarrow \forall d \in D \ \ \forall A \text{ assegn.} \quad v_{I,A}(Y\,\{x \mapsto d\}) = \mathbf{t} \Leftrightarrow$$
$$\Leftrightarrow \forall d \in D \ \ M \models Y\,\{x \mapsto d\}$$

⟨1⟩2. Dim. 2.
⟨2⟩1. Dim. ⇔

$$M \models \underline{(\exists x)Y} \Leftrightarrow \forall A \text{ assegn.} \quad v_{I,A}(\underline{(\exists x)Y}) = \mathbf{t} \Leftrightarrow \forall A \text{ assegn.} \quad \exists B \in A_x : \ \ v_{I,B}(Y) = \mathbf{t} \Leftrightarrow$$
$$\underbrace{\Leftrightarrow}_{\text{[5.31,pg.80]}} \quad \forall A \text{ assegn.} \quad \exists d \in D \ \ v_{I,\{x\mapsto d\}A}(Y) = \mathbf{t} \ \ (1)$$
$$v_{I,\{x\mapsto d\}A}(Y) \underbrace{=}_{\text{come prima}} v_{I,A}(Y\,\{x \mapsto d\}) \ \ (2)$$
$$(1) \underbrace{\Leftrightarrow}_{(2)} \Rightarrow \forall A \text{ assegn.} \quad \exists d \in D \ \ v_{I,A}(Y\,\{x \mapsto d\}) = \mathbf{t} \ \ (3)$$

⟨3⟩1. Fissato un qualsiasi $d \in D$, abbiamo
$$v_{I,A}(Y\,\{x \mapsto d\}) = v_I(Y\,\{x \mapsto d\}) \ \ (4)$$
Infatti,
$$\begin{cases} \underline{(\exists x)Y} \in \text{CForm} \Rightarrow \text{Free}(Y) \subseteq \{x\} \\ d \in D = \text{CTerm} \end{cases} \Rightarrow \text{Free}(Y\,\{x \mapsto d\}) = \emptyset \Leftrightarrow Y\,\{x \mapsto d\}) \in \text{CForm} \Rightarrow$$
$$\underbrace{\Rightarrow}_{\text{[5.26,pg.77]}} \quad v_{I,A}(Y\,\{x \mapsto d\}) = v_I(Y\,\{x \mapsto d\})$$

$$(3) \underbrace{\Leftrightarrow}_{(4)} \forall A \text{ assegn.} \quad \exists d \in D \;\; v_I(Y\{x \mapsto d\}) = \mathbf{t} \Leftrightarrow \exists d \in D \;\; \forall A \;\; v_I(Y\{x \mapsto d\}) = \mathbf{t} \Leftrightarrow$$

$$\Leftrightarrow \exists d \in D \;\; \forall A \;\; v_{I,A}(Y\{x \mapsto d\}) = \mathbf{t} \Leftrightarrow \exists d \in D \;\; M \models Y\{x \mapsto d\}$$

□

5.3 Notazione uniforme

Definition 5.33. Cosi' come abbiamo fatto in [4.25,pg.15], possiamo suddividere le formule in 4 categorie: le formule α, β, γ,δ.
Le α e le β sono le stesse di quelle gia' viste. Invece, per formule γ, δ sono cosi' definite:

$$\gamma \text{ e' una } \gamma\text{-formula} \overset{\text{def}}{\Leftrightarrow} \begin{cases} \gamma \in \left\{ \underline{(\forall x)Y}, \;\; \underline{\neg(\exists x)Y} \mid Y \in \text{Form}, \;\; x \in \text{Var} \right\} \\ \models \underline{\gamma \Leftrightarrow (\forall y)\gamma(y)}, \text{ con } y \in \text{Var}\backslash\text{Var}(\gamma) \cup \{x\}, \; \gamma(t) \in \{Y\{x \mapsto t\}, \; \neg Y\{x \mapsto t\}\}, \; \forall t \in \text{Term} \end{cases}$$

$$\delta \text{ e' una } \delta\text{-formula} \overset{\text{def}}{\Leftrightarrow} \begin{cases} \delta \in \left\{ \underline{(\exists x)Y}, \;\; \underline{\neg(\forall x)Y} \mid Y \in \text{Form}, \;\; x \in \text{Var} \right\} \\ \models \underline{\delta \Leftrightarrow (\exists t)\,\delta(t)}, \text{ con } y \in \text{Var}\backslash\text{Var}(\delta) \cup \{x\}, \; \delta(t) \in \{Y\{x \mapsto t\}, \; \neg Y\{x \mapsto t\}\}, \; \forall\, t \in \text{Term} \end{cases}$$

$\gamma(t)$ e' una funzione $\gamma(t) : \text{Term} \longrightarrow \text{Form}$ e si chiama *componente* o *istanza* della γ-formula γ.
Discorso analogo per $\delta(t)$.

Ecco la lista schematica delle γ e δ formule e delle relative istanze.

- γ-formula

$$\begin{array}{c|c} (\forall x)Y & Y\{x \mapsto t\} \\ \neg(\exists x)Y & \neg Y\{x \mapsto t\} \end{array}$$

Definiamo Form_γ come l'insieme delle γ-formule.

- β-formula

$$\begin{array}{c|c} (\exists x)Y & Y\{x \mapsto t\} \\ \neg(\forall x)Y & \neg Y\{x \mapsto t\} \end{array}$$

Definiamo Form_δ come l'insieme delle δ-formule.

Remark 5.1. Se $\gamma \in \{(\exists x)Y, \;\; \neg(\forall x)Y\}$ e $\gamma(t) \in \{Y\{x \mapsto t\}, \neg Y\{x \mapsto t\}\}$ e' l'istanza di γ, allora

$$\gamma(t) = Y\{x \mapsto t\} \Rightarrow \gamma(x) = Y$$
$$\gamma(t) = \neg Y\{x \mapsto t\} \Rightarrow \gamma(x) = \neg Y$$

e quindi

$$\gamma(t) = \gamma(x)\{x \mapsto t\}$$

Questa osservazione ci fara' comodo in seguito.

Proposition 5.34.

$$\gamma \in \text{Form}_\gamma, \;\; y \in \text{Var}\backslash(\text{Var}(\gamma) \cup \{x\}) \Rightarrow \models \underline{\gamma \Leftrightarrow (\forall y)\gamma(y)}$$
$$\delta \in \text{Form}_\delta, \;\; y \in \text{Var}\backslash(\text{Var}(\delta) \cup \{x\}) \Rightarrow \models \underline{\delta \Leftrightarrow (\exists y)\delta(y)}$$

dove x e' la variabile del quantificatore di γ, ad esempio, $\gamma = (\forall x)Z$.

Proposition 5.35. *Sia* $S\cup\{\gamma\}\subseteq$CForm*, con* γ *γ-formula, allora*

$$S\cup\{\gamma\}\ \textit{soddisfacibile}\ \Leftrightarrow \forall t\in \text{CTerm}\ \ S\cup\{\gamma,\gamma(t)\}\ \textit{e' soddisfacibile}$$

Sia $S\cup\{\delta\}\subseteq$CForm*, con* δ *δ-formula, allora*

$$S\cup\{\delta\}\ \textit{soddisfacibile}\ \Leftrightarrow \forall p\in \mathcal{F}_0\backslash\mathcal{F}_0(S\cup\{\delta\}):\quad S\cup\{\delta,\delta(p)\}\ \textit{e' soddisfacibile}$$

nota[9]

Proof:
⟨1⟩1. Dim. 1.
⟨2⟩1. Dim. $\Leftarrow$
Immediato in quanto, $S\cup\{\gamma\}\subseteq S\cup\{\gamma,\gamma(t)\}$
⟨2⟩2. Dim. $\Rightarrow$

$S\cup\{\gamma\}$ soddisfacibile $\Rightarrow \exists M=(D,I)$ modello t.c. M soddisfa $S\cup\{\gamma\}\Rightarrow$
$\Rightarrow \exists A$ assegnamento: $v_{I,A}(Y)=\mathbf{t}\ \ \forall Y\in S\cup\{\gamma\}\ \underbrace{\Leftrightarrow}_{S\cup\{\gamma\}\subseteq\text{CForm},\ [5.27,\text{pg.}78],\ [5.26,\text{pg.}77]}\ M\models S\cup\{\gamma\}\Rightarrow M\models\gamma\Rightarrow$

$$\underbrace{\Rightarrow}_{[5.34,\text{pg.}82],[5.22,\text{pg.}76]}\ M\models(\forall x)\gamma(x)\ \underbrace{\Rightarrow}_{[5.23,\text{pg.}76]}\ M\models\gamma(x)\Rightarrow v_{I,A}(\gamma(x))=\mathbf{t}\ \ \forall A\ \ (1)$$

Sia A' un assegnamento t.c. $x^{A'}=t^I$ (2)

$$v_{I,A'}(\gamma(t))\ \underbrace{=}_{[5.1,\text{pg.}82]}\ v_{I,A'}(\gamma(x)\{x\mapsto t\})\ \underbrace{=}_{(2),[5.28,\text{pg.}78]}\ v_{I,A'}(\gamma(x))\underbrace{=}_{(1)}\mathbf{t}$$

Quindi con A' riusciamo a soddisfare $\gamma(t)$. Poiche' $M\models S\cup\{\gamma\}$, $S\cup\{\gamma\}$ e' soddisfatto da qualsiasi A, in particolare da A'. Quindi A' soddisfa $S\cup\{\gamma,\ \gamma(t)\}$.
⟨2⟩3. Dim. 2.
Analogamente a prima.
⟨3⟩1. Dim. $\Leftarrow$
Immediato in quanto, $S\cup\{\delta\}\subseteq S\cup\{\delta,\delta(t)\}$
⟨3⟩2. Dim. $\Rightarrow$
Come prima abbiamo

$$\exists M=(D,I):\ \ M\models S\cup\{\delta\}\Rightarrow M\models\delta\Rightarrow M\models(\exists x)\delta(x)$$

Inoltre
$M\models(\exists x)\delta(x)\Rightarrow\forall A\ \ \exists B\in A_x:\ \ v_{I,B}(\delta(x))=\mathbf{t}\ \ (1)$
Sia $p\in\mathcal{F}_0\backslash\mathcal{F}_0(S\cup\{\delta\})$
Fissiamo un particolare A e sia J una interpretazione di D tale che sia uguale a I tranne in p e sia $p^J=x^B$
Poiche' $p\notin\text{Term}(S\cup\{\delta\})$, le intepretazioni delle formule $S\cup\{\delta\}$ rimangono uguali sia in $M=(D,I)$ che in $M'=(D,J)$, quindi, la (1) continua a valere per il modello $M'=(D,J)$:

$$M'\models S\cup\{\delta\},\ M'\models\delta,\quad v_{J,B}(\delta(x))=\mathbf{t}\ \ (1')$$

$$v_{J,B}(\delta(p))\ \underbrace{=}_{[5.1,\text{pg.}82]}\ v_{J,B}(\delta(x)\{x\mapsto p\})\ \underbrace{=}_{x^B=p^J,\ [5.28,\text{pg.}78]}\ v_{J,B}(\delta(x))\underbrace{=}_{(1')}\mathbf{t}$$

Quindi con il modello M' siamo riusciti a soddisfare $S\cup\{\delta,\delta(p)\}$

□

Theorem 5.36. *$\alpha,\beta,\gamma,\delta$-Induzione.*
Cosi' come abbiamo fatto in [4.29,pg.16], adesso presentiamo una tecnica di induzione strutturale

[9] con $\mathcal{F}_0(S\cup\{\delta\})$ intendiamo tutte le costanti presenti nelle formule $S\cup\{\delta\}$

che si basa sulle $\alpha, \beta, \gamma, \delta$-formule.
Sia $Q \subseteq$ Form. Se valgono le seguenti:

1. $\mathrm{AT}, \neg\mathrm{AT} \subseteq Q$, *dove* $\neg\mathrm{AT} = \{\neg X \mid X \in \mathrm{AT}\}$
2. $X \in Q \Rightarrow \neg\neg X \in Q$
3. $\alpha \in \mathrm{Form}_\alpha,\ \alpha_1, \alpha_2 \in Q \Rightarrow \alpha \in Q$
4. $\beta \in \mathrm{Form}_\beta,\ \beta_1, \beta_2 \in Q \Rightarrow \beta \in Q$
5. $\gamma \in \mathrm{Form}_\gamma,\ \forall t \in \mathrm{Term}\ \ \gamma(t) \in Q \Rightarrow \gamma \in Q$
6. $\delta \in \mathrm{Form}_\delta,\ \forall t \in \mathrm{Term}\ \ \delta(t) \in Q \Rightarrow \delta \in Q$

allora

$$Q = \mathrm{Form}$$

Proof: Si procede proprio come in [4.29,pg.16]: poniamo quindi

$$Q' = \{X \mid X, \neg X \in Q\} \subseteq Q \subseteq \mathrm{Form}$$

Dimostriamo solo i passi induttivi riguardanti i quantificatori. Nota, stavolta utilizziamo un induzione sul numero di costruttori (cosi' come abbiamo gia' fatto nella dimostrazione del punto 3. di [5.30,pg.79]).
Dimostriamo che $\gamma = (\forall x)X \in Q'$

Per Hp induttiva $X, X\{x \mapsto t\} \in Q' \Rightarrow X,\ \neg X,\ X\{x \mapsto t\},\ \neg X\{x \mapsto t\} \in Q$ (1)

$$\gamma(t) \underbrace{=}_{[5.1,\mathrm{pg}.82]} X\{x \mapsto t\} \underbrace{\in}_{(1)} Q \underbrace{\Rightarrow}_{5.} \gamma \in Q$$

$$\neg\gamma = \delta,\ \delta(x) = \neg X \Rightarrow \delta(t) = (\neg X)\{x \mapsto t\} \underbrace{\in Q}_{(1)} \underbrace{\Rightarrow}_{6.} \delta = \neg\gamma \in Q$$

$$\gamma, \neg\gamma \in Q \Rightarrow \gamma \in Q'$$

Analogamente si procede per il caso $(\exists x)$. □

Proposition 5.37. *Cosi' come abbiamo fatto in [4.32,pg.17], anche con le $\alpha, \beta, \gamma, \delta$-formule possiamo costruire senza problemi funzioni induttive.*

5.4 Sistemi di deduzione

Adesso vedremo l'estensione alla logica del I ordine dei sistemi di deduzione della logica proposizionale. Supporremo che il linguaggio L in cui operiamo contiene una infinita' numerabile di costanti "supplementari", che chiameremo *parametri*. In altre parole, se F_0 e' l'insieme delle costanti di L, allora utilizzeremo il linguaggio L^{Par} che ha $F_0 \cup \mathrm{Par}$ come insieme delle costanti e $|\mathrm{Par}| = |\mathbb{N}|$.

Inoltre, utilizzeremo principalmente formule chiuse (CTerm).

Theorem 5.38. *Il problema di verificare la validita' di una formula del I ordine e' indecidibile, ovvero non esiste alcun algoritmo che data una formula $Y \in$ Form come input, si ferma e dice se Y e' valida oppure no.*

Per questo motivo, tutti i sistemi di deduzione che vedremo potrebbero non fermarsi mai per qualche formula.

Proof: Tutti i sistemi di deduzione sono corretti e completi (lo vedremo in seguito), ovvero

$$\vdash X \Leftrightarrow \models X \quad (1)$$

Se, per assurdo, un sistema di deduzione giunge sempre a termine per qualsiasi formula X, possiamo sapere se

$$\vdash X \text{ oppure } \nvdash X$$

e quindi dalla (1) sappiamo se

$$\models X \text{ oppure } \not\models X$$

Quindi, utilizzando il sistema di deduzione abbiamo un algoritmo che risolve il problema di verificare la validita' di una formula. Ma questo e' assurdo perche' quest'ultimo problema e' indecidibile.
Quindi, esistera' almeno una formula che non ferma il sistema di deduzione.

Includiamo qui di seguito la dimostrazione dell'indecibilita':

The Undecidability of First Order Logic

A first order logic is given by a set of function symbols and a set of predicate symbols. Each function or predicate symbol comes with an arity, which is natural number. Function symbols of arity 0 are known as constant symbols. Now *terms* are recursively defined by

- variables are terms, and
- if t_i are terms for i=1,...,n and f is an n-ary function symbol, then $f(t_1,...,t_n)$ is a term.

Formulas are recursively defined by

- if t_i are terms for i=1,...,n and p is an n-ary predicate symbol, then $p(t_1,...,t_n)$ is a formula.
- if P and Q are formulas, then ¬P is a formula, P ∧ Q is a formula, P ∨ Q is a formula, P → Q is a formula, and P <=> Q is a formula.
- if P is a formula and x a variable, then ∀x: P and ∃ x: P are formulas.

The completeness theorem for first order logic says that a formula is provable from the laws of first order logic (not given here) if and only if it is true in under all possible interpretations, i.e. regardless of the meaning of the function and predicate symbols.

Theorem: It is undecidable whether a first order logic formula is provable (or true under all possible interpretations).

Proof: Suppose there is an algorithm B that, given a first order logic and a formula in that logic, decides whether that formula is valid (holds under all possible interpretations). I will use that to give a decision algorithm for the language {(M,w) | M is the description of a Turing machine that accepts the string w}. As the latter problem is undecidable this will show that B cannot exists.

Given M and w, create a first order logic by declaring a constant λ, a unary function symbol *a* for every letter *a* in the alphabet, and a binary predicate f_q for every state q of M.

Consider the following interpretation of this logic: Variables x range over strings over the given alphabet, λ denotes the empty string, *a*(w) denotes the string *a*w, and f_q(x,y) indicates that M, when given input w, can reach a

configuration with state q, in which xy is on the tape, with x in reverse order, and the head of M points at the first position of y. Under this interpretation $f_{q0}(\lambda,w)$ is certainly a true formula, as the initial configuration is surely reachable. Here q0 is the initial state, and w is a representation of w made from the constant and function symbols of the logic. Furthermore the formula $\exists x\ \exists y\colon f_{q\text{-}acc}(x,y)$ with q-acc the acceptance state, holds iff M accepts w.

Whenever M has a transition from state q to state r, reading a, writing b, and moving right, the formula

$$\forall x\ \forall y\colon f_q(x,ay) \Rightarrow f_r(bx,y)$$

holds. Here x and y are variables. Likewise, if M has a transition from state q to state r, reading a, writing b, and moving left, the formulas

$$\forall x\ \forall y\colon f_q(cx,ay) \Rightarrow f_r(x,cby)$$

hold for every choice of a letter c. In addition we have

$$\forall x\ \forall y\colon f_q(\lambda,ay) \Rightarrow f_r(\lambda,by),$$

covering the case that M cannot move left, because its head is already in the left-most position. Finally, there are variants of the formulas above for the case that a is the blank symbol and that square of the tape is visited for the first time:

$$\forall x\ \forall y\colon f_q(x,\lambda) \Rightarrow f_r(bx,\lambda)$$
$$\forall x\ \forall y\colon f_q(cx,\lambda) \Rightarrow f_r(x,cb\lambda)$$
$$\forall x\ \forall y\colon f_q(\lambda,\lambda) \Rightarrow f_r(\lambda,b\lambda).$$

Let T be the conjunction of all implication formulas mentioned above. As M has finitely many transitions and the alphabet is finite, this conjunction is finite as well, and thus a formula of first order logic. Now consider the formula

$$f_{q0}(\lambda,w)\ \&\ T \Rightarrow \exists x\ \exists y\colon f_{q\text{-}acc}(x,y).$$

In case M accepts w, there is a valid computation leading to an accept state. Each step therein corresponds with a substitution instance of one of the conjuncts in T, and using the laws of first order logic it is easy to check that the formula above is provable and thus true under all interpretations. If, on the other hand, the formula above is true under all interpretations, it is surely true in the given interpretation, which implies that M has an accepting computation starting on w.

Thus, in order to decide whether or not M accepts w, it suffices to check whether or not the formula above is a theorem of first order logic.

Rob van Glabbeek rvg@cs.stanford.edu

□

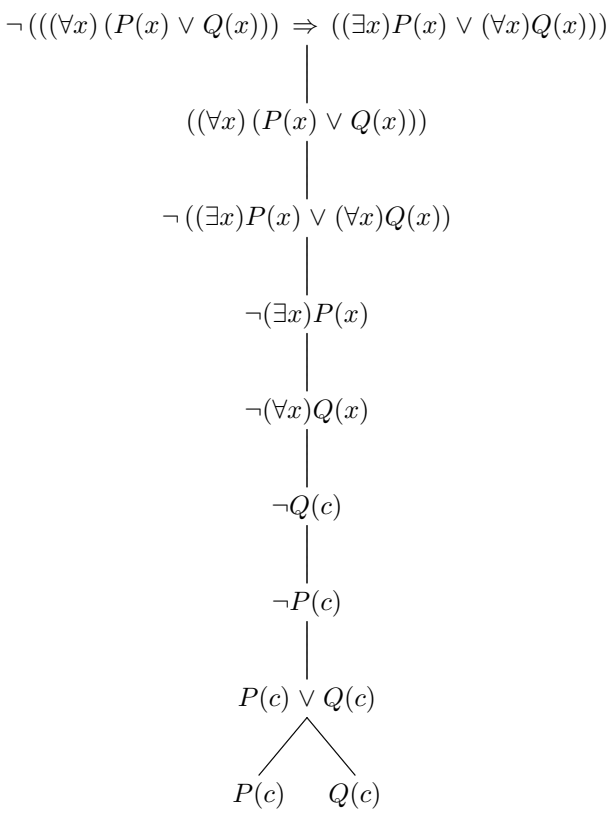
5.4.1 Tableaux I ordine

Definition 5.39. I tableaux per la logica del I ordine sono un estensione dei tableaux della logica proposizionale (vedi [4.38,pg.22]). Le uniche due regole che vengono aggiunte sono quelle riguardanti le γ e δ formule:

6. **Se** $N = \underline{\gamma}$:
   $r := r, [\gamma(t)]$ con $t \in \text{CTerm} \cup \text{Par}$
7. **Se** $N = \underline{\delta}$:
   $r : r, [\delta(p)]$ con $p \in \text{Par} \backslash \text{Par}(r)$, ovvero p e' un parametro nuovo per il ramo r

In pratica, la 7. dice che se $(\exists x)Y(x)$ e' vera, allora lo sara' anche $Y(p)$ per qualche p.

Example 5.40.



In ordine abbiamo applicato le seguenti regole:

$$\text{Riga 1} \underbrace{\longrightarrow}_{\text{regola 4.}} \text{Riga 2,3}$$
$$\text{Riga 3} \underbrace{\longrightarrow}_{\text{regola 4.}} \text{Riga 4,5}$$
$$\text{Riga 5} \underbrace{\longrightarrow}_{\text{regola 7.}} \text{Riga 6}$$
$$\text{Riga 4} \underbrace{\longrightarrow}_{\text{regola 6.}} \text{Riga 7}$$
$$\text{Riga 2} \underbrace{\longrightarrow}_{\text{regola 6.}} \text{Riga 8}$$
$$\text{Riga 8} \underbrace{\longrightarrow}_{\text{regola 5.}} \text{Riga 9}$$

Notiamo anche che abbiamo ottenuto un tableaux chiuso.

Theorem 5.41. *di correttezza per i tableaux del I ordine.*
Sia $X \in$ CForm*, allora*

$$\vdash X \Rightarrow \models X$$

Proof: La dimostrazione e' la stessa di quella vista in [4.42,pg.24]. Si deve solo estendere il lemma [4.41,pg.23]: per far questo, continuiamo la dimostrazione di [4.41,pg.23], verificando che l'applicazione delle regole 6., 7. (per le γ, δ-formule) genera ancora formule soddisfacibili e chiuse ($\in$ CTerm).
Case: r' e' stato ottenuto con la regola 6.
Quindi,

$$\exists N \in r : \ N = \gamma$$
$$r' = r, \gamma(t)$$

Poniamo $S = r$. Nota che $\gamma = N \in S$. Per Hp r contiene solo formule chiuse ($\in$ CTerm), quindi per il lemma [5.35,pg.83] abbiamo:

$$S \subseteq \text{CForm soddisfacibile} \ \Rightarrow S \cup \{\gamma(t)\} \ \text{soddisfacibile}$$

e poiche' $r' = S \cup \{\gamma(t)\} \subseteq$CTerm, abbiamo la tesi.
Case: r' e' stato ottenuto con la regola 7.
Si procede analogamente a prima.

□

5.4.2 Risoluzioni I ordine

Definition 5.42. Anche le risoluzioni vengono estese come per i Tableaux, aggiungendo le regole per le γ e δ.

Example 5.43.

$[\neg\left(\left(\left(\forall x\right)\left(P(x)\vee Q(x)\right)\right)\Rightarrow\left(\left(\exists x\right)P(x)\vee\left(\forall x\right)Q(x)\right)\right)]$
$[\left(\left(\forall x\right)\left(P(x)\vee Q(x)\right)\right)]$
$[\neg\left(\left(\exists x\right)P(x)\vee\left(\forall x\right)Q(x)\right)]$
$[\neg(\exists x)P(x)]$
$[\neg(\forall x)Q(x)]$
$[\neg Q(c)]$
$[\neg P(c)]$
$[P(c)\vee Q(c)]$
$[P(c),Q(c)]$
$[Q(c)]$
$[]$

Theorem 5.44. *di correttezza per le risoluzioni del I ordine.*
Sia $X\in$ CForm*, allora*

$$\vdash X\Rightarrow\ \models X$$

Proof: Cosi' come abbiamo fatto per la dimostrazione di [5.41,pg.90], si estende il lemma [4.47,pg.27]. Dopodiche' si puo' usare per intero la dimostrazione del thm [4.48,pg.28]. □

5.4.3 Hilbert del I ordine

Definition 5.45. Si usano gli stessi assiomi di [4.70,pg.48], con l'aggiunta del

$$\mathcal{A}_{10}=\left\{\underline{\gamma\Rightarrow\gamma(t)}\mid\gamma\in\mathrm{Form}_{\gamma},\quad t\in\mathrm{CTerm}\right\}$$

In piu', oltre al modus ponens si usa la regola della *Generalizzazione Universale* (g.u.):

$$S\cup\left\{\underline{Y\Rightarrow\gamma(p)}\right\}\vdash\underline{Y\Rightarrow\gamma}$$

dove $p\in\mathrm{Par}\backslash\mathrm{Par}(S\cup\left\{\underline{Y\Rightarrow\gamma}\right\})$ e' un parametro non contenuto ne' S, ne' in $\underline{Y\Rightarrow\gamma}$.

Proposition 5.46. *Sia* $S\subseteq$Form, $p\in\mathrm{Par}\backslash\mathrm{Par}(S),\gamma\in\mathrm{Form}_{\gamma}$*, si ha*

$$S\cup\{\gamma(p)\}\vdash\gamma$$

Proof:

1.	$\gamma(p)$	$\in\{\gamma(p)\}$
2.	$\gamma(p)\Rightarrow\mathbf{t}$	$\in\mathcal{A}_4$
3.	$\mathbf{t}$	*m.p.*
4.	$\gamma(p)\Rightarrow(\mathbf{t}\Rightarrow\gamma(p))$	$\in\mathcal{A}_1$
5.	$\mathbf{t}\Rightarrow\gamma(p)$	*m.p.*
6.	$\mathbf{t}\Rightarrow\gamma$	*g.u.*
7.	γ	m.p. tra la 3. e 6.

□

Theorem 5.47. *di deduzione.*

$$S \cup \{X\} \vdash Y \Leftrightarrow S \vdash \underline{X \Rightarrow Y}$$

Proof:
⟨1⟩1. Dim. $\Leftarrow$
Si procede come per la dimostrazione di [4.73,pg.49].
⟨1⟩2. Dim. $\Rightarrow$
Si procede come per la dimostrazione di [4.73,pg.49], ma stavolta abbiamo un caso in piu':
CASE: Z_i ottenuto da Z_h con la g.u., dove $h < i$
Supponiamo che $Z_h = \underline{Y \Rightarrow \gamma(p)}$, $Z_i = \underline{Y \Rightarrow \gamma}$. Nella derivazione $Z_1, \ldots, Z_n$ abbiamo quindi un pezzo del tipo:

$$Z_h = \underline{Y \Rightarrow \gamma(p)}$$
$$Z_i$$

Per Hp induttiva riusciamo a dimostrare da S la seguente formula:

$$X \Rightarrow Z_h$$

ovvero, abbiamo una dimostrazione a partire da S del tipo:

$$\vdots$$
$$X \Rightarrow (Y \Rightarrow \gamma(p))$$

da questa creiamo la seguente:

$$\vdots$$
1. $X \Rightarrow (Y \Rightarrow \gamma(p))$
2. $(X \wedge Y) \Rightarrow \gamma(p)$ dalla 1. per la logica proposizionale
3. $(X \wedge Y) \Rightarrow \gamma$ g.u.

Abbiamo cosi' trovato una dimostrazione di $X \Rightarrow Z_i$ a partire da S.
Nota: abbiamo potuto inserire la 2. perche' nella logica proposizionale si ha

$$\left\{\underline{(A \Rightarrow (B \Rightarrow C))}\right\} \models \underline{(A \wedge B) \Rightarrow C}$$

allora, per il thm di completezza dei sistemi di Hilbert (vedi [4.75,pg.52]), riusciamo a trovare una dimostrazione di $\underline{(X \wedge Y) \Rightarrow \gamma(p)}$ a partire da $\underline{X \Rightarrow (Y \Rightarrow \gamma(p))}$

□

Proposition 5.48.

1. $S \vdash \underline{\delta \Rightarrow \delta(p)}$
2. $S \cup \left\{\underline{\delta(p) \Rightarrow \phi}\right\} \vdash \underline{\delta \Rightarrow \phi}$

con $p \in \text{Par} \backslash \text{Par}(S \cup \{\delta \Rightarrow \phi\})$.

Proof:
⟨1⟩1. Dim. 1.
Per la prop [5.46,pg.91] e il thm di deduzione abbiamo:

$$S' \cup \{\gamma(p)\} \vdash \gamma \Leftrightarrow S' \vdash \underline{\gamma(p) \Rightarrow \gamma}$$

In particolare, posto $\gamma = \neg\delta$, si ha

$$S' \vdash \underline{\neg\delta(p) \Rightarrow \neg\delta} \quad (1)$$

Quindi:

1. $\neg\delta(p) \Rightarrow \neg\delta$ per la (1)
2. $\delta \Rightarrow \delta(p)$ dalla 2., per la logica proposizionale

⟨1⟩2. Dim. 2.

Posto $S' = S \cup \left\{\underline{\delta(p) \Rightarrow \phi}\right\}$, Costruiamo la seguente derivazione:

1. $\delta(p) \Rightarrow \phi$ $\in S'$
2. $\delta \Rightarrow \delta(p)$ passo 1.
3. $\delta \Rightarrow \phi$ dalla 1. e 3., per la logica proposizionale

□

Theorem 5.49. *di correttezza di Hilbert al I ordine.*
Sia $S \subseteq$ CForm *un insieme di enunciati,* $X \in$ CForm*, allora*

$$S \vdash X \Rightarrow S \models X$$

Proof: Riprendiamo la dimostrazione di [4.75,pg.52].
Nel caso base dobbiamo considerare anche l'assioma $\mathcal{A}_{10}$: basta verificare che

$$\models \{\gamma \Rightarrow \gamma(t)\}$$

ovvero che l'assioma 10 e' una formula valida.
Per il passo induttivo abbiamo un caso in piu':
Case: X_i ottenuto da X_j con la g.u

Abbiamo una dimostrazione del tipo:

$$X_j = \underline{Y \Rightarrow \gamma(p)}$$
$$\vdots$$
$$X_i = \underline{Y \Rightarrow \gamma}$$

Per Hp induttiva

$M = (D, I) \models S \Rightarrow M \models \underline{Y \Rightarrow \gamma(p)} \Leftrightarrow \forall A$ assegnamento di M $v_{I,A}(Y) \Rightarrow v_{I,A}(\gamma(p))$ (1)

Vogliamo dimostrare che $M \models \underline{Y \Rightarrow \gamma}$, ovvero

$$v_{I,A}(Y) \Rightarrow v_{I,A}(\gamma) \ \forall A$$

Fissiamo A e supponiamo $v_{I,A}(Y) = \mathbf{t}$. Dimostriamo che $v_{I,A}(\gamma) = \mathbf{t}$.

$v_{I,A}(\gamma) = \mathbf{t} \Leftrightarrow \forall B \in A_x v_{I,B}(\gamma(x)) = \mathbf{t}$ (2)

Fissiamo B. Sia J una interpretazione di D tale che sia uguale a I tranne in p e sia $p^J = x^B$. Sia $M' = (D, J)$

Poiche' $p \notin \text{Par}(S)$, si ha

$M \models S \Leftrightarrow M' \models S \underbrace{\Rightarrow}_{\text{Hp induttiva}} M' \models \underline{Y \Rightarrow \gamma(p)} \Leftrightarrow \forall A'$ assegn. in M' $v_{J,A'}(Y) \Rightarrow v_{J,A'}(\gamma(p))$ (1′)

Notiamo anche che gli assegnamenti di M e quelli di M' sono gli stessi, in quanto D e' invariato, quindi dalla (1′), ponendo $A' = A$, abbiamo:

$v_{J,A}(Y) \Rightarrow v_{J,A}(\gamma(p))$ (1″)

$\mathbf{t} = v_{I,A}(Y) \underbrace{=}_{p \notin \text{Par}(Y \Rightarrow \gamma)} v_{J,A}(Y) \underbrace{\Rightarrow}_{(1'')} v_{J,A}(\gamma(p)) = \mathbf{t}$ (3)

$$v_{J,A}(\gamma(p)) \underbrace{=}_{B \in A_x,\ x \notin \text{Free}(\gamma(p)) = \text{Free}(\gamma(x)\{x \mapsto p\})} v_{J,B}(\gamma(p)) \underbrace{=}_{[5.1,\text{pg.82}]} v_{J,B}(\gamma(x)\{x \mapsto p\}) \underbrace{=}_{x^B = p^J,\ [5.28,\text{pg.78}]} v_{J,B}(\gamma(x)) =$$
$$\underbrace{=}_{p \notin \text{Par}(\underline{Y \Rightarrow \gamma}) \Rightarrow p \notin \text{Par}(\gamma(x))} v_{I,B}(\gamma(x)) \quad (4)$$

Per l'arbitrarieta' di B e per la (2), (4) abbiamo quindi:

$$v_{I,A}(\gamma) = \mathbf{t}$$

ovvero la tesi.

□

5.4.4 Deduzione naturale I ordine

Definition 5.50. Anche il sistema di deduzione naturale viene esteso aggiungendo due regole:

$$\begin{aligned} &D12: \quad \gamma \vdash \gamma(t), \quad \text{con } t \in \text{CTerm} \\ &D13: \quad S \cup \{\delta\} \vdash \delta(p), \quad \text{con } p \in \text{Par} \backslash \text{Par}(S \cup \{\delta\}) \end{aligned}$$

Theorem 5.51. *di correttezza. Sia* $S \subseteq \text{CForm}$, $X \in \text{CForm}$, *allora*

$$S \vdash X \Rightarrow S \models X$$

Proof: Anche qui riprendiamo la dimostrazione del thm [4.12,pg.11]. Supponiamo quindi $X_n = \gamma(t) \notin S \cup A_n$ e dimostriamo che

$$S \cup A_n \models X_n$$

Dato che $X_n \notin S \cup A_n$, X_n e' stato ricavato con la D_{12}, ovvero

$$X_n = \gamma(t)$$
$$\gamma = X_m, \quad m < n$$
$$\text{Hp induttiva} \Rightarrow S \cup A_m \models \gamma \quad (*)$$

Per utilizzare la D_{12}, la X_m deve essere una formula attiva per X_n, ovvero non deve stare in scatole chiuse in cui X_n non e' presente. In altre parole,

$$A_m \subseteq A_n$$

quindi

$$M \models S \cup A_n \underbrace{\Rightarrow}_{A_m \subseteq A_n} M \models S \cup A_m \underbrace{\Rightarrow}_{(*)} M \models \gamma \Rightarrow \forall A \;\; v_{I,A}(\gamma(x)) = \mathbf{t} \Rightarrow$$

$$\Rightarrow \forall A: \;\; x^A = t^I \;\; v_{I,A}(\gamma(t)) = v_{I,A}(\gamma(x)\{x \mapsto t\}) = v_{I,A}(\gamma(x)) = \mathbf{t} \underbrace{\Rightarrow}_{\gamma \in \text{CForm} \Rightarrow \gamma(t) \in \text{CForm}} v_I(\gamma(t)) = \mathbf{t} \Rightarrow M \models \gamma(t)$$

Abbiamo cosi' dimostrato che

$$S \cup A_n \models \gamma(t)$$

Per il caso $X_n = \delta(p)$: analogamente a prima:

$$M \models A_n \underbrace{\Rightarrow}_{A_m \subseteq A_n} M \models S \cup A_m \Rightarrow M \models \delta \Rightarrow \forall A \, \forall B \in A_x \;\; v_{I,B}(\delta(x)) = \mathbf{t} \;\; (*)$$

$$M \models \delta(p) \Leftrightarrow \forall A \;\; v_{I,A}(\delta(p)) = \mathbf{t} \;\; (**)$$

Fissato A, sia $B \in A_x$ t.c. $x^B = p^I$

$$v_{I,A}(\delta(p)) \underbrace{=}_{x \notin \text{Free}(\delta(p))} v_{I,B}(\delta(p)) = v_{I,B}(\delta(x)\{x \mapsto p\}) \underbrace{=}_{x^B = p^I} v_{I,B}(\delta(x)) \underbrace{=}_{(*)} \mathbf{t}$$

Quindi la $(**)$ e' vera. □

5.5 Completezza semantica

Per dimostrare i rispettivi teoremi di completezza semantica, dobbiamo estendere il concetto di insieme di Hintikka (vedi [4.49,pg.28]) e di Proprieta' di consistenza (vedi [4.51,pg.29]) alla logica del I ordine. Fatto cio', sara' possibile convertire facilmente le dimostrazioni di completezza dei vari sistemi di deduzione dal caso proposizione a quello del I ordine.

Definition 5.52. La seguente definizione e' l'estensione della def [4.49,pg.28].
Sia $H \subseteq \text{CForm}$ un insieme di enunciati.

H e' un insieme di *Hintikka* $\overset{\text{def}}{\Leftrightarrow}$

1. $R(t_1,\ldots,t_n) \in \text{AT} \Rightarrow (R(t_1,\ldots,t_n) \notin H \text{ oppure } \neg R(t_1,\ldots,t_n) \notin H)$
2. $\neg\mathbf{t},\ \mathbf{f} \notin H$
3. $\neg\neg Z \in H \Rightarrow Z \in H$
4. $\alpha \in H \Rightarrow \alpha_1, \alpha_2 \in H$
5. $\beta \in H \Rightarrow \beta_1 \in H$ oppure $\beta_2 \in H$
6. $\gamma \in H \Rightarrow \gamma(t) \in H \ \ \forall t \in \text{CTerm}$
7. $\delta \in H \Rightarrow \exists t \in \text{CTerm} : \ \delta(t) \in \text{CTerm}$

Proposition 5.53. *Supposto che* CTerm $\neq \emptyset$ *e* $H \subseteq$ CForm*, si ha*

$$H \text{ Hintikka } \Rightarrow H \text{ e' soddisfacibile}$$

In particolare H *sara' soddisfacibile in un modello di Herbrand.*

Proof: Proof sketch:Sia $D = \text{CTerm}$. Costruiamo una interpretazione I che soddisfa H. Per far cio' bastera' assegnare $\mathbf{t}$ ad ogni formula atomica in H. Gli assiomi di H garantiranno che tutte le formule costruite da quelle formule sono $\mathbf{t}$ Definiamo I:
$I : \mathcal{F} \cup \mathcal{R} \longrightarrow D \cup \{\text{funzioni su D}\} \cup \{\text{relazioni su D}\}$ (1)

$c^I = c \ \ \forall c \in \mathcal{F}_0$

f^I e' quella funzione cosi' definita: $f^I(t_1,\ldots,t_n) = f(t_1,\ldots,t_n) \ \ \forall t_i \in \text{CTerm}$, nota che $f(t_1,\ldots,t_n) \in \text{CTerm} = D$

R^I e' la relazione cosi' definita: $R^I(t_1,\ldots,t_n) = \begin{cases} \mathbf{t} & R(t_1,\ldots,t_n) \in H \\ \mathbf{f} & \text{altrimenti} \end{cases} \quad \forall t_i \in \text{CTerm}$, ovvero

$(t_1,\ldots,t_n) \in R^I \Leftrightarrow R(t_1,\ldots,t_n) \in H \ \ \forall t_i \in \text{CTerm}$

Nota: nelle definizioni di sopra stiamo considerando solo termini t_i chiusi perche' $D = \text{CTerm}$.
Addesso dimostriamo per induzione (vedi [5.36,pg.83]) che

$$Y \in H \Rightarrow (D, I) \models Y$$

Case: base

Case: $Y = R(t_1,\ldots,t_n)$

$$v_{I,A}(R(t_1,\ldots,t_n)) = \mathbf{t} \underbrace{\Leftrightarrow}_{Y \in H \subseteq \text{CForm} \Rightarrow t_i \in \text{CTerm},\ [5.26,\text{pg.}77]} v_I(R(t_1,\ldots,t_n)) = \mathbf{t} \Leftrightarrow (t_1^I,\ldots,t_n^I) \in R^I \underbrace{\Leftrightarrow}_{(1)} Y \in H$$

Case: $Y = \neg R(t_1,\ldots,t_n)$

$$\neg R(t_1,\ldots,t_n) \in H \underbrace{\Rightarrow}_{H \text{ Hintikka}} R(t_1,\ldots,t_n) \notin H \underbrace{\Leftrightarrow}_{(1)} (t_1,\ldots,t_n) \notin R^I \Leftrightarrow v_I(R(t_1,\ldots,t_n)) = \mathbf{f} \Leftrightarrow$$

$$\Leftrightarrow v_I(\neg R(t_1,\ldots,t_n)) = \mathbf{t}$$

Case: $Y = \mathbf{t}$

$$v_I(\mathbf{t}) = \mathbf{t}$$

Case: $Y = \mathbf{f}$

Questo caso non si puo' verificare perche' $\mathbf{f} \notin H$

Case: $Y = \neg\mathbf{t}$

Anche questo caso non si verifica

Case: $Y = \neg\mathbf{f}$

$$v_I(Y) = \neg \mathbf{f} = \mathbf{t}$$

CASE: $Y = \alpha$

$$Y = \alpha \in H \Rightarrow \alpha_1, \alpha_2 \in H \underbrace{\Rightarrow}_{\text{Hp induttiva}} v_I(\alpha_1) = v_I(\alpha_2) = \mathbf{t} \Rightarrow v_I(\alpha) = (v(\alpha_1) \wedge v_I(\alpha_2)) = \mathbf{t}$$

CASE: $Y = \beta$

Analogamente a prima.

CASE: $Y = \gamma$

$$Y = \gamma \in H \Rightarrow \gamma(t) \in H \ \ \forall t \in D \underbrace{\Rightarrow}_{\text{Hp indutt}} v_I(\gamma(t)) = \mathbf{t} \ \ \forall t \in D \Leftrightarrow M \models \gamma(t) \ \ \forall t \in D \Leftrightarrow M \models \gamma\{x \mapsto t\} \ \ \forall t \in D \Leftrightarrow$$

$$\underbrace{\Leftrightarrow}_{[5.32,\text{pg.}81]} M \models (\forall x)\gamma(x)$$

CASE: $Y = \delta$

Analogamente a prima.

□

Definition 5.54. Sia $C \subseteq \mathcal{P}(\text{CForm}(L^{\text{Par}}))$, dove $\text{CForm}(L^{\text{Par}})$ sono le formule chiuse di L^{Par}, estensione di L.
C e' una *proprieta' di consistenza* di $L \overset{\text{def}}{\Leftrightarrow} \forall S \in C$

1. $R(t_1, \ldots, t_n) \in \text{AT} \Rightarrow R(t_1, \ldots, t_n) \notin S$ oppure $\neg R(t_1, \ldots, t_n) \notin S$
2. $\mathbf{f}$, $\neg \mathbf{t} \notin S$
3. $\neg\neg Z \in S \Rightarrow S \cup \{Z\} \in C$
4. $\alpha \in S \Rightarrow S \cup \{\alpha_1, \alpha_2\} \in C$
5. $\beta \in S \Rightarrow S \cup \{\beta_1\}$ oppure $S \cup \{\beta_2\} \in C$
6. $\gamma \in S \Rightarrow S \cup \{\gamma(t)\} \in C \ \ \forall t \in \text{CTerm}(L^{\text{Par}})$
7. $\delta \in S \Rightarrow \exists p \in \text{Par} : \ \ S \cup \{\delta(p)\} \in C$

Theorem 5.55. *di esistenza del modello per il I ordine.*
Sia C una proprieta' di consistenza, allora

$$S \in C \Rightarrow S \text{ e' soddisfacibile}$$

In particolare S e' soddisfacibile in un modello di Herbrand di L^{Par}.

Proof: Poiche' $\text{CTerm}(L^{\text{Par}})$ e' numerabile, possiamo scrivere:

$$\text{CTerm}(L^{\text{Par}}) = \{t_1, t_2, \ldots, \}$$

Consideriamo il seguente algoritmo:

$T_0 := \{(X, 1) \mid X \in S\}$
$U_0 := S$
$i := 0$
While $T_i \neq \emptyset$
  $(X, j) := \mathrm{Pop}(T_i, i)$
  $T_i := T_i \setminus \{(X, j)\}$
  Se $X := \neg\neg Z$:
    $T_{i+1} := T_i \cup \{(Z, 1)\}$
    $U_{i+1} := U_i \cup \{Z\}$
  Se $X := \alpha$:
    $T_{i+1} := T_i \cup \{(\alpha_1, 1), (\alpha_2, 1)\}$
    $U_{i+1} := U_i \cup \{\alpha_1, \alpha_2\}$
  Se $X := \beta$:
    Sia $b \in \{\beta_1, \beta_2\} : \ U_i \cup \{b\} \in C$
    $T_{i+1} := T_i \cup \{(b, 1)\}$
    $U_{i+1} := U_i \cup \{b\}$
  Se $X := \gamma$:
    Se $j + 1 \leq |\mathrm{CTerm}|$:
      $T_{i+1} := T_i \cup \{(\gamma(t_j), 1), (\gamma, j + 1)\}$
    Altrimenti :
      $T_{i+1} := T_i \cup \{(\gamma(t_j), 1)\}$
    $U_{i+1} := U_i \cup \{\gamma(t_j)\}$
  Se $X := \delta$:
    Sia $p \in \mathrm{Par} : \ U_i \cup \{\delta(p)\} \in C$
    $T_{i+1} := T_i \cup \{(\delta(p), 1)\}$
    $U_{i+1} := U_i \cup \{\delta(p)\}$
  $i := i + 1$

dove la funzione Pop e' cosi' definita:

$$(X, j) = \mathrm{Pop}(T_i, i) \in T_i \text{ tale che } j = \min\{k \in \mathbb{N} \mid (X', k) \in T_i\} \quad (2)$$

Pop serve a pescare un elemento da T_i in modo tale da avere:

$$\forall i \, \forall (X, j) \in T_i \ \ \exists k \geq i : \ \ \mathrm{Pop}(T_k, k) = (X, j) \quad (2')$$

ovvero, in modo da prendere tutti gli elementi dei T_i in un numero finito di passi finito. Infatti, se non fosse rispettata la (2), l'algoritmo potrebbe cadere nel seguente loop infinito:

$\mathrm{Pop}(T_0, 0) = (X, j) = (\gamma, 1) \in T_0$
$T_1 = T_0 \cup \{(\gamma, 2)\}$
$\mathrm{Pop}(T_1, 1) = (X, j) = (\gamma, 2) \in T_1$
$T_2 = T_1 \cup \{(\gamma, 3)\}$
$\vdots$

oppure

$$\begin{aligned}
&\mathrm{Pop}(T_0, 0) = (X, j) = (\gamma, 1) \in T_0\\
&T_1 = T_0 \cup \{(\gamma, 2)\}\\
&\mathrm{Pop}(T_1, 1) = (X, j) = (\gamma', 1) \in T_1\\
&T_2 = T_1 \cup \{(\gamma', 2)\}\\
&\mathrm{Pop}(T_2, 2) = (X, j) = (\gamma, 2) \in T_2\\
&T_3 = T_2 \cup \{(\gamma, 3)\}\\
&\mathrm{Pop}(T_3, 3) = (X, j) = (\gamma', 2) \in T_3\\
&T_4 = T_3 \cup \{(\gamma', 4)\}\\
&\vdots
\end{aligned}$$

Intuitivamente, l'algoritmo utilizza T come un contatore e U come accumulatore, in cui aggiunge tutto quello che e' possibile estrapolare da S (avevamo gia' visto un algoritmo simile, vedi [4.53,pg.30]). Valgono le seguenti proprieta':

$$U_i \subseteq U_{i+1} \quad \forall i \quad (3)$$
$$U_i \in C \quad \forall i \quad (4)$$
$$S = U_0 \subseteq U_i \quad \forall i \quad (5)$$
$$(X, j) \in T_i \Rightarrow X \in U_i \quad \forall i \quad (6)$$

$\langle 1 \rangle 1$. Dim. che $H = \bigcup_{i \in \mathbb{N}} U_i$ e' un insieme di Hintikka

1. Se per assurdo $R(t_1, \ldots, t_n), \neg R(t_1, \ldots, t_n) \in H$, allora
   $$R(t_1, \ldots, t_n) \in U_i$$
   $$\neg R(t_1, \ldots, t_n) \in U_j$$
   $$U_{\min\{i,j\}} \subseteq U_{\max\{i,j\}} \Rightarrow \{R(t_1, \ldots, t_n), \neg R(t_1, \ldots, t_n)\} \subseteq U_{\max\{i,j\}}$$
   assurdo contro (4)
2. Analogamente $\neg t, \mathbf{f} \notin H$
3. Se $\neg\neg Z \in H$, allora
   Case: $\neg\neg Z \in S = U_0$

   $$\neg\neg Z \in S = U_0 \Rightarrow (\neg\neg Z, 1) \in T_0 \underbrace{\Rightarrow}_{(2')} \exists k : \ \mathrm{Pop}(T_k, k) = (\neg\neg Z, 1)$$

   Quindi, la formula $\neg\neg Z$ verra' processata dall'algoritmo e si avra'
   $$Z \in U_{k+1} \subseteq H$$
   Case: $\neg\neg Z \notin S = U_0$
   In questo caso, $\neg\neg Z$ e' stata aggiunta dall'algoritmo in qualche U_i e inoltre
   $$(\neg\neg Z, 1) \in T_i$$
   Quindi, per la $(2')$
   $$\exists k : \ \mathrm{Pop}(T_k, k) = (\neg\neg Z, 1)$$
   e come prima, si avra':
   $$Z \in U_{k+1} \subseteq H$$
4. Gli altri casi si dimostrano analogamente.

$$H \text{ Hintikka} \underbrace{\Rightarrow}_{[5.53,\text{pg.}95]} H \text{ soddisf} \underbrace{\Rightarrow}_{S \subseteq H} S \text{ soddisf}$$

□

Theorem 5.56. *di completezza.*
Le dimostrazioni dei teoremi di completezza sfruttano il teorema di esistenza del modello [5.55,pg.96]

e si possono facilmente ricavare dalle dimostrazioni gia' viste per la logica proposizionale.
Per la dimostrazione del thm di completezza di Hilbert si sfrutta anche la prop [5.48,pg.92].

Theorem 5.57. *di compattezza per il I ordine. Sia S un insieme di enunciati del linguaggio L, allora*

$$S \text{ finitamente soddisfacibile } \Rightarrow S \text{ soddisfacibile}$$

In particolare, S sara' soddisfatto da un modello di Herbrand in L^{Par}.

Proof: Sia

$$C = \{W \subseteq \mathrm{CForm} \mid W \text{ finit. soddisf. }, |\mathrm{Par} \backslash \mathrm{Par}(W)| = \infty\}$$

$S \in C$ perche' e' finitamente soddisfacibile, ed essendo in L non utilizza parametri, ovvero $\mathrm{Par}(S) = \emptyset$.
Dimostriamo che C e' una proprieta' di consistenza.
Sia $W \in C$.
Consideriamo solo il caso γ e δ dato che per gli altri si procede analogamente a quanto visto in [4.61,pg.41].
Supponiamo $\gamma \in W$, dimostriamo $W \cup \{\gamma(t)\} \in C$. Chiaramente, $|\mathrm{Par} \backslash \mathrm{Par}(W \cup \{\gamma(t)\})| = \infty$
Per assurdo $\exists D \subseteq W \cup \{\gamma(t)\}$ finito ma non soddisfacibile con $\gamma(t) \in D$. Allora,

$$(D \setminus \{\gamma(t)\}) \cup \{\gamma\} \subseteq W \text{ e' finito } \underbrace{\Rightarrow}_{W \text{ fin. sodd.}} \Rightarrow (D \setminus \{\gamma(t)\}) \cup \{\gamma\} \text{ soddisf } \Rightarrow$$

$$\underbrace{\Rightarrow}_{[5.35,\mathrm{pg}.83]} (D \setminus \{\gamma(t)\}) \cup \{\gamma\} \cup \{\gamma(t)\} \text{ soddisf} \Rightarrow D \text{ soddisf}$$

assurdo.

Supponiamo $\delta \in W$, dimostriamo $\exists p \in \mathrm{Par} : W \cup \{\delta(p)\} \in C$.

$$\mathrm{Par} \backslash \mathrm{Par}(W) \neq \emptyset \underbrace{\Rightarrow}_{\delta \in W} \exists p' \in \mathrm{Par} \backslash \mathrm{Par}(W \cup \{\delta\}) \subseteq \mathcal{F}_0 \backslash \mathcal{F}_0(W \cup \{\delta\}) \quad (*)$$

Per assurdo

$$\forall p \in \mathrm{Par}\ \exists D_p \subseteq W \cup \{\delta(p)\} \text{ finito ma non soddisfacibile, con } \delta(p) \in D_p$$

$$(*) \Rightarrow p' \notin \mathcal{F}_0(W) \underbrace{\Rightarrow}_{D_{p'} \setminus \{\delta(p')\} \subseteq W} p' \notin \mathcal{F}_0(D_{p'} \setminus \{\delta(p')\}) \quad (**)$$

$$\left(D'_p \setminus \{\delta(p')\}\right) \cup \{\delta\} \subseteq W \text{ e' finito } \underbrace{\Rightarrow}_{W \text{ fin. soddisf.}} \left(D'_p \setminus \{\delta(p')\}\right) \cup \{\delta\} \text{ soddisf } \Rightarrow$$

$$\underbrace{\Rightarrow}_{(**),\ [5.35,\mathrm{pg}.83]} \left(D'_p \setminus \{\delta(p')\}\right) \cup \{\delta, \delta(p')\} \supseteq D'_p \text{ soddisf}$$

assurdo.

$$S \in C \underbrace{\Rightarrow}_{\text{thm esistenza del modello}} S \text{ soddisf.}$$

□

Theorem 5.58. *di Herbrand.*
Sia $S \subseteq$ CForm, nel linguaggio L, allora

$$S \text{ soddisfacibile } \Leftrightarrow S \text{ soddisfacibile in un modello di Herbrand di } L^{\mathrm{Par}}$$

Proof:
⟨1⟩1. Dim. $\Leftarrow$

Per definizione.
⟨1⟩2. Dim. $\Rightarrow$
Consideriamo il seguente insieme

$$C = \{W \subseteq \text{CForm} \mid W \text{ soddisfacibile}, |\text{Par}\backslash\text{Par}(W)| = \infty\}$$

E' facile verificare che C e' una proprieta' di consistenza (infatti, le regole della def. [5.54,pg.96] preservano tutte la soddisfacibilita'). Inoltre,

$$\begin{cases} S \text{ soddisf} \\ S \subseteq \text{CForm nel linguaggio } L \Rightarrow \text{Par}(S) = \emptyset \end{cases} \Rightarrow S \in C \underbrace{\Rightarrow}_{[5.55,\text{pg.}96]} S \text{ soddisf. in un modello di Herbrand di } L^{\text{Par}}$$

□

Corollary 5.59. *Sia $X \in$ CForm, nel linguaggio L, allora*

$$X \textit{ valida } \Leftrightarrow X \textit{ e' vera in ogni modello di Herbrand di } L^{\text{Par}}$$

Proof:
⟨1⟩1. Dim. $\Rightarrow$

$$X \text{ valida } \Rightarrow X \text{ vera in ogni modello}$$

e quindi la tesi.
⟨1⟩2. Dim. $\Leftarrow$
Supponiamo per assurdo che

$$\exists M \text{ modello: } X \text{ non e' vera in } M \Leftrightarrow M \text{ soddisfa } \neg X \Leftrightarrow$$
$$\underbrace{\Leftrightarrow}_{[5.58,\text{pg.}99]} \neg X \text{ e' soddisfatto in un modello di Herbrand di } L^{\text{Par}} \Leftrightarrow$$
$$\Leftrightarrow X \text{ non e' vera in un modello di Herbrand di } L^{\text{Par}}$$

assurdo.

□

5.6 Skolemizzazione

Theorem 5.60. *di sostituzione per il I ordine.*
Sia $a \in$ AT, $X, Y, Z \in$ Form e M un modello.

$$M \models \underline{Y \Leftrightarrow Z} \Rightarrow M \models \underline{X(a|Y) \Leftrightarrow X(a|Z)}$$

dove con $X(a|Y)$ indichiamo la formula X in cui ogni occorrenza di a e' stata sostituita con Y (e' l'analogo di quanto facevamo in [4.21,pg.14]).

In particolare si ha

$$\models \underline{Y \Leftrightarrow Z} \Rightarrow \models \underline{X(a|Y) \Leftrightarrow X(a|Z)}$$

Proof: Si procede per induzione strutturale. Analizziamo solo il caso delle γ e β.
Per Hp induttiva

$$M \models \underline{\gamma(x)(a|Y) \Leftrightarrow \gamma(x)(a|X)} \Leftrightarrow \forall B \text{ assegnamento } M, B \models \underline{\gamma(x)(a|Y) \Leftrightarrow \gamma(x)(a|X)} \quad (1)$$

Se per assurdo $M \not\models \underline{\gamma(a|Y) \Leftrightarrow \gamma(a|X)} \Rightarrow \exists A$ assegn. t.c. $M, A \not\models \underline{\gamma(a|Y) \Leftrightarrow \gamma(a|X)}$

Quindi, supponendo che $v_{I,A}(\gamma(a|Y)) = \mathbf{t}$, $v_{I,A}(\gamma(a|X) = \mathbf{f}$, abbiamo[10]

$$v_{I,A}(\gamma(a|Y)) = \mathbf{t} \Leftrightarrow \forall B \in A_x \;\; v_{I,B}(\gamma(x)(a|Y)) = \mathbf{t}$$
$$v_{I,A}(\gamma(a|X)) = \mathbf{f} \Leftrightarrow \exists C \in A_x : \quad v_{I,C}(\gamma(x)(a|X)) = \mathbf{f}$$
$$\Rightarrow v_{I,C}(\gamma(x)(a|Y)) \neq v_{I,C}(\gamma(x)(a|X))$$

assurdo contro la (1).
Analogamente si procede per la δ □

Definition 5.61. Sia $a \in \mathrm{AT}$ e $Y(a) \in \mathrm{Form}$ (nota[11]). Diremo che tutte le occorrenze di a in Y sono positive, e scriveremo $a \in \mathrm{AT}^+(Y) \;\Leftrightarrow\;$ accade una delle seguenti condizioni:

1. $Y = a$
2. $Y = \neg\neg a$
3. $Y = \alpha, \;\; a \in \mathrm{AT}^+(\alpha_1)$ e $a \in \mathrm{AT}^+(\alpha_2, a)$
4. $Y = \beta, \;\; a \in \mathrm{AT}^+(\beta_1)$ e $a \in \mathrm{AT}^+(\beta_2)$
5. $Y = (\forall x)\gamma(x), \;\; a \in \mathrm{AT}^+(\gamma(x))$
6. $Y = (\exists x)\delta(x), \;\; a \in \mathrm{AT}^+(\delta(x))$

In pratica, a occorre positivamente in Y quando non compare mai come $\neg a$ in Y, dopo tutte le semplificazioni dei $\neg$.
Analogamente, a occorre negativamente in Y quando tutte le sue occorrenze sono del tipo $\neg a$, o piu' precisamente

$$a \in \mathrm{AT}^-(Y) \stackrel{\mathrm{def}}{\Leftrightarrow} a \in \mathrm{AT}^+(\neg Y)$$

Theorem 5.62. *di sostituzione per implicazioni.*
Sia $X, Y, Z \in \mathrm{Form}$. *Allora si ha*

1. $\begin{cases} a \in \mathrm{AT}^+(X) \\ M \models \underline{Y \Rightarrow Z} \end{cases} \Rightarrow M \models \underline{X(a|Y) \Rightarrow X(a|Z)}$
2. $\begin{cases} b \in \mathrm{AT}^-(X) \\ M \models \underline{Y \Rightarrow Z} \Rightarrow M \models \underline{X(b|Z) \Rightarrow X(b|Y)} \end{cases}$

Proof:
⟨1⟩1. Dim. 1.
Supponiamo che $a \in \mathrm{AT}(X)$ (altrimenti $X \Rightarrow X$ e' una tautologia e non abbiamo nulla da dimostrare).
Per induzione strutturale su X.
Case: Caso atomico: $X = a$

$$X(a|Y) = Y, \; X(a|Z) = Z$$

e quindi la tesi e' chiaramente vera.

Nota il caso $X = \neg a$ non puo' verificarsi in quanto a occorre positivamente in X.
Case: $X = \alpha$

[10] per definizione di sostituzione, si ha $\gamma(a|Y) = (\forall x)(\gamma(x)(a|Y))$
[11] con $Y(A)$ indichiamo che a e' una sottoformula (atomica) di Y

$X(a|Y) = \alpha(a|Y) = (\alpha_1(a|Y) \wedge \alpha_2(a|Y))$

Hp indutt. $\Rightarrow \begin{cases} M \models \underline{\alpha_1(a|Y) \Rightarrow \alpha_1(a|Z)} \\ M \models \underline{\alpha_2(a|Y) \Rightarrow \alpha_2(a|Z)} \end{cases} \Rightarrow M \models \underline{(\alpha_1(a|Y) \wedge \alpha_2(a|Y)) \Rightarrow (\alpha_1(a|Z) \wedge \alpha_2(a|Z))} \Leftrightarrow$

$\Leftrightarrow \underline{X(a|Y) \Rightarrow X(a|Z)}$

Case: $X = \beta$

Analogamente a prima.

Case: $X = \neg\neg F$

$X(a|Y) = \neg\neg F(a|Y)$

Per Hp induttiva $M \models \underline{F(a|Y) \Rightarrow F(a|Z)} \underbrace{\Rightarrow}_{\models \neg\neg A \Leftrightarrow A} M \models \underline{\neg\neg F(a|Y) \Rightarrow \neg\neg F(a|Z)}$

Case: $X = \gamma$

$X(a|Y) = ((\forall x)\gamma(x))(a|Y) = (\forall x)(\gamma(x)(a|Y))$

Per Hp induttiva $M \models \underline{\gamma(x)(a|Y) \Rightarrow \gamma(x)(a|Z)} \Leftrightarrow \forall A$ assegnamento $(v_{I,A}(\gamma(x)(a|Y)) \Rightarrow v_{I,A}(\gamma(x)(a|Z)))$ $(*)$

Se per assurdo $M \not\models \underline{\gamma(a|Y) \Rightarrow \gamma(a|Z)} \Rightarrow \exists A$ assegnamento : $v_{I,A}\left(\underline{\gamma(a|Y) \Rightarrow \gamma(a|Z)}\right) = \mathbf{f} \Leftrightarrow$

$\Leftrightarrow v_{I,A}(\gamma(a|Y)) = \mathbf{t},\ v_{I,A}(\gamma(a|Z)) = \mathbf{f} \Leftrightarrow \forall B \in A_x\ v_{I,B}(\gamma(x)(a|Y)) = \mathbf{t},\ \exists B' \in A_x :\ v_{I,B'}(\gamma(x)(a|Z)) = \mathbf{f} \Rightarrow$

$\Rightarrow (v_{I,B'}(\gamma(x)(a|Y)) \not\Rightarrow v_{I,B'}(\gamma(x)(a|Z)))$

assurdo contro $(*)$.

Case: $X = \delta$

Analogamente a prima.

$\langle 1 \rangle 2$. Dim. 2.

Stavolta, l'unico caso atomico possibile e' $X = \neg b$.

Case: Caso atomico: $X = \neg b$

$X(b|Y) = \neg Y,\ X(b|Z) = \neg Z$

$M \models \underline{Y \Rightarrow Z} \Rightarrow M \models \underline{\neg Z \Rightarrow \neg Y} \Leftrightarrow M \models \underline{X(b|Z) \Rightarrow X(b|Y)}$

e quindi la tesi e' vera.

Negli altri casi si procede analogamente a prima.

□

Lemma 5.63. *Sia* $\Psi \in$ Form *con* $\text{Free}(\Psi) = \{x, y_1, y_2, \ldots, y_n\}$ *variabili libere. Sia* $f \in \mathcal{F}_n \backslash \mathcal{F}_n(\Psi)$ *e sia* $M = (D, I)$ *un allora*

1. $\exists M_1 = (D, J_1)$ *modello t.c.* $M_1 \models \underline{(\exists x)\Psi \Rightarrow \Psi\{x \mapsto f(y_1, \ldots, y_n)\}}$
2. $\exists M_2 = (D, J_2)$ *modello t.c.* $M_2 \models \underline{\Psi\{x \mapsto f(y_1, \ldots, y_n)\} \Rightarrow (\forall x)\Psi}$

dove J_1, J_2 *sono due interpretazioni che differiscono da* I *solo in* f.
In altre parole, la 1. *permette di eliminare* $(\exists x)$ *sostituendo* x *con una opportuna funzione (che dipende dalle variabili libere di* $(\exists x)\Psi$*). La* 2. *e' invece un procedimento inverso: possiamo rimuovere una funzione dipendente dalle variabili libere, al prezzo di introdurre il quantificatore* $(\forall x)$.

Proof:

⟨1⟩1. Dim. 1.

Consideriamo un $d_0 \in D$ qualsiasi. Sia A un assegnamento. Specifichiamo l'intepretazione J_1 su f:

$$f^{J_1}(d_1,\dots,d_n) = \begin{cases} x^B & v_{I,A}((\exists x)\Psi) = \mathbf{t},\ \text{con } B \in A_x :\ \ v_{I,B}(\Psi) = \mathbf{t} \\ d_0 & v_{I,A}((\exists x)\Psi) = \mathbf{f} \end{cases}$$

dove A e' un qualsiasi assegnamento tale che $(y_1^A,\dots,y_n^A) = (d_1,\dots,d_n)$

Per dimostrare

$$M_1 \models \underline{(\exists x)\Psi \Rightarrow \Psi\{x \mapsto f(y_1,\dots,y_n)\}}$$

supponiamo che $v_{J_1,A'}((\exists x)\Psi) = \mathbf{t}$ e dimostriamo che $v_{J_1,A'}(\Psi\{x \mapsto f(y_1,\dots,y_n)\}) = \mathbf{t}$.

$v_{J_1,A'}((\exists x)\Psi) = \mathbf{t} \underbrace{\Leftrightarrow}_{f\notin\mathcal{F}((\exists x)\Psi),\ J_1 \text{ e' uguale a } I \text{ a meno di } f} v_{I,A'}((\exists x)\Psi) = \mathbf{t} \underbrace{\Rightarrow}_{\text{per def. di } J_1} f^{J_1}(y_1^{A'},\dots,y_n^{A'}) = x^B \Leftrightarrow$

$\Leftrightarrow (f(y_1,\dots,y_n))^{J_1,A'} = x^B$, dove $B \in A'_x$, $v_{I,B}(\Psi) = \mathbf{t} \underbrace{\Leftrightarrow}_{f\notin\mathcal{F}(\Psi)} v_{J_1,B}(\Psi) = \mathbf{t}$

Quindi $v_{J_1,A'}(\Psi\{x \mapsto f(y_1,\dots,y_n)\}) = v_{J_1,B}(\Psi) = \mathbf{t}$

⟨1⟩2. Dim. 2.

Consideriamo la formula $\neg\Psi$. Per il passo 1. sappiamo che

$$\exists M_2 :\ M_2 \models \underline{(\exists x)\neg\Psi \Rightarrow \neg\Psi\{x \mapsto f(y_1,\dots,y_n)\}} \quad (1)$$

ma la (1) e' equivalente ad

$$M_2 \models \underline{\neg\neg\Psi\{x \mapsto f(y_1,\dots,y_n)\} \Rightarrow \neg(\exists x)\neg\Psi} \Leftrightarrow M_2 \models \underline{\Psi\{x \mapsto f(y_1,\dots,y_n)\} \Rightarrow (\forall x)\Psi}$$

□

Theorem 5.64. *di Skolemizzazione.*
Sia $a \in \mathrm{AT}$, $\Phi,\Psi \in \mathrm{Form}$, $\mathrm{Free}(\Psi) = \{x, y_1,\dots,y_n\}$. *Supponendo che* $\Phi(a|(\exists x)\Psi) \in \mathrm{CForm}$ *sia un enunciato e considerato* $f \in \mathcal{F}_n \backslash \mathcal{F}_n(\Phi(a|(\exists x)\Psi))$ *si ha*

1. $a \in \mathrm{AT}^+(\Phi) \Rightarrow \Big(\Phi(a|(\exists x)\Psi)$ *soddisf.* $\Leftrightarrow \Phi(a|\Psi\{x \mapsto f(y_1,\dots,y_n)\})$ *soddisf.* $\Big)$
2. $a \in \mathrm{AT}^-(\Phi) \Rightarrow \Big(\Phi(a|(\forall x)\Psi)$ *soddisf.* $\Leftrightarrow \Phi(a|\Psi\{x \mapsto f(y_1,\dots,y_n)\})$ *soddisf.* $\Big)$

La skolemizzazione permette di eliminare i quantificatori (nei casi opportuni) preservando la soddisfacibilita'. Questo e' utile per i sistemi di deduzione che cercano di dimostrare l'insoddisfacibilita' di $\neg X$, *per poter dimostrare che* $\models X$.
La procedura di skolemizzazione consiste nell'applicare la 1. *e* 2. *per rimuovere il maggior numero di quantificatori.*

Proof:

⟨1⟩1. Dim 1.

⟨2⟩1. Dim. $\Leftarrow$

Supponiamo che

$$\exists M :\ \ M \models \Phi(a|\Psi\{x \mapsto f(y_1,\dots,y_n)\}) \quad (0)$$

Si ha

$\models \underline{\Psi\{x \mapsto f(y_1,\dots,y_n)\} \Rightarrow (\exists x)\Psi}$ e' una formula valida (1)

(1), a occorre positivamente in $\Phi \underbrace{\Rightarrow}_{\text{sostituzione [5.62,pg.101]}} M \models \underline{\Phi(a|\Psi\{x \mapsto f(y_1,\dots,y_n)\}) \Rightarrow \Phi(a|(\exists x)\Psi)}$ (2)

(2), (0) $\Rightarrow M \models \Phi(a|(\exists x)\Psi)$

⟨2⟩2. Dim. $\Rightarrow$

Supponiamo che

$$M \models \Phi(a|(\exists x)\Psi) \quad (0)$$

Per il lemma [5.63,pg.102], si ha:

$\exists M_1 = (D, J_1)$ modello t.c. $M_1 \models \underline{(\exists x)\Psi \Rightarrow \Psi\{x \mapsto f(y_1, \ldots, y_n)\}} \Rightarrow$

$$\underbrace{\Rightarrow}_{\text{sostituzione [5.62,pg.101]}} \quad M_1 \models \underline{\Phi(a|(\exists x)\Psi) \Rightarrow \Phi(a|\Psi\{x \mapsto f(y_1, \ldots, y_n)\})} \quad (1)$$

$f \notin \mathcal{F}(\Phi(a|(\exists x)\Psi))$, M_1 uguale a M tranne che per f , $(0) \Rightarrow M_1 \models \Phi(a|(\exists x)\Psi)$ $(0')$

(1), $(0') \Rightarrow M_1 \models \Phi(a|\Psi\{x \mapsto f(y_1, \ldots, y_n)\})$

$\langle 1 \rangle 2$. Dim 2.

Si procede analogamente a prima, sfruttando pero' la 2. del lemma [5.63,pg.102] e la tautologia

$$\models \underline{(\forall x)\Psi \Rightarrow \Psi\{x \mapsto f(y_1, \ldots, y_n)\}}$$

e la 2. del thm [5.62,pg.101]

□

Example 5.65. di skolemizzazione.

1. $(\forall x)(\exists y)[(\exists z)(\forall w)R(x, y, z, w) \Rightarrow (\exists w)P(w)]$
2. $(\forall x)((\exists z)(\forall w)R(x, f(x), z, w) \Rightarrow (\exists w)P(w))$
3. $(\forall x)((\exists z)(\forall w)R(x, f(x), z, w) \Rightarrow P(h))$
4. $(\forall x)((\exists z)R(x, f(x), z, g(x, z)) \Rightarrow P(h))$

La 2. e' stata ottenuta eliminando $(\exists y)$: la sottoformula che seguiva $(\exists y)$ occorreva positivamente nella formula grande, quindi abbiamo potuto applicare la 1. di [5.64,pg.103].
La 3. e' stata ottenuta analogamente. Nota: abbiamo in questo caso $n = 0$ e la f del thm [5.64,pg.103] e' una costante.
La 4. e' stata ottenuta eleminando $(\forall w)$. Nota che $R(x, f(x), z, w)$ occorre negativamente nella formula perche' si trova nella prima componente della β-formula $\Rightarrow$, ovvero, e' come se avessimo

$$\neg((\exists z)(\forall w)R(x, f(x), z, w)) \vee P(h)$$

Dalla 4. non si puo' piu' proseguire.

Definition 5.66. formule in forma premessa.
Una formula X e' in forma *premessa*, quando e' del tipo

$$(Q_1 x_1)(Q_2 x_2) \ldots (Q_m x_m)Y$$

dove $Q_i \in \{\forall, \exists\}$ e Y e' una formula prima di quantificatori universali.
In altre parole, X e' in forma premessa quando tutti i quantificatori sono posti all'inizio.

Proposition 5.67. *Per ogni formula X esiste, e si puo' costruire, una formula X' in forma premessa, equivalente a X.*
I passi principali dell'algoritmo di conversione sono:

1. *Da X costruiamo la formula Y: per ottenere Y rinominiamo le variabili di X in modo che non ci siano due quantificatori con la stessa variabile e che ogni quantificatore utilizzi una variabile diversa da quelle libere presenti in X.*

2. Applichiamo a Y le seguenti regole (che rispettano l'equivalenza), fino ad ottenere X':

$$\neg(\exists x)A \Leftrightarrow (\forall x)\neg A$$
$$\neg(\forall x)A \Leftrightarrow (\exists x)\neg A$$
$$(\forall x)A \wedge B \Leftrightarrow (\forall x)(A \wedge B)$$
$$A \wedge (\forall x)B \Leftrightarrow (\forall x)(A \wedge B)$$
$$(\forall x)A \vee B \Leftrightarrow (\forall x)(A \vee B)$$
$$A \vee (\forall x)B \Leftrightarrow (\forall x)(A \vee B)$$
$$(\exists x)A \vee B \Leftrightarrow (\exists x)(A \vee B)$$
$$A \vee (\exists x)B \Leftrightarrow (\exists x)(A \vee B)$$
$$((\forall x)A \Rightarrow B) \Leftrightarrow ((\exists x)(A \Rightarrow B))$$
$$(A \Rightarrow (\forall x)B) \Leftrightarrow ((\forall x)(A \Rightarrow B))$$
$$((\exists x)A \Rightarrow B) \Leftrightarrow ((\forall x)(A \Rightarrow B))$$
$$(A \Rightarrow (\exists x)B) \Leftrightarrow ((\exists x)(A \Rightarrow B))$$

Nota: per ottenere quelle con $\Rightarrow$, si considera che $(A \Rightarrow B) \Leftrightarrow (\neg A \vee B)$.
Nota: queste regole devono essere applicate dopo il passo 1.. Quindi, nella formula $(\forall x)A \vee B$ la variabile x non puo' occorrere dentro B.

Example 5.68.

$$(\exists x)(\forall y)R(x,y) \Rightarrow (\forall y)(\exists x)R(x,y)$$
$$(\forall x)\left((\forall y)R(x,y) \Rightarrow (\forall y)(\exists x)R(x,y)\right)$$
$$(\forall x)(\exists y)\left(R(x,y) \Rightarrow (\forall y)(\exists x)R(x,y)\right)$$
$$(\forall x)(\exists y)(\forall y)\left(R(x,y) \Rightarrow (\exists x)R(x,y)\right)$$
$$(\forall x)(\exists y)(\forall y)(\exists x)\left(R(x,y) \Rightarrow R(x,y)\right)$$

Theorem 5.69. *di Herbrand costruttivo.*
Sia X un enunciato,

$$X \text{ e' valida } \Leftrightarrow \bigwedge_{i=1}^{l} Y(d_1^i,\ldots,d_n^i) \text{ e' insoddisfacibile}$$

dove Y e' una formula chiusa priva di quantificatori, e quindi e' sostanzialmente una formula proposizionale.
Il thm di Herbrand permette di risolvere la validita' di X, esaminando la soddisfacibilita' di Y.
Il problema principale e' quello di trovare Y.

Proof: Diamo solo una bozza della dimostrazione:
X e' valida $\Leftrightarrow \neg X$ insoddisfacibile $\Leftrightarrow \Gamma$ insoddisfacibile (1)

dove Γ e' $\neg X$ convertita in forma premessa e skolemizzata : $\Gamma = (\forall y_1)(\forall y_2)\ldots(\forall y_n)Y$

(1) $\underbrace{\Leftrightarrow}_{[5.35,\text{pg.}83]}$ $Z = \{Y(d_1,\ldots,d_n) \mid d_i \in \text{CTerm}\}$ insoddisfacibile $\Leftrightarrow$

$\underbrace{\Leftrightarrow}_{\text{thm compattezza}}$ $\exists Z' \subseteq Z$ finito e insoddisfacibile , $Z' = \left\{Y(d_1^i,\ldots,d_n^i) \mid i = 1,\ldots,l\right\}$ $\Leftrightarrow$

$\Leftrightarrow \bigwedge_{i=1}^{l} Y(d_1^i,\ldots,d_n^i)$ insoddisfacibile

□

Definition 5.70. Tableaux skolemizzato.
Utilizzando la skolemizzazione e le sostituzioni e' possibile definire delle nuove regole che sostituiscono quelle date in [5.39,pg.89]:

$6'$. **Se** $N = \underline{\gamma}$:

$r := r, [\gamma(x)]$ con x variabile che non e' gia' utilizzata nel tableaux come variabile legata

$7'$. **Se** $N = \underline{\delta}$:

$r : r, [\delta(f(x_1, \ldots, x_n))]$ con $f \in \mathcal{F}_n$ simbolo funzionale nuovo per il tableaux, $x_1, \ldots, x_n$ variabili libere di δ

I simboli funzionali nuovi li prendiamo da un insieme Sko che svolge lo stesso ruolo di Par.
Viene aggiunta inoltre la regola della *sostituzione*: sia T un tableaux e sia σ una sostituzione libera per T, allora

Il tableaux $T\sigma$ e' ottenuto applicando la sostituzione σ a tutte le formule del tableaux T

Questa regola serve per sostituire le variabili introdotte con la $6'$. con opportuni termini, nella speranza di chiudere il tableaux. Infatti, per chiudere il tableaux si cercano due formule $A, \neg B$ e una sostituzione che unifica A e B, ovvero:

$$A\sigma = B\sigma$$

cosi' facendo si potra' chiudere il ramo di $T\sigma$:

$$A\sigma$$
$$\neg(A\sigma)$$

Nota: in un'implementazione meccanica, σ si puo' costruire applicando l'algoritmo di unificazione [5.16,pg.73].

Example 5.71. Dimostriamo la seguente formula con i tableaux skolemizzati

1. $\neg\left[(\exists w)(\forall x)R(x, w, f(x, w)) \Rightarrow (\exists w)(\forall x)(\exists y)R(x, w, y)\right]$
2. $(\exists w)(\forall x)R(x, w, f(x, w))$
3. $\neg\left((\exists w)(\forall x)(\exists y)R(x, w, y)\right)$
4. $(\forall x)R(x, c_1, f(x, c_1))$
5. $\neg(\forall x)(\exists y)R(x, x_1, y)$
6. $\neg(\exists y)R(g(x_1), x_1, y)$
7. $R(x_2, c_1, f(x_2, c_1))$
8. $\neg R(g(x_1), x_1, x_3)$

2., 3. da 1. con la regola per le α-formule
4. da 2. con la regola $7'$.. Nota: c_1 e' una nuova funzione, 0-aria, ovvero una costante di Sko.
5. da 3. con la regola $6'$.. Nota: x' e' una nuova variabile
6. da 5. con la $7'$.
7. da 4. con la regola $6'$.
8. da 6. con la regola $6'$.

A questo punto utilizzando la sostituzione $\sigma = \{x_1 \mapsto c_1,\ x_2 \mapsto g(c_1),\ x_3 \mapsto f(g(c_1), c_1)\}$ otteniamo un nuovo tableaux che finisce con:

7. $R(g(c_1), c_1, f(g(c_1), c_1))$
8. $\neg R(g(c_1), c_1, f(g(c_1), c_1))$

ed e' quindi chiuso.

6 *

Index

*, 106

albero binario, 3
albero sintattico, 68

CNF e DNF, 19
Compattezza, 42
compattezza, 42
compattezza I ordine, 98
Completezza, 29
Completezza semantica, 93
congiunzioni letterali, 19
Conjunctive Normal Form, 19
connettivi primari, 15
Conseguenza proposizionale, 48
coppia di disaccordo, 74
costanti, 68

Deduzione naturale I ordine, 92
disgiunzioni letterali, 19
Disjunctive Normal Form, 19

enunciati, 69

Forme normali, 15
formule atomiche, 8
funzione induttiva, 6

Generalizzazione Universale, 90

Hilbert del I ordine, 90
Hintikka, 29
Hintikka I ordine, 93

Induzione, 1
    strutturale, 5
insieme indotto, 2
insieme libero, 5
interpretazione, 77
Introduzione, 1
invariante, 21
istanza, 84

letterali, 19
libera, 69
Logica del I ordine, 67
Logica proposizionale, 7

modello di Herbrand, 80
Modus Ponens, 50

Notazione uniforme, 84

operatore duale, 13

Prefazione, 1
premessa, 103
Procedura di Davis Putnam, 62
proprieta' di consistenza, 30
proprieta' di consistenza I ordine, 95

rango, 18
regole di espansione, 27
regole risolutive, 27
Risoluzioni, 26
Risoluzioni I ordine, 89
Risoluzioni ristrette, 43
Robinson, 47

Semantica, 77
sequenza di costruzione, 4
Sistema di deduzione naturale, 56
Sistemi di deduzione, 87
Sistemi di Hilbert, 49
Skolemizzazione, 99
sostituzione, 69
sostituzione libera, 71

Tableaux, 22
tableaux, 22
Tableaux I ordine, 87
termini, 68
Termini chiusi, 78
thm di Herbrand, 98

Unificazione concorrente, 76
Unificazione multipla, 76

valutazione booleana, 9
variabile legata, 69