# Logica Matematica

Lecture Notes

**Corso del Prof. Stefano Aguzzoli**

Edoardo Marangoni

University of Milan
Department of Computer Science
11 febbraio 2021

# Indice

# Disclaimer

Questo documento contiene degli appunti presi durante il corso tenutosi nell'A.A. 2020/2021. Benché i materiali si attengono a ciò che è stato spiegato nel corso, queste note **non sono in alcun modo approvate dal docente**, con tutte le conseguenze implicate.

CAPITOLO 1

# Introduzione

La Logica è una materia che ha una tradizione millenaria e trae le sue origini in ambito filosofico: la definizione che vedremo noi è infatti presa da quell'ambito, e noi la declineremo in una forma moderna. La Logica è lo studio dei meccanismi del ragionamento razionale. In altre parole, si intende lo studio della capacità di trarre conseguenze (corrette) da date assunzioni o premesse. Le assunzioni sono delle informazioni che affermano che il mondo sta in un certo modo e sono gestibili formalizzandole in un linguaggio formale. Queste informazioni codificano uno o più mondi possibili: vogliamo trarne delle conclusioni a partire di esse in un modo razionale.

## 1.1 Motivazioni

La Logica studia come si ragiona in maniera corretta e, per studiare come si ragione, si può utilizzare come prima schematizzazione il partire da delle assunzioni vere e da quelle discendere a delle conclusioni. Un esempio: ogni uomo è mortale. Socrate è un uomo. Dunque, Socrate è mortale. Questo è, in linguaggio naturale, un esempio di quanto detto prima: a partire da due informazioni date per vere (ogni uomo è mortale e Socrate è un uomo) si traggono delle conseguenze. Quanto fatto prima è un *sillogismo*, un punto antichissimo nella storia della Logica.

Altro esempio: ogni gatto ha sette zampe. Pluto è un gatto. Dunque, Pluto ha sette zampe. Anche questa è una deduzione esatta, nonostante per l'esperienza comune la prima assunzione è falsa; tuttavia, la Logica si occupa di *ogni* universo e pertanto il ragionamento è valido. Ogni Blabla è glug. Sbappo è Blabla. Dunque, Sbappo è glug. Questa forma di ragionamento è altrettanto corretta. Per non farsi distrarre dalle stranezze irrilevanti, si utilizza un linguaggio centrale per il discorso della Logica. Si formalizza quindi questo ragionamento: in primo luogo si astrae, fornendo un modello matematico per ragionare.

$$(\forall x P(x) \to Q(x) \land P(s)) \to Q(s)$$

Ogni fiore è profumato. La Rosa è profumata. Dunque, la Rosa è un fiore. Per mostrare che questo ragionamento è falso, si può anche utilizzare l'intuizione: se Rosa è mia nonna, benché il senso metaforico sia valido, Rosa è un po' vecchia e pertanto il ragionamento non è valido. In che modo è cambiato il ragionamento?

$$(\forall x P(x) \to Q(x) \land Q(s)) \nrightarrow P(s)$$

In questo caso, si sta cercando di verificare la premessa data la conclusione, al contrario di quanto accadeva per il sillogismo aristotelico; questo modo di ragionare non può funzionare.

**Matematica** Vi sono almeno due sensi per cui la Logica è matematica. Il primo è quello che abbiamo introdotto immediatamente al discorso iniziale: la matematica è utilizzata per la necessità di *astrarre* solamente le informazioni rilevanti scartando il resto in un contesto con molte informazioni che non ci interessano, come accade per il linguaggio naturale. La trasformazione delle assunzioni e delle conclusioni da una forma in linguaggio naturale alla forma astratta permette di arrivare a delle **forme**. I concetti che andremo a formalizzare avranno una sintassi dettata da un **linguaggio formale** e un significato semantico **algebrico-insiemistico**, ottenendo un **formalismo**, un modo preciso, rigoroso e privo di ambiguità per esprimere ciò che si vuole esprimere in Logica.

In seconda battuta, la Logica si usa per studiare le strutture matematiche, ossia si usa *per fare* matematica. Aprendo un testo qualsiasi di Logica matematica si vedrà come gli esempi più interessanti siano basati sulla matematica: gruppi, campi e teoremi vari.

Una terza parola chiave, che conclude la parte motivazionale, è **Informatica**, intesa come *Computer Science*, inteso come capire il processo dei sistemi computazionali. È stata infatti una grande rivincita della Logica durante il secolo scorso, che ha visto nascere i fondamenti della computazione partendo da strumenti logici. Se esiste, la differenza tra *Logica* e *Informatica* sono i focus diversi: la prima è più vicina ad un approccio dichiarativo, concentrandosi su ciò che si può concludere da determinate premesse, mentre la seconda è più vicina ad approcci procedurali o imperativi.

**Esercizio** Se piove prendo l'ombrello. È lo stesso caso di dire che:

1. Se non piove non prendo l'ombrello
2. Se non prendo l'ombrello non piove
3. Se prendo l'ombrello allora piove
4. O non piove o prendo l'ombrello
5. Piove solo se prendo l'ombrello
6. Se prendo l'ombrello piove
7. Piove se e solo se prendo l'ombrello
8. Nessuna delle precedenti

Benché non si sappia cosa voglia dire "lo stesso caso", tentiamo di dare le soluzioni a questo problema. Nel linguaggio naturale non si può fare a meno di sentire una dinamica: si vede che piove e allora si prende l'ombrello e si esce. La logica proposizionale non vede questa dinamicità: per farlo si devono elaborare formule che esplicitano la dinamicità. Una definizione più precisa di cosa voglia dire che due "frasi" siano "uguali": esse sono "equivalenti" quando sono vere nelle medesime circostanze. Questa è nuovamente una definizione che pecca di precisione in quanto non espressa matematicamente. Cosa vuol dire "medesime" e "circostanze"? Un'interpretazione intuitiva che mette in luce la "circostanza" della frase "Se piove prendo l'ombrello" è la seguente. Vi è una dipendenza tra il fatto che *piove* e il fatto di *prendere l'ombrello*.

In tutte le circostanze possibili, vi sono delle situazioni in cui è vero che piove e delle situazioni in cui non è vero e analogamente accade per il fatto di prendere l'ombrello. Di tutte le possibili circostanze ci interessano solo quattro: quando non piove e quando non si prende l'ombrello, quando piove e si prende l'ombrello, quando piove e non si prende l'ombrello e, infine, quando non piove e si prende l'ombrello.

La frase si può dunque rappresentare nella forma

$$P \Rightarrow Q$$

che rappresenta il *se...allora*. L'implicazione è infatti la più difficile da accettare a livello intuitivo. Senz'altro ci sono delle situazioni in cui non abbiamo dubbi: per esempio, quando sia $P$ e $Q$ sono vere, cioè, sapendo che piove e che si prende l'ombrello la relazione causale sembra sussistere e quindi $P \Rightarrow Q$ è verificata; quando $P$ è vero e $Q$ è falso, risulta infine che $P \Rightarrow Q$ è falsa. Ora, potrebbe succedere che la risposta non sia esattamente quella che ci aspettiamo: cominciamo assumendo che non piova ma si prenda l'ombrello. Risulta che $P \Rightarrow Q$ è vera, anche se l'antecedente è falso: già questa cosa può suonare strano, in quanto non suona giusto che il fatto che non piova implichi il fatto che si prenda l'ombrello. La questione riguarda sostanzialmente il linguaggio naturale di per sé e torneremo su questo discorso in futuro: con lo stesso approccio, si arriva a dire che se $P$ e $Q$ sono false allora $P \Rightarrow Q$ è vera.

Allora, per concludere il nostro esempio: si può cominciare dicendo che l'ottava frase è falsa, ossia vi sono, tra le prime sette frasi, alcune frasi equivalenti. La prima è sbagliata, in quanto vi è la possibilità di prendere l'ombrello anche se non piove. Il rapporto di causalità si rivede anche nella terza frase, che è falsa. La quarta frase necessità l'analisi della disgiunzione, l'OR, che in questa situazione va interpretato come un OR inclusivo, ossia un $\vee$. Analizzando la frase "O non piove o prendo l'ombrello" ci si accorge come si possa tradurre in $\neg P \vee Q$, che ha la stessa "immagine di verità" dell'implicazione, pertanto anche la quarta è uguale. Questo è importante in quanto è una realizzazione materiale dell'implicazione! La quinta frase si può nuovamente interpretare come $P \Rightarrow Q$ ed è pertanto vera. La sesta frase inverte il rapporto, facendo in modo che $Q \Rightarrow P$, che è falso; analogamente accade per la settima.

Un'ulteriore interpretazione dell'implicazione è: Ogni qualvolta $P$ è vera, anche $Q$ è vera.

## 1.2 Programma

Il corso tratterà inizialmente la Logica Proposizionale, mentre la seconda parte tratterà la Logica Predicativa o del Prim'ordine. Della Logica Proposizionale si definirà la sintassi, quindi Alfabeto, Connettivi e Formule per poi parlare di valutazione, tabelle di verità e princìpi come verofunzionalità e bivalenza. Si discuteranno le tautologie, le contraddizioni, le formule soddisfacibili e la nozione centrale di Conseguenza logica. Seguentemente si tratteranno decidibilità, correttezza e completezza, ma in realtà il primo Teorema che tratteremo sarà quello di Compattezza. Dopodiché si potranno affrontare i metodi formali di deduzione per la Logica Proposizionale. Accenneremo ai Seguenti e ai Tableau, oltre che ai calcoli alla Hilbert e i metodi assiomatici. Ci concentreremo sulla metodologia più adatta alla deduzione automatica, ossia i metodi refutazionali basati sul principio di risoluzione.

La seconda parte del corso tratterà la Logica dei Predicati, che per noi sarà un sinonimo di Logica del Prim'ordine. Dal punto di vista sintattico, si affronteranno più approfonditamente alfabeto, quantificatori, simboli di predicato e i simboli di funzione. Seguirà la semantica: descriveremo la semantica di Tarski, con la nozione fondamentale di L-Struttura e modelli; il concetto di Conseguenza logica,completezza e correttezza nell'ambito della Logica del Prim'ordine. Termineremo con i metodi di deduzione e le forme normali. Assieme alla Teoria di Herbrand, quest'ultime ci permetteranno di arrivare alle tecniche di deduzione automatica.

# Parte I

# Logica Proposizionale

CAPITOLO 2

# Introduzione alla Logica Proposizionale

Si comincia ora l'introduzione alla Logica Proposizionale, partendo tuttavia da un concetto espresso senza formalizzarlo immediatamente:

**Definizione** (Enunciato)**.** *Con il termine* ***enunciato*** *si intende una frase o un'espressione per la quale sia sensato chiedersi se sia vera o se sia falsa in ogni data circostanza, ossia ha un* ***valore di verità*** *relativo ad una certa circostanza.*

"Piove" è un enunciato, così come "prendo l'ombrello" e "se piove prendo l'ombrello". Sapremmo già dire che quest'ultimo ha qualcosa di diverso dai primi: quest'ultimo infatti è un **enunciato composto**, mentre i primi sono **enunciati atomici**. Ci sono frasi che non sono enunciati e possiamo anche limitarci all'italiano per trovarne alcuni: "Paolo corre?" e "Piove?" non sono enunciati. Oltre al linguaggio naturale vi sono anche altre frasi che non sono enunciati, per esempio "2".

## 2.1 Senso, denotazione e connotazione di un enunciato

Il senso filosofico dei concetti di **denotazione** e **connotazione** verrà tralasciato e verranno infatti trattati in una maniera poco profonda, soprattutto per capire la distinzione tra denotazione e connotazione. Ecco alcune espressioni del linguaggio dell'aritmetica:

- $4$
- $2^2$
- il predecessore di $5$
- $3+1$

Nessuno di questi è un enunciato, ma non è necessario che lo siano. Sappiamo dire cosa significhino, in quanto matematicamente sono sempre modi per esprimere *il numero naturale quattro*. Questo esempio inquadra a livello intuitivo cosa sia la **denotazione** (il numero naturale quattro) e la **connotazione** (quattro diversi modi per ottenere quattro). Anche a questo livello stiamo dicendo una cosa interessante, in quanto questo implica che dobbiamo essere molto precisi riguardo cosa dovrebbe essere la *denotazione* di qualcosa. Un'espressione ha, quindi, una denotazione che è qualcosa di diverso dalla sua connotazione. In un modo astratto, una espressione $E$ è un *nome* di qualcosa, come per esempio $4$, $2^2$, il predecessore di $5$ e $3+1$, il quale si riferisce in modo univoco a qualche entità. L'*entità* alla quale si riferisce l'espressione è essa stessa la denotazione. La connotazione, in questo senso, è quanto l'espressione effettivamente esprime, ossia tutto il resto dell'informazione contenuta nell'espressione stessa.

La logica studia la denotazione degli enunciati (chiamati anche *sentences*) e non le connotazioni, in quanto esse sono troppo difficili da gestire a livello iniziale. Le denotazioni godono infatti dell'importante proprietà dell'**invarianza per sostituzione**, ossia se ad un'espressione si cambiano delle parti sostituendole con parti denotazionalmente uguali, la denotazione globale non cambia, mentre la connotazione può potenzialmente cambiare totalmente, come dimostrano le quattro frasi iniziali.

### 2.1.1 Denotazione di un Enunciato

Si può definire ora, più formalmente, cosa sia la **denotazione** di un enunciato. Si prenda, per esempio, l'enunciato

$$4 = \text{pred}(5)$$

e si applichi una sostituzione con espressioni denotazionalmente equivalenti:

$$4 = 4$$

Il principio d'invarianza dice che questi due enunciati sono **denotazionalmente** equivalenti, benché l'ultimo enunciato non contiene nessuna informazione ulteriore rispetto alla denotazione stessa (circa). La denotazione di un enunciato è, quindi, il loro valore di verità, ossia il fatto che sono una forma connotazionale di una costante `vero` o `falso`. Quindi, l'oggetto della Logica Proposizionale sono le **proposizioni**, ossia il contenuto denotazionale degli enunciati, che può essere `vero` o `falso`.

## 2.2 Enunciati e Connettivi

Alcuni enunciati semplici come "Piove" o "Paolo corre" sono definiti **atomici** in quanto la loro denotazione è solamente un valore di verità. Altri enunciati, definiti **composti**, sono enunciati che si possono "smontare", come "Piove e c'è vento", che è chiaramente composto dagli enunciati atomici "Piove" e "c'è vento". Il connettivo "e" è ciò che li unisce, come potrebbe accadere anche per "o", "non" e "se...allora".

In una maniera più formale, gli enunciati semplici devono solamente rappresentare il fatto che denotano un valore di verità e saranno quindi rappresentati da **simboli** appartenente all'insieme infinito $L$ chiamato **linguaggio proposizionale**. I simboli $p, q, r, p_1, \cdots \in L$ sono chiamati **lettere proposizionali**. Oltre alla sintassi, formalmente il valore semantico (quindi denotazionale) di ogni lettera proposizionale è un valore di verità.

Gli enunciati composti sono formalizzabili con una simbologia che rispetta i simboli per gli enunciati atomici: $p \wedge q$, $p \vee q$, $\neg p$ e $p \rightarrow q$ sono enunciati composti. I simboli $\wedge, \vee, \neg$ e $\rightarrow$ sono chiamati **connettivi**. Il valore denotazionale di ogni enunciato composto dipenderà dai valori denotazionali degli enunciati atomici e dal valore semantico dei connettivi che lo compongono.

CAPITOLO 3

# Sintassi della Logica Proposizionale

Dopo aver introdotto la Logica Proposizionale, si può ora formalizzare la struttura sintattica degli enunciati: l'insieme $F_L$ degli enunciati costruibili sul linguaggio $L$ rispettando la **sintassi degli enunciati** è definito come segue:

**Definizione.** *(Sintassi degli Enunciati)*

- *$F_L$ è il più piccolo insieme tale che*
  - *per ogni $p \in L$ si ha $p \in F_L$*
  - *se $A, B \in F_L$ allora anche $(A \wedge B) \in F_L$, $(A \vee B) \in F_L$, $(A \to B) \in F_L$ e $(\neg A) \in F_L$, dove A e B possono essere a loro volta enunciati complessi.*
- *$F_L$ è l'intersezione di tutti gli insiemi X tali che*
  - *$L \subseteq X$*
  - *se $A, B \in X$ allora $(A \wedge B)$, $(A \vee B)$, $(\neg A)$ e $(A \to B)$ sono contenuti in X.*
- *(induttiva) $F_L$ è l'insieme che rispetta le condizioni seguenti:*
  - *$L \subseteq F_L$: se $p \in L$, allora $p \in F_L$*
  - *Se $A, B \in F_L$, allora $(A \wedge B)$, $(A \vee B)$, $(\neg A)$ e $(A \to B)$ sono contenuti in $F_L$*
  - *Nient'altro appartiene a $F_L$.*

**Esercizio** Scrivere qualcosa che non sia un enunciato utilizzando solamente la sintassi della logica proposizionale. Un esempio è $pq\neg$. Un altro è $\to q$.

## 3.1 L-Costruzioni

Per stabilire se una stringa $w \in (\{\wedge, \vee, \neg, \to\} \cup L)^*$[1] è anche $w \in F_L$ si utilizza una **L-costruzione** o **certificato**, ossia una sequenza finita di formule $w_1, w_2, \cdots, w_n$ tale che per ogni $i = 1, \cdots, n$ si ha $w_i \in L$, ossia $w$ è una lettera proposizionale, oppure esiste $j < i$ tale che $w_i = \neg w_j$, ossia $w_i$ è la forma negata di una formula più semplice che appare prima nell'elenco di formule; oppure

[1] L'utilizzo dell'operatore $^*$ è paragonabile all'operatore Stella di Kleene nella Teoria dei Linguaggi e denota l'insieme di tutte le stringhe di lunghezza finita componibili utilizzando le lettere dell'alfabeto indicato, in questo caso i connettivi e le lettere proposizionali.

esistono $k, j < i$ tali per cui $w_i = w_k \wedge w_j$ o $w_i = w_k \vee w_j$ o $w_i = w_k \to w_j$. Sia, per esempio $w = (p \wedge q) \to r$. Per dimostrare che è $w \in F_L$ si possono introdurre come L-costruzione

$$w_1 = p, w_2 = q, w_3 = (p \wedge q), w_4 = r, w_5 = (p \wedge q) \to r$$

questa L-costruzione *certifica* che $w \in F_L$. In termini computazionali, si può controllare velocemente che quanto fatto sopra sia una costruzione corretta e che pertanto $w \in F_L$. Si noti, inoltre, che questa L-costruzione non è unica. I certificati portano con sé una proprietà importante riguardo le formule, ossia

**Proprietà** (di unica leggibilità dei Certificati)**.** *Se $w \in F_L$, allora vale uno e uno solo dei seguenti casi: o $w \in L$, o esiste $v$ tale che $w = \neg v$, o esistono $v_1, v_2$ tali che $w = (v_1 \wedge v_2)$, $(v_1 \vee v_2)$ o $(v_1 \to v_2)$, dove i $v_i$ sono determinati univocamente.*

Questa proprietà garantisce l'esistenza di una sorta di operazione inversa della costruzione di certificati: mentre quest'ultimo "monta" una formula, questa proprietà garantisce il fatto che sia possibile "smontarla" in un unico modo. Questa proprietà non è condivisa tra tutti i linguaggi formali: per esempio nelle grammatiche una stringa $w = ciao$ può essere composta da $v_1 = \varepsilon$ e $v_2 = ciao$ eccetera. Spesso, il concetto di leggibilità può essere espresso anche tramite il concetto di **albero di parsing** di una formula, che è unico e mostra come essa sia costruita.

## 3.2 Osservazioni e convenzioni riguardo alla sintassi

Esistono ulteriori connettivi (o connettivi derivati) oltre a quelli utilizzati fino ad ora, ossia $\wedge$, $\vee$, $\neg$ e $\to$: uno di questi è $\bot$, che è un connettivo di arità zero che denota il falso; un altro è $\top$, che è un connettivo zerario che denota il sempre vero e chiaramente $\bot = \neg\top$. Altro connettivo è $\iff$, definito come $(A \to B) \wedge (B \to A)$.

Una seconda osservazione riguarda l'uso delle parentesi: per come abbiamo definito le formule, l'oggetto $p \wedge q \notin F_L$ in quando mancano le parentesi. In formule complesse, le parentesi possono aggiungere complicatezza e portare l'errore: useremo il buonsenso per "dimenticarci" delle parentesi laddove non ci sia pericolo di confusione. Tuttavia non bisogna farsi trasportare troppo, in quanto esistono delle parentesi necessarie, come per esempio $(p \wedge q) \vee r$ oppure $p \wedge (q \vee r)$.

CAPITOLO 4

# Semantica della Logica Proposizionale Classica

## 4.1 Fondamenti della Semantica

Per dare una definizione di semantica si utilizzano due princìpi guida: il **principio di bivalenza** e il **principio di verofunzionalità**.

**Principio** (Bivalenza)**.** *Un enunciato, in ogni circostanza, è o vero o falso.*

Il principio di bivalenza ha come conseguenza il principio del terzo escluso.

**Principio** (Verofunzionalità, composizionalità o estensibilità)**.** *Il valore di verità di un enunciato composto dipende solo dal valore di verità degli enunciati che lo compongono e dal significato del connettivo che li unisce.*

Dato il principio di verosimiglianza, per dare la semantica ad un connettivo come $\wedge$ si deve dire qual è, sotto ogni circostanza, il valore di verità di $A \wedge B$, il quale può essere `vero` o `falso` e può dipendere solo dal valore di verità di $A$ e $B$ e dal significato fissato per $\wedge$. È necessario, ora, definire informalmente cosa sia una *circostanza*: una **circostanza** è un assegnamento che definisce lo stato vero o falso di una lettera proposizionale (o di una formula), definita come una funzione $\mathfrak{v}$:

$$\mathfrak{v} : L \to \{0,1\}$$

Quindi $\mathfrak{v}(A \wedge B) \in \{0,1\}$ e, grazie al principio di Verofunzionalità, dipende solo da $\mathfrak{v}(A)$, $\mathfrak{v}(B)$ e dal significato fissato per $\wedge$, definito come

$$I_\wedge : \{0,1\}^2 : \{0,1\}$$

E si ha, quindi

$$\mathfrak{v}(A \wedge B) = I_\wedge(\mathfrak{v}(A), \mathfrak{v}(B))$$

**Importanza della Verofunzionalità** Si supponga per un attimo che invece di Logica si stia studiando Probabilità, tentando di formalizzarla come stiamo formalizzando ora la Logica Proposizionale. Siamo in una circostanza in cui un certo evento ha probabilità $p(A) = \frac{1}{2}$ e un altro evento ha probabilità $p(B) = \frac{1}{2}$. La probabilità $p(A \to A) = 1$ rappresenta l'evento certo. Qual è la probabilità $p(A \to B)$? Se fossimo in una situazione verofunzionale, questa probabilità dovrebbe essere 1, in quanto se $p(\frac{1}{2} \to \frac{1}{2}) = 1$, allora anche per $p(A \to B) = 1$. Per concretezza, si immagini $A =$ domani piove e $B =$ a Pasqua nevica. In conclusione, la Verofunzionalità è una caratteristica stringente della Logica Proposizionale da non dare affatto per scontata.

### 4.1.1 Definizione di Semantica

Siamo finalmente pronti per dare una definizione formale della **semantica** della Logica Proposizionale:

**Definizione** (Semantica della Logica Proposizionale)**.** *Un assegnamento (o valutazione) è un'arbitraria funzione*

$$\mathfrak{v} : L \to \{0,1\}$$

che formalizza la nozione intuitiva di circostanza (o *mondo possibile*).

**Definizione** (Semantica dei connettivi)**.** *La **semantica dei connettivi** è espressa tramite delle funzioni:*

$$I_\wedge : \{0,1\}^2 : \{0,1\}$$

che identificano una *tabella di verità*.

**Definizione** (Semantica degli Enunciati)**.** *La **semantica degli enunciati** è l'**estensione canonica***

$$\widetilde{\mathfrak{v}} : F_L \to \{0,1\}$$

*ossia l'estensione di $\mathfrak{v}$ a tutte le formule, definita in questo modo:*

- $\widetilde{\mathfrak{v}}(p) = \mathfrak{v}(p)$ *se* $p \in L$
- $\widetilde{\mathfrak{v}}(A \wedge B) = I_\wedge(\widetilde{\mathfrak{v}}(A), \widetilde{\mathfrak{v}}(B))$ *se* $p \in F_L$
- $\widetilde{\mathfrak{v}}(A \vee B) = I_\vee(\widetilde{\mathfrak{v}}(A), \widetilde{\mathfrak{v}}(B))$*se* $p \in F_L$
- $\widetilde{\mathfrak{v}}(A \to B) = I_\to(\widetilde{\mathfrak{v}}(A), \widetilde{\mathfrak{v}}(B))$ *se* $p \in F_L$
- $\widetilde{\mathfrak{v}}(\neg A) = I_\neg(\widetilde{\mathfrak{v}}(A))$ *se* $p \in F_L$

Vi sono inoltre delle forme algebriche per esprimere i valori di verità dei connettivi, per esempio

$$\widetilde{\mathfrak{v}}(A \to B) = min\{1 - \widetilde{\mathfrak{v}}(A), \widetilde{\mathfrak{v}}(B)\}$$

Con un poco importante abuso notazionale, si tenderà ad evitare di esplicitare $\widetilde{\mathfrak{v}}$ in favore della notazione $\mathfrak{v}$.

## 4.2 Nozioni Semantiche Fondamentali

**Definizione** (Tautologia)**.** *Una formula $F \in F_L$ è una tautologia se e solo se*

$$\mathfrak{v}(F) = 1 \forall \mathfrak{v} : F_L \to \{0,1\}$$

**Definizione** (Formula Soddisfacibile)**.** *Una formula $F \in F_L$ è soddisfacibile se e solo se*

$$\exists \mathfrak{v} : F_L \to \{0,1\} : \mathfrak{v}(F) = 1$$

**Definizione** (Contraddizione)**.** *Una formula $F \in F_L$ è una contraddizione (o insoddisfacibile, refutabile) se e solo se*

$$\mathfrak{v}(F) = 0 \forall \mathfrak{v} : F_L \to \{0,1\}$$

C'è un fatto molto semplice che lega tra di loro questi concetti:

**Teorema.** *$F \in F_L$ è una tautologia se e solo se $\neg F$ è insoddisfacibile.*

*Dimostrazione.* $F$ è tautologica se e solo se $\widetilde{\mathfrak{v}}(F) = 1$ per ogni $\mathfrak{v} : L \in \{0,1\}$, ossia se e solo se $\widetilde{\mathfrak{v}}(\neg F) = 0$ per ogni $\mathfrak{v} : L \in \{0,1\}$, ossia $\neg F$ è insoddisfacibile. □

### 4.2.1 Principio d'Induzione su $F_L$

Benché non sia una sfaccettatura prettamente semantica, in quanto verrà (anzi, è già stato) utilizzato abbondantemente, è importante formalizzare il **principio d'induzione**. Sia $P$ una proprietà delle formule; si ha che $P$ vale per ogni formula $F \in F_L$ se e solo se:

1. **(base):** $P$ vale per ogni $p \in L$
2. **(passo induttivo)**: se $P$ vale per $A, B \in F_L$, allora vale anche per $\neg A$, $A \to B$, $A \vee B$ e $A \wedge B$.

Si può dimostrare induttivamente una proprietà anche grazie alla definizione induttiva di $F_L$. Sia

$$I = \{F \in F_L : \text{ la proprietà } P \text{ vale per } F\}$$

vogliamo mostrare $I = F_L$, ossia che $I$ è esattamente l'insieme di tutte le formule; questo si dimostra mostrando che $I \subseteq F_L$ e che $F_L \subseteq I$. È ovvio, per definizione, che $I \subseteq F_L$ in quanto è definito come un sottoinseme di formule. Per dimostrare il contrario, si dimostra che $I$ soddisfa il primo punto della definizione induttiva di $F_L$, ossia contiene tutte le lettere proposizionali (dalla base induttiva) e anche il secondo, in quanto se $A, B \in I$ allora anche $(\neg A)$, $(A \to B)$, $(A \wedge B)$ e $(A \vee B)$. Grazie al terzo punto della definizione di $F_L$, si può inoltre affermare che in quanto non sono contenute altre formule in $I$ (per definizione) $I = F_L$.

Come esempio, si dimostra induttivamente il seguente lemma:

**Lemma** (Verità di una Formula)**.** *Sia $F \in F_L$ e siano $\mathfrak{v}, \mathfrak{v}' : L \to \{0,1\}$.*
*Se $\mathfrak{v}(p) = \mathfrak{v}'(p) \forall p \in L \in F$[1], allora $\widetilde{\mathfrak{v}}(F) = \widetilde{\mathfrak{v}}'(F)$.*

*Dimostrazione. Per induzione su $F_L$.*

- **base:** Se $F = p \in L$ allora per ipotesi $\mathfrak{v}(p) = \mathfrak{v}'(p)$ e anche $\widetilde{\mathfrak{v}}(p) = \widetilde{\mathfrak{v}}'(p)$.
- Se $F = \neg A$ allora per ipotesi induttiva $\widetilde{\mathfrak{v}}(p) = \widetilde{\mathfrak{v}}'(p)$ e $\widetilde{\mathfrak{v}}(F) = 1 - \widetilde{\mathfrak{v}}(A) = 1 - \widetilde{\mathfrak{v}}'(A) = \widetilde{\mathfrak{v}}'F$
- Se $F = (A \wedge B)$ per ipotesi induttiva $\widetilde{\mathfrak{v}}(A) = \widetilde{\mathfrak{v}}'(A)$ e $\widetilde{\mathfrak{v}}(B) = \widetilde{\mathfrak{v}}'(B)$ e pertanto $\widetilde{\mathfrak{v}}(F) = \min\{\widetilde{\mathfrak{v}}(A), \widetilde{\mathfrak{v}}(B)\} = \min\{\widetilde{\mathfrak{v}}'(A), \widetilde{\mathfrak{v}}'(B)\} = \widetilde{\mathfrak{v}}'(F)$.
- uguale per gli altri connettivi.

□

Il lemma appena provato ci garantisce che se vogliamo calcolare $\widetilde{\mathfrak{v}}(F)$ ci basta calcolare la tabella di verità di $F$. A questo punto si può calcolare, per ogni assegnamento, se una formula è vera, soddisfacibile, tautologica o insoddisfacibile.

**Esercizio** Date $A, B \in F_L$ si esaminino le seguenti formule. La formula $A \wedge \neg A$ è insoddisfacibile, come si può osservare dalla sua tabella di verità nella Tabella 4.1.

| $A$ | $A \wedge \neg A$ |
|---|---|
| 0 | 0 |
| 1 | 0 |

Tabella 4.1

| $A$ | $A \to \neg A$ |
|---|---|
| 0 | 1 |
| 1 | 0 |

Tabella 4.2

La formula $A \to \neg A$ è soddisfacibile ma non tautologica, come si può osservare dalla sua tabella di verità nella Tabella 4.2.

Si noti come è stato detto che $A, B$ siano state definite come appartenenti all'insieme delle formule e non alle lettere ($L$), "imbrogliando", un poco, rispetto al lemma precedente. Tuttavia, $A$ e $B$ possono essere considerate come *metavariabili* a prescindere dalla loro complessità. Non è invece possibile fare il contrario, ossia considerare lettere come delle formule: per esempio, non è possibile dire che una lettera sia una tautologia.

**Esercizio** Verificare che le seguenti siano Tautologie per ogni $A, B \in F_L$:

1. $A \to (B \to A)$ (Weakening, Prefixing)
2. $(\neg A \to A) \to A$ (Consequentia Mirabilis)
3. $(A \to B) \vee (B \to A)$
4. $(A \to B) \to (\neg B \to \neg A)$ (Contronominale)
5. $(A \wedge B) \to A$
6. $A \vee \neg A$ (Tertium non datur)
7. $\bot \to A$ (Ex Falsum Quodlibet Sequitur)

**Definizione** (Tautologia)**.** *Se la formula $F$ è una tautologia, si indicherà*

$$\models F$$

## 4.3 Semantica degli Insiemi di Formule

**Definizione** (Teoria)**.** *Sia $\Gamma \subseteq F_L$ un insieme di formule. $\Gamma$ è detto **teoria** ed è un modello per un mondo in cui tutte le formule che gli appartengono sono vere.*

Si dice che $\Gamma$ è soddisfacibile se e solo se

$$\exists \mathfrak{v} : L \to \{0,1\} \quad \forall \gamma \in \Gamma : \mathfrak{v}(\gamma) = 1$$

o, in notazione alternativa, $\mathfrak{v} \models \gamma$ e contestualmente si dice $\mathfrak{v} \models \Gamma$. Al contrario, se e solo se

$$\forall \mathfrak{v} : L \to \{0,1\} \quad \exists \gamma \in \Gamma : \mathfrak{v}(\gamma) = 0$$

la teoria $\Gamma$ è insoddisfacibile e $\mathfrak{v} \not\models \gamma$ e $\mathfrak{v} \not\models \Gamma$.

**Definizione** (Conseguenza Logica)**.** *Sia $\Gamma \subseteq F_L$ e $A \in F_L$. La formula $A$ è una **conseguenza logica** di $\Gamma$ se e solo se*

$$\forall \mathfrak{v} : L \to \{0,1\} : \mathfrak{v} \models \gamma \in \Gamma$$

*(quindi $\mathfrak{v} \models \Gamma$) e $\mathfrak{v} \models A$. Si denoterà questo fatto con la notazione $\Gamma \models A$.*

[1] Per ogni letterale nell'insieme dei letterali nella formula.

**Esercizio** Esprimere il concetto di formula tautologica o tautologia ($\models A$) attraverso il concetto di conseguenza logica. Quando si ammette una teoria, si restringe il campo dei possibili assegnamenti. Una formula è tautologica quanto è vera in ogni circostanza ed è verificata da ogni assegnamento. Sia $\Gamma$ definito in questo modo. $\Gamma \subseteq F_L$ un insieme composto solo da tautologie. Allora $\Gamma \models A$. Dato che è sempre meglio avere come teoria la più semplice possibile, si può definire $\Gamma = \emptyset$, concludendo $\emptyset \models A$.
Ogni tanto si può utilizzare la notazione $\Gamma \cup \{A\} \models B$, che a volte viene semplificato in $\Gamma, A \models B$.

### 4.3.1 Proprietà semantiche fondamentali delle Teorie

**Lemma** (Conseguenza Logica di una Formula da una Teoria insoddisfacibile)**.** *Sia $\Gamma \subseteq F_L$ una teoria e $A \in F_L$. Si ha che $\Gamma \models A$ se e solo se $\Gamma \cup \{\neg A\}$ è insoddisfacibile.*

*Dimostrazione.* Dimostriamo che $\Gamma \models A$ se e solo se $\mathfrak{v} \models \gamma \forall \gamma \in \Gamma \rightarrow \mathfrak{v} \models A$, in altre parole per ogni assegnamento di verità tale che $\mathfrak{v} \models \Gamma$ si ha $\mathfrak{v}(A) = 1$. Si può tradurre quanto scritto utilizzando la definizione di implicazione materiale, ossia

$$\begin{aligned} & \neg(\mathfrak{v} \models \gamma \forall \gamma \in \Gamma) \vee \mathfrak{v} \models A \\ \iff & \neg(\mathfrak{v}(\gamma) = 1 \forall \gamma \in \Gamma) \vee \mathfrak{v}(A) = 1 \\ \iff & (\exists \gamma \in \Gamma : \mathfrak{v}(\gamma) = 0) \vee \mathfrak{v}(\neg A) = 0 \\ \iff & \exists B \in \Gamma \cup \{\neg A\} : \mathfrak{v}(B) = 0 \\ \iff & \Gamma \cup \{\neg A\} \text{ è insodd.} \end{aligned}$$

□

**Lemma** (Deduzione, semantico)**.** *Siano P, Q due formule. Allora si dice che $P \models Q$ se e solo se $\models P \rightarrow Q$.*

*Dimostrazione.* Assumiamo che $P \models Q$. Per definizione, $\forall \mathfrak{v} : L \rightarrow \{0,1\} : \mathfrak{v}(P) = 1 \rightarrow \mathfrak{v}(Q) = 1$. Assumendo, quindi, che ogni volta che $\mathfrak{v}(P) = 1$ si ha $\mathfrak{v}(Q) = 1$, si ha direttamente che $\models P \rightarrow Q$, in quanto sostanzialmente si rimuove la possibilità di avere $P$ vero e $Q$ falso. □

Il teorema generalizza il lemma alla seguente situazione.

**Teorema** (Deduzione,semantico)**.** *Sia $\Gamma$ una teoria e P, Q formule. Allora $\Gamma, P \models Q$ se e solo se $\Gamma \models P \rightarrow Q$.*

*Dimostrazione.* Si dimostra dicendo che $\Gamma, P \models Q$ se e solo se $\forall \mathfrak{v} : L \rightarrow \{0,1\} : \mathfrak{v}(\gamma) = 1 \forall \gamma \in \Gamma$ e $\mathfrak{v}(P) = 1$ si ha che $\mathfrak{v}(Q) = 1$. Questo avviene se e solo se $\forall \mathfrak{v} : L \rightarrow \{0,1\} : \mathfrak{v}(\gamma) = 1$ si ha $\mathfrak{v}(P \rightarrow Q) = 1$ e segue per definizione $\Gamma, P \models Q$. □

### 4.3.2 Teorema di Compattezza

Il teorema di compattezza è un teorema fondamentale della logica e varrà anche per la logica del prim'ordine. Lo proviamo ora per la logica proposizionale. È in qualche modo, in forma astratta, un teorema di completezza.

Prima di mostrare l'enunciato, si introduce il concetto. È stata data la nozione di teoria, $\Gamma \subseteq F_L$: ci si chiede se serve, nella logica proposizionale, considerare teorie infinite (composte da un numero infinito di formule). A priori, sembrerebbe proprio di sì - d'altronde $F_L$ è di cardinalità numerabile - e si vedrà affrontando la Logica dei Predicati che una singola formula

al Prim'Ordine contiene informazioni di un numero infinito di formule proposizionali; questo basta per giustificare il caso in cui $\Gamma$ sia una teoria infinita. Il punto è che non sembra possibile gestire la teoria infinita: qui torna utile il teorema di compattezza.

Il teorema di compattezza permette di ridurre l'analisi della soddisfacibilità di una teoria eventualmente infinita all'esame della soddisfacibilità dei suoi sottoinsiemi finiti. Tuttavia, è ovvio che il numero di sottoinsiemi finiti sia infinito.

Si enuncia, ora, il teorema di compattezza.

**Teorema** (Compattezza)**.** *Un insieme $\Gamma \subset F_L$ è soddisfacibile se e solo se lo è ogni $\Gamma' \subseteq \Gamma$, con $\Gamma'$ finito, con notazione: $\Gamma' \subseteq_\omega \Gamma$ (leggasi "$\Gamma'$ sottoinsieme finito di $\Gamma$").*

La dimostrazione consiste nel provare che che se ogni $\Gamma' \subseteq_\omega \Gamma$ è soddisfacibile allora lo è anche $\Gamma$, in quanto l'altro verso è ovvio dalla definizione di soddisfacibilità di una teoria, mentre il fatto che sottoinsiemi di $\Gamma$ siano soddisfacibili non implica ovviamente che $\Gamma$ sia soddisfacibile.

**Dimostrazione**

Per prima cosa, si definisce una porzione di terminologia utile per la dimostrazione.

**Definizione.** *Definiamo $\Gamma$ finitamente soddisfacibile (fin. sodd.) se per ogni $\Gamma' \subseteq_\omega \Gamma$ si ha che $\Gamma'$ è soddisfacibile.*

Il succo della dimostrazione sarà la relazione tra finitamente soddisfacibile e soddisfacibile, ossia si proverà che $\Gamma$ fin. sodd. implica $\Gamma$ soddisfacibile.

Fissato una successione

$$F_1, F_2, \cdots, F_k, \cdots$$

senza ripetizioni di tutte le formule in $F_L$ ($F_i \in F_L$), si costruisce una successione infinita di insiemi di formule

$$D_0, D_1, \cdots, D_k, \cdots$$

definita induttivamente sull'indice $i$ di $D_i$. Sia

$$\begin{cases} D_0 = \Gamma \\ D_{n+1} = \begin{cases} D_n \cup \{F_{n+1}\} & \text{se } D_n \cup \{F_{n+1}\} \text{ è finitamente soddisfacibile} \\ D_n \cup \{\neg F_{n+1}\} & \text{altrimenti} \end{cases} \end{cases}$$

Bisogna sottolineare che la definizione di $D_n$ non garantisce, a priori, che ogni $D_n$ sia finitamente soddisfacibile, anche se si dimostrerà che è in effetti così. In altre parole, definire $D_n = D_{n-1} \cup \{\neg F_{n+1}\}$ se l'alternativa non è finitamente soddisfacibile, non ci assicura a priori che $D_n$ sia fin. sodd.; per arrivare a tale conclusione è necessaria una dimostrazione. Si definisce infine

$$D = \bigcup_{i \in \mathbb{N}} D_i$$

Prima di passare alle effettive dimostrazioni, si noti come per dimostrare una proprietà di $\Gamma$ stiamo cercando di dimostrare qualcosa di relativo a un insieme infinitamente più grande, $D$; benché questa possa sembrare un'idea balzana, il Teorema di Compattezza ci dimostrerà che questo è il procedimento giusto per ottime ragioni.

**Primo fatto:** Per dimostrare che $D_n$ sia fin. sodd. per ogni $n$ ci si basa sull'induzione sull'indice $n$. Per la base dell'induzione, $D_0 = \Gamma$ è finitamente soddisfacibile per ipotesi. Per $n \neq 0$, si suppone che questo fatto sia vero per $D_0, \cdots, D_n$ e si prova vero per $D_{n+1}$.

Per mostrare $D_{n+1}$ finitamente soddisfacibile si procede per assurdo: si assume $D_{n+1}$ non finitamente soddisfacibile in modo di arrivare ad una contraddizione. Se $D_{n+1}$ non è finitamente soddisfacibile, allora esiste un insieme $D' \subseteq_\omega D_n \cup \{F_{n+1}\}$ tale che $D'$ è insoddisfacibile ed esiste $D'' \subseteq_\omega D_n \cup \{\neg F_{n+1}\}$ tale che $D''$ è insoddisfacibile.

Senza perdita di generalità, si può assumere che

$$F_{n+1} \in D' \text{ e } \neg F_{n+1} \in D''$$

in altre parole $D' = E' \cup \{F_{n+1}\}$ e $D'' = E'' \cup \{\neg F_{n+1}\}$. Si può affermare che $E', E'' \subseteq D_n$ e $F_{n+1} \notin E'$, $\neg F_{n+1} \notin E''$. Per concludere la dimostrazione del fatto, si noti che

$$E' \cup E'' \cup \{F_{n+1}\} \text{ e } E' \cup E'' \cup \{\neg F_{n+1}\}$$

sono insoddisfacibili perché contengono $D'$ e $D''$ rispettivamente. Ma $E' \cup E'' \subseteq_\omega D_n$ e per ipotesi induttiva $D_n$ è finitamente soddisfacibile, ed è quindi $E' \cup E''$ soddisfacibile ed esiste un assegnamento tale che $\mathfrak{v} \models E' \cup E''$. Sappiamo che $\mathfrak{v}(F_{n+1})$ è uguale a 0 o a 1 e pertanto non è possibile che entrambi $\{F_{n+1}\}$ e $\{\neg F_{n+1}\}$ siano falsi. Abbiamo raggiunto la contraddizione che conclude la prova per assurdo mostrando $D_{n+1}$ finitamente soddisfacibile. Questo chiude a sua volta la prova per induzione, mostrando che ogni $D_n$ per $n \in N$ è finitamente soddisfacibile.

**Secondo fatto:** $D = \cup_{i \in \omega} D_i$ è a sua volta finitamente soddisfacibile. Questo non è necessariamente ovvio a partire dal fatto che $D$ sia l'unione di insiemi finitamente soddisfacibili. Si consideri un sottoinsieme finito $D' \subseteq_\omega D$. Per dimostrare che $D'$ sia soddisfacibile (e, vista la generalità, ogni insieme $D'$ lo sia, quindi $D$ sia soddisfacibile), si può pensare di elencarne i membri $D' = \{F_{i_1}, F_{i_2}, \cdots, F_{i_u}\}$. Sia $k = \max_{j=1,\ldots,u} i_j$ l'indice massimo. Allora, $D' \subseteq_\omega D_k$, ma per il primo fatto $D_k$ è finitamente soddisfacibile e si può concludere $D'$ soddisfacibile e $D$ finitamente soddisfacibile.

**Terzo fatto:** Il terzo fatto da dimostrare è che per ogni formula o enunciato $F_t$ esattamente una tra $F_t$ e $\neg F_t$ appartiene a $D$. Anche questo non è necessariemente garantito, in quanto $F_{n+1}$ e $\neg F_{n+1}$ non hanno lo stesso indice, ossia $\neg F_{n+1}$ non ha indice $n+1$. La dimostrazione di questo fatto arriva col ragionamento seguente: per costruzione della sequenza dei $D_i$ $F_t$ o $\neg F_t$ appartiene a $D_t$ e quindi, dato che ogni $D_t \subseteq D$ si ha $F_t \in D$ o $\neg F_t \in D$. Bisogna escludere che ci siano entrambe, e che quindi la congiunzione "o" diventi uno xor. Si ha che $\{F_t, \neg F_t\} \nsubseteq D$, perché altrimenti per il secondo fatto $D$ è finitamente soddisfacibile e si avrebbe che $\{F_t, \neg F_t\}$ è soddisfsacibile, ma è assurdo, dato che ovviamente non può essere soddisfacibile per la semantica della negazione.

**Quarto fatto:** Infine, si definisce l'assegnamento $\mathfrak{v}_D : L \to \{0,1\}$: per ogni $p \in L$ si ha $\mathfrak{v}_D(p) = 1 \iff p \in D$. Si noti che $\mathfrak{v}_D$ è ben definito in quanto necessarimente $p \in D$ o $p \notin D$. L'ultimo fatto afferma che $\mathfrak{v}_D \models D$, che significa che $\forall F \in D \mathfrak{v}_D(F) = 1$. Se questo fatto è vero, allora è vero che $\mathfrak{v}_D \models \Gamma$ dato che $\Gamma \subseteq D$, dunque $\Gamma$ è soddisfacibile. Questo fatto si dimostra per induzione strutturale l'affermazione seguente: per ogni $F \in F_L$ si ha $\mathfrak{v}_D(F) = 1$ se e solo se $F \in D$. Questo non è uguale alla definizione precedente, in quanto in principio è stato definito per le lettere proposizionali e non per la funzione estesa ($\tilde{\mathfrak{v}}_D$). Ci basterebbe provare che $F \in D$ implica $\mathfrak{v}_D(F) = 1$ ma per convenienza proviamo anche che $\mathfrak{v}_D(F) = 1$ implica $F \in D$.

La base induttiva è $F = p$ e $p \in L$, l'asserto da provare segue dalla definizione di $\mathfrak{v}_D$. Il passo induttivo si svolge esaminando i connettivi uno per volta. Se $F = \neg G$, allora $\mathfrak{v}_D(F) = 1 \iff \mathfrak{v}_D(G) = 0 \iff G \notin D \iff \neg G \in D$ dal fatto tre $\iff F \in D$.

Se $F = (G \wedge H)$ allora $\mathfrak{v}_D(F) = 1 \iff \mathfrak{v}_D(G \wedge H) = 1 \iff \mathfrak{v}_D(G) = 1 \wedge \mathfrak{v}_D(H) = 1 \iff G \in D \wedge G \in D$. Questo ultimo fatto è dimostrabile perché se $G, H \in D$ allora $G \wedge G \in D$, poiché altrimenti se $\neg(G \wedge H) \in D$ implicherebbe che $\{G, H, \neg\{G \wedge H\}\} \subseteq_\omega \in D$ finitamente soddisfacibile, assurdo; e se $G \wedge H \in D$ allora $G, H \in D$, poichè altrimenti $G \notin D$ o $H \notin D$. Questo implica che $\{G \wedge H, \neg G\}$ o $\{G \wedge H, \neg H\} \in D$ siano finitamente soddisfacibili, assurdo.

Se $F = (G \vee H)$ allora $\mathfrak{v}_D(F) = \mathfrak{v}_D(G \vee H) \iff \mathfrak{v}_D(G) = 1 \vee \mathfrak{v}_D(H) = 1 \iff G \in D$ o $H \in D$ allora se $G \in D$ o $H \in D$, si ha che $(G \vee H) \in D$ poiché altrimenti se $(G \vee H) \notin D$ allora $\neg(G \vee H) \in D$, e se $G \in D$ allora $\{G, \neg\{G \vee H\}\} \subseteq_\omega D$ o se $H \in D$ allora $\{H, \neg\{G \vee H\}\} \subseteq_\omega D$ finitamente soddisfacibili, assurdo. Se $G \vee H \in D$ allora $G \in D$ o $H \in D$, poiché altrimenti se $G \notin D$ e $H \notin D$ allora $\neg G \in D$ e $\neg H \in D$ e $\{\neg G, \neg H, \{G \vee H\}\} \subseteq_\omega D$ finitamente soddisfacibile, impossibile.

**Dimostrazione della contronominale**

$\Gamma$ è insoddisfacibile se e solo se esiste $\Gamma' \subseteq_\omega \Gamma$ e $\Gamma'$ è insoddisfacibile.

$\leftarrow$ **contronominale** Se c'è $\Gamma' \subseteq_\omega \Gamma$ insoddisfacibile, allora $\Gamma$ è insoddisfacibile in quanto $\Gamma$ è più restrittivo di $\Gamma'$, ha meno assegnamenti di $\Gamma'$ che possono soddisfarlo.

$\rightarrow$ **contronominale** Se $\Gamma$ è insoddisfacibile, allora esiste $\Gamma' \subseteq_\omega \Gamma$, $\Gamma'$ insoddisfacibile.

### 4.3.3 Osservazioni sul Teorema di Completezza

L'idea di dimostrare la validità del teorema di compattezza partendo da una teoria composta da un insieme infinito di formule per creare una catena di insiemi

$$\Gamma = D_0 \subseteq D_1 \subseteq D_2 \cdots \subseteq D_k \cdots \subseteq D$$

è curioso. Si può concludere inoltre che $D$ non può essere ulteriormente ampliato senza perderne la soddisfacibilità, in quanto se $F \notin D$, allora $\overline{D} = D \cup \{F\}$ non è soddisfacibile: $F \notin D$ implica $\neg F \in D$. Questo vuol dire che $D$ è un **ampliamento massimale** di $\Gamma$.

**Definizione** (Insieme massimale soddisfacibile)**.** *Si definisce* ***insieme massimale soddisfacibile*** *ogni sottoinsieme $E \subseteq F_L$ tale che $E$ è soddisfacibile e $\forall F \in F_L : F \notin E$ si ha che $E \cup \{F\}$ è non soddisfacibile.*

$D$ è solo uno di tali insiemi ed è quello che è stato costruito a partire da $\Gamma$. $D$ è un ampliamento massimale di $\Gamma$ o *un* ampliamento di $\Gamma$? Pragmaticamente, se si elencano le formule in modo differente si ottiene, in genere, un ampliamento differente. A livello delle informazioni contenute, un insieme massimale soddisfacibile $D$ si comporta come un assegnamento. Per ogni assegnamento $\mathfrak{v} : L \to \{0, 1\}$ si può creare $D_\mathfrak{v} = \{F \in F_L : \mathfrak{v}(F) = 1\}$, un insieme massimale soddisfacibile. Il teorema di compattezza verrà usato per decidere i casi in cui $\Gamma \models A$. Infatti, per un lemma che abbiamo già dimostrato, $\Gamma \models A \iff \Gamma \cup \{\neg A\}$ è insoddisfacibile. Si supponga che $\Gamma$ sia una teoria finita, $\Gamma = \{B_1, \cdots, B_n\}$ e si vuole sapere se $\Gamma \models A$. Questo è vero se e solo se $B_1, \cdots, B_n \models A \iff \{B_1, \cdots, B_n, \neg A\}$ è insoddisfacibile, se e solo se $B_1 \wedge \cdots B_n \wedge \neg A$ è insoddisfacibile. Se, invece, $\Gamma$ è infinito viene finalmente in nostro aiuto il teorema di compattezza.

## 4.4 Equivalenza Semantica

Sintatticamente, come enunciati, cioè come stringhe di simboli, le due formule

$$A \vee B$$

e

$$B \vee A$$

sono due formule differenti, nonostante $A$ sia $A$ e $B$ sia $B$: sono *scritte* in modo diverso. Il loro **significato**, tuttavia, è uguale e, pertanto, per ogni $\mathfrak{v} : L \to \{0,1\}$ si ha che $\mathfrak{v}(A \vee B) \equiv \mathfrak{v}(B \vee A)$. Quella appena definita è una **relazione**.

**Definizione** (Relazione)**.** *Una relazione n-aria R su un insieme S è un sottoinsieme dell'insieme di tutte le n-ple di tutti gli elementi di S, in altre parole* $R \subseteq S^n$.

Per esempio, una relazione binaria è una relazione $R$ su $S$ tale che $R \subseteq S^2$.

**Definizione** (Relazione di equivalenza)**.** *Una* ***relazione d'equivalenza*** *è una relazione binaria R su S,* $R \subseteq S^2$ *tale che*

- *R è riflessiva:* $\forall s \in S \;\; (s,s) \in R$
- *R è simmetrica:* $\forall s_1, s_2 \in S \;\; (s_1,s_2) \in R \to (s_2,s_1) \in R$
- *R è transitiva:* $\forall s_1, s_2, s_3 \in S \;\; (s_1,s_2),(s_2,s_3) \in R \to (s_1,s_3) \in R$

### 4.4.1 Partizioni e classi di equivalenza

Sia $R$ una relazione di equivalenza su $S$. Per ogni $s \in S$, la **classe di equivalenza** rispetto ad $R$ di $s \in S$ è definita

$$[s]_R = \{t \in S : (s,t) \in R\}$$

È importante osservare che se $(s,t) \in R$ allora $[s]_R = [t]_R$ e viceversa, ossia se $[s]_R = [t]_R$ allora $(s,t) \in R$. Una **partizione** di un insieme $S$ è un insieme $\{B_1, \cdots, B_k, \cdots\}$ non necessariamente finito di sottoinsiemi di $S$ ($B_i \subseteq S$) tale che $B_i \cap B_j = \emptyset$ per ogni $i, j \in I$, dove $I$ è l'insieme di indici di $B$. Inoltre, $\cup_{i \in I} B_i = S$. Gli insiemi $B_i$ vengono chiamati *blocchi* della partizione. I due concetti, partizioni e relazioni di equivalenza, sono legati tra di loro: ogni relazione di equivalenza $R$ su $S$ determina una partizione $P_R$ di $S$ dove

$$P_R = \{[s]_R : s \in S\}$$

ossia le classi di equivalenza definite da $R$. La relazione inversa è meno nota, ossia che ogni partizione $P$ di $S$ determina una relazione d'equivalenza $R_P$ su $S$, ossia

$$R_P := (s,t) \in R_p \implies \exists i : s,t \in B_i$$

che è chiaramente riflessiva, simmetrica e transitiva.

### 4.4.2 Equivalenza Logica

Verificare che la relazione seguente (equivalenza logica $\equiv$) è d'equivalenza:

$$\equiv \subseteq F_L^2$$

$$\forall A,B : (A,B) \in \equiv \forall \mathfrak{v} : L \to \{0,1\} \mathfrak{v}(A) = \mathfrak{v}(B)$$

1. $\equiv$ riflessiva: $A \equiv A$ infatti $\forall \mathfrak{v} : L \to \{0,1\} \mathfrak{v}(A) = \mathfrak{v}(A)$
2. $\equiv$ simmetrica: per ogni $(A,B) \in F_L^2$ se $A \equiv B$ allora $B \equiv A$, ossia se per ogni assegnamento $\mathfrak{v} : L \to \{0,1\}$ si ha che $\mathfrak{v}(A) = \mathfrak{v}(B)$ allora per ogni $v$ si ha $\mathfrak{v}(B) = \mathfrak{v}(A)$.
3. $\equiv$ transitiva: per ogni $(A,B,C) \in F_L^3$ se $A \equiv B$ e $B \equiv C$ allora $A \equiv C$ (dalla transitività della funzione $=$ stessa).

$\equiv$ partiziona $F_L$, infatti

$$P_{\equiv} = F_L / \equiv = \{[A]_{\equiv} : A \in F_L\}$$

### 4.4.3 Sostituzione di lettera proposizionale con Formula

Arricchiremo l'insieme delle classi di equivalenza delle formule, rispetto all'equivalenza logica, con delle operazioni che equipaggeranno tale insieme di una struttura algebrica. Siano $A,B \in F_L$ delle formule e sia $p \in L$ una lettera proposizionale. Si definisce la formula che si ottiene sostituendo nella formula $A$ ogni occorrenza della lettera proposizionale $p$ con la formula $B$

$$A[B/p]$$

induttivamente, con l'idea che per esempio data $(p \wedge \neg q)$ si ha $[(t \to q)/p] = (t \to q) \wedge \neg q$.
**base:** $A = q, p \in L$ allora

$$q[B/p] := \begin{cases} B & p = q \\ q & p \neq q \end{cases}$$

**passo:** $(\neg A)[B/p] := \neg(A[B/p])$, $(A_1 \wedge A_2)[B/p] := (A_1[B/p]) \wedge (A_2[B/p])$, $(A_1 \vee A_2)[B/p] := (A_1[B/p]) \vee (A_2[B/p])$, $(A_1 \implies A_2)[B/p] := (A_1[B/p] \implies A_2[B/p])$.

**Lemma** (di sostituzione)**.** *Sia* $\mathfrak{v} : L \to \{0,1\}$ *e siamo* $S,T \in F_L$*. Se* $\mathfrak{v}(S) = \mathfrak{v}(T)$*, allora per ogni* $A \in F_L$ *e ogni* $p \in L$

$$\mathfrak{v}(A[S/p]) = \mathfrak{v}(A[T/p])$$

ossia, se si sostituisce nella stessa formula una lettera proposizionale con due enunciati che hanno lo stesso valore di verità il valore delle due sostituzioni è uguale. La dimostrazione è per induzione strutturale su $A$:

**base:** $A = q$, $q \in L$ allora

$$\begin{cases} q \neq p & q[S/p] = q = q[T/p] \\ q = p & q[S/p] = S \wedge q[T/p] = T, \mathfrak{v}(S) = \mathfrak{v}(T) \to \mathfrak{v}(q[S/p]) = \mathfrak{v}(q[T/p]) \end{cases}$$

**passo**: $A = \neg B$ per ipotesi induttiva $\mathfrak{v}(B[S/p]) = \mathfrak{v}(B[T/p])$, e quindi $\neg\mathfrak{v}(B[S/p]) = \neg\mathfrak{v}(B[T/p])$. Analogamente per gli operatori binari: $A = (B \wedge C)$ e per i.h. $\mathfrak{v}(B[S/p]) = \mathfrak{v}(B[T/p])$ e $\mathfrak{v}(C[S/p]) = \mathfrak{v}(c[T/p])$ e quindi $\mathfrak{v}((B \wedge C)[S/p]) = \mathfrak{v}(B[S/p] \wedge C[S/p])$.

Grazie al lemma di sostituzione si può provare il seguente teorema.

**Teorema** (di sostituzione)**.** *Se $S \equiv T$, allora per ogni $A \in F_L$ e $p \in L$ vale*

$$A[S/p] \equiv A[T/p]$$

La differenza arriva dall'utilizzo di un solo assegnamento nel lemma e dall'utilizzo di equivalenza in questo teorema. La dimostrazione è ovvia grazie alla definizione di equivalenza e dal lemma di sostituzione. Ricordando che per ogni $\mathfrak{v} : L \to \{0,1\}$ vale $\mathfrak{v}(S) = \mathfrak{v}(T)$ in quanto per ipotesi $S \equiv T$ si ha che per il lemma di sostituzione vale

$$\forall \mathfrak{v} : L \to \{0,1\} \mathfrak{v}(A[S/p]) = \mathfrak{v}(A[T/p])$$

dunque, per la definizione di $\equiv$ si ha $A[S/p] \equiv A[T/p]$.

### 4.4.4 Teoremi di equivalenze logiche

Date $F, G, H \in F_L$ le seguenti sono equivalenze logiche:
**idempotenza:**

$$\begin{cases} F \vee F \equiv F \\ F \wedge F \equiv F \end{cases}$$

**commutatività:**

$$\begin{cases} F \vee G \equiv G \vee F \\ F \wedge G \equiv F \wedge F \end{cases}$$

**associatività:**

$$\begin{cases} F \wedge (G \wedge H) \equiv (F \wedge G) \wedge H \\ F \vee (G \vee H) \equiv (F \vee G) \vee H \end{cases}$$

**assorbimento:**

$$\begin{cases} F \vee (F \wedge G) \equiv F \\ F \wedge (F \vee G) \equiv F \end{cases}$$

**distributività:**

$$\begin{cases} F \wedge (G \vee H) \equiv (F \wedge G) \vee (F \wedge H) \\ F \vee (G \wedge H) \equiv (F \vee G) \wedge (F \vee H) \end{cases}$$

**doppia negazione:**

$$\neg\neg F \equiv F$$

**leggi di De Morgan:**

$$\begin{cases} \neg(F \vee G) \equiv \neg F \wedge \neg G \\ \neg(F \wedge G) \equiv \neg F \vee \neg G \end{cases}$$

**interdefinibilità:**

$$\begin{cases} F \to G \equiv (\neg F) \vee G \\ F \to G \equiv \neg(F \wedge \neg G) \\ F \vee G \equiv (\neg F) \to G \\ F \wedge G \equiv \neg(F \to G) \\ F \vee G \equiv \neg(\neg F \wedge \neg G) \\ F \wedge G \equiv \neg(\neg F \vee \neg G) \end{cases}$$

**doppia implicazione**

$$F \iff G \equiv (F \to G) \wedge (G \to F)$$

**tautologia iff.**

$$\models F \iff G \iff F \equiv G$$

**costanti logiche**

$$\begin{cases} \bot := I_\bot : \{0,1\}^0 \to \{0,1\} \text{ la funzione costante } I_\bot = 0 \\ \top := I_\top : \{0,1\}^0 \to \{0,1\} \text{ la funzione costante } I_\top = 1 \end{cases}$$

come primitivi oppure, come derivati

$$\begin{cases} \bot := (\neg\top) \text{ oppure } p \wedge \neg p \\ \top := (\neg\bot) \text{ oppure } p \to p, p \vee \neg p \\ \neg A \equiv A \implies \bot \\ A \equiv \top \implies A \end{cases}$$

**complemento:**

$$\begin{cases} A \wedge \neg A \equiv \bot \\ A \vee \neg A \equiv \top \end{cases}$$

**elementi neutri:**

$$\begin{cases} A \wedge \bot \equiv \bot \\ A \vee \bot \equiv A \\ A \wedge \top \equiv A \\ A \vee \top \equiv \top \end{cases}$$

**De Morgan su più formule:**

$$\begin{cases} \neg(F_1 \vee F_2 \vee \cdots F_n) = \neg F_1 \wedge \neg F_2 \wedge \cdots \neg F_n \\ \neg(F_1 \wedge F_2 \wedge \cdots F_n) = \neg F_1 \vee \neg F_2 \vee \cdots \neg F_n \end{cases}$$

**Distributibilità generalizzata**

$$\begin{cases} (F_1 \vee \cdots \vee F_u) \wedge (G_1 \vee \cdots \vee G_n) \equiv \vee_{i=1}^{u} \vee_{j=1}^{v} (F_i \wedge G_j) \\ (F_1 \wedge \cdots \wedge F_u) \vee (G_1 \wedge \cdots \wedge G_n) \equiv \wedge_{i=1}^{u} \wedge_{j=1}^{v} (F_i \wedge G_j) \end{cases}$$

**Esercizio** Provare l'idempotenza tramite l'assorbimento.

Provare l'idempotenza tramite l'assorbimento delle formule: $F \vee F \equiv F$ Sia $p \in L$, con $p \notin F$. Si operano le sostutizioni

$$(F \vee p)[F/p] = F \vee F = F$$

$$\begin{aligned} (F \vee p)[F \wedge (F \vee F)/p] &= F \vee (F \wedge (F \vee F)) \\ = F \vee (F \wedge (F \vee F)) &= F \end{aligned}$$

Le due forme sono, per assorbimento, uguali.

Tutte le leggi che abbiamo indicato discendono dalle quattro leggi di **commutatività**, **associatività**, **assorbimento**, **distrubtività** e **complemento**, pertanto sarebbe stato abbastanza postulare queste quattro leggi e le rimanenti sarebbero "autonomamente" verificate.

Per inquadrare meglio questo fatto, facciamo delle osservazioni finali su $\equiv$.

**Osservazione.** *Se $A$ e $B$ sono due enunciati e $A \equiv B$ e $\models A$, allora $\models B$.*

Una conseguenza di ciò è che tutte le tautologie sono tra di loro logicamente equivalenti e analogamente le contraddizioni; in altre parole c'è una classe di equivalenza tra le tautologie e una classe di equivalenza tra le contraddizioni.

**Osservazione.** *Se $A \equiv B$ e $A$ è insoddisfacibile, allora $B$ è insoddisfacibile.*

**Osservazione.** *Se $A \models B$ e $B \models A$ allora $A \equiv B$.*

Questo si può vedere grazie al teorema di deduzione: $\models A \implies B$ e $\models B \implies A$ e grazie alla definizione di coimplicazione si ha $A \equiv B$.

L'osservazione più importante è la seguente:

**Osservazione.** *L'equivalenza logica è più di una relazione di equivalenza rispetto ad un'equivalenza semplice: infatti, $\equiv$ è una **congurenza** rispetto ai connettivi pensati come operazioni.*

Per esempio, preso il connettivo $\wedge$, quattro formule $A, B, C, D \in F_L$, allora se $A \equiv B$ e $C \equiv D$, si avrà $A \wedge C \equiv B \wedge D$; analogamente accade per ogni connettivo. Questa proprietà non è ovvia! Vi sono, infatti, situazioni in cui questo non accade anche per una relazione d'equivalenza, la quale non è automaticamente anche una congruenza. Per esempio, si prenda come relazione d'equivalenza quella tale per cui ogni numero dispari appartiene allo stesso blocco e ogni numero pari appartiene al proprio singleton e si consideri come operazione la somma dei numeri naturali. In altre parole, si ha che, fissata la relazione $R$, $[3]_R = \{1,3,5,7,9,\cdots\}$, $[2]_R = \{2\}$, $[4]_R = \{4\}$ eccetera. Questa è una relazione d'equivalenza poiché è una partizione di $\mathbb{N}$, ma non è una congruenza rispetto alla somma dei naturali: sommando, per esempio, $1+1$ il risultato ricade in $[2]$, mentre sommando $3+1$ il risultato ricade in $[4]$, anche se $1 \in [3]$ appartengono alla stessa classe d'equivalenza.

## 4.5 Algebrizzabilità

Il fatto che l'equivalenza logica sia una congruenza, è la chiave per definire l'**algebrizzabilità** della Logica Booleana, la quale sarà utilizzata per definire le *Forme Normali* in seguito. Prima di introdurre formalmente questo concetto, è necessario introdurre un concetto molto importante, ossia le **relazioni d'ordine**.

### 4.5.1 Relazioni d'ordine (parziale)

Una **relazione d'ordine** $R$ è una relazione binaria $R \subseteq S^2$ che soddisfa tre proprietà:

- Riflessiva $\forall s \in S \;\; (s,s) \in R$
- Antisimmetrica $\forall s,t \in S \;\; ((s,t) \in R \wedge (t,s) \in R) \rightarrow s = t$
- Transitiva $s,t,r \in S \;\; ((s,t) \in R \wedge (t,r) \in R) \rightarrow (s,r) \in R$

Spesso, per indicare questa relazione si utilizza il simbolo $\leq$, a prescindere dal fatto che si indichi effettivamente il senso di ordinamento che si intende solitamente. Le relazioni d'ordine possono essere totali o parziali: nel primo caso, dati qualunque due elementi dell'insieme di appartenenza sussiste necessariamente una relazione tra i due. Nel secondo caso, invece, questo non è necessariamente vero.

Un esempio di relazione d'ordine *totale* è $(\mathbb{N}, \leq)$, ossia $a \leq b$ se esiste $c \in \mathbb{N}$ tale che $a + c = b$; questo ordine è totale poiché dati due numeri naturali qualunque si può sempre definire un ordine tra di essi.

Un esempio di relazione *parziale* è $(\mathbb{N}, |)$ ossia si indica che sussiste la relazione tra due numeri $a, b$ dicendo $a \leq b$ se $a|b$, ossia $a$ divide $b$ ed esiste $c$ tale che $a \cdot c = b$; questa relazione è decisamente diversa.

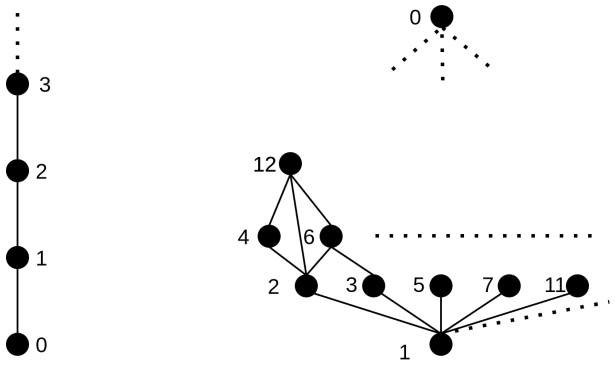


Figura 4.1: Due relazioni d'ordine.

Nella Figura 4.1 si può vedere una rappresentazione (tramite reticoli, definiti a seguire) delle due relazioni d'ordine. La prima, quella totale, è rappresentata a destra: si può notare come sia "lineare" a confronto della seconda, parziale, che è rappresentata a sinistra: quest'ultima infatti si dirama: alla base ha 1, il numero che divide tutti gli altri; al primo "strato" ha tutti i numeri primi, il secondo i multipli dei multipli dei numeri primi (che saranno comunque collegati ai numeri primi, in quanto saranno divisibili anche per essi) e, commettendo un abuso che solitamente viene concesso, in cima vi è il numero 0 che è divisibile da tutti gli altri, benché non sia divisibile per sé stesso.

### Reticoli

Un insieme $P$ con una relazione d'ordine (parziale) $\leq$, notato $(P, \leq)$ è detto insieme parzialmente ordinato o, in inglese, partially ordered set (*poset*). L'insieme delle classi di equivalenza delle formule è un insieme parzialmente ordinato con delle peculiarità.

Tra tutti i *poset*, ci interessano quelli con alcune particolari proprietà strutturali, chiamati **reticoli**.

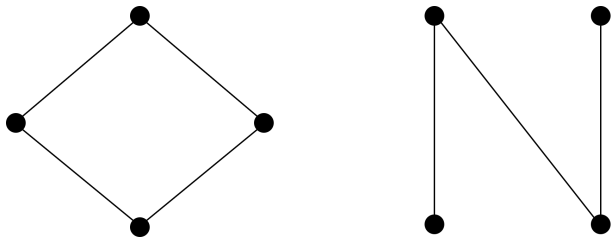

Figura 4.2: Due reticoli.

La Figura 4.2 riporta due immagini. Non entrambi sono reticoli: quello più a destra, infatti, non lo è. Cosa li distingue? La proprietà strutturale (più importante per i nostri scopi) che li divide, è che per quello più a sinistra, il reticolo, data ogni qualsiasi coppia di elementi possibile, si può sempre trovare quale dei due sia il minimo elemento che è più grande della

coppia e il massimo elemento che è più piccolo della coppia. Questo non avviene nell'altro caso, infatti i due elementi "in alto", non hanno un minimo elemento più grande di loro, mentre i due elementi "in basso" non hanno un massimo elemento più piccolo di loro. In altre parole, non hanno **estremi inferiori** ed **estremi superiori**, che sono invece la proprietà fondamentale dei reticoli.

Si definiscono quindi questi due concetti.

**Definizione** (Estremo inferiore)**.** *Dati due elementi $x,y$ appartenti ad un insieme parzialmente ordinato $(P,\leq)$, si definisce*

$$\inf(\{x,y\}) = \max\{z \in (P,\leq) : z \leq x \; e \; z \leq y\}$$

e si definisce, analogamente

**Definizione** (Estremo superiore)**.** *Dati due elementi $x,y$ appartenti ad un insieme parzialmente ordinato $(P,\leq)$, si definisce*

$$\sup(\{x,y\}) = \min\{z \in (P,\leq) : x \leq z \; e \; y \leq z\}$$

Si possono generalizzare inf e sup ad un insieme $S \subseteq P$, con $P$ poset $(P,\leq)$, affermando che

$$\inf(S) = \max\{z \in (P,\leq) : z \leq s \forall s \in S\}$$

e

$$\sup(S) = \min\{z \in (P,\leq) : s \leq z \forall s \in S\}$$

A questo punto si può definire il massimo di un insieme (e analogamente il minimo) definendolo come

$$\max(S) = \sup(S) \text{ se } \sup(S) \in S$$

Si può ora definire formalmente un reticolo:

**Definizione** (Reticolo)**.** *Un reticolo è un insieme parzialmente ordinato $(R,\leq)$ tale che per ogni $x,y \in R$ esistono* $\inf(\{x,y\})$ *e* $\sup(\{x,y\})$.

Dopo aver definito i reticoli, si definisce un'ulteriore struttura, necessaria per concludere l'argomento dell'Algebrizzazione della Logica Booleana.

### 4.5.2 Struttura Algebrica

Dato un insieme $S$ e date le operazioni $*_1, *_2, \cdots, *_n$ delle operazioni in $S$, allora $(S, *_1, *_2, \cdots, *_n)$ è una **struttura algebrica**.

Visto come una struttura algebrica, un reticolo è $(R,\sqcup,\sqcap)$, con $\sqcup,\sqcap$ operazioni $\sqcup : R^2 \to R$ e $\sqcap : R^2 \to R$, tale che le due operazioni siano commutative, associative e valga l'assorbento. Ogni reticolo visto come insieme parzialmente ordinato è anche una struttura algebrica

$$(R,\leq) \iff (R,\sqcup,\sqcap)$$

Infatti si può associare una struttura algebrica a una struttura ordinata (ossia si può definire il reticolo come struttura algebrica partendo dalla sua definizione come poset) definendo $R \equiv R$, $a \sqcup b := \inf\{a,b\}$ e $a \sqcap b := \sup\{a,b\}$ per ogni $a,b \in R$. Si verifichino le proprietà di commutatività, assorbimento e associatività.

Si può associare una struttura ordinata a ogni struttura algebrica (ossia si può definire il reticolo come poset partendo dalla sua definizione come struttura algebrica) definendo $R \equiv R$ e sostanzialmente definendo quando accade che $a \leq b$.

Per ogni $a, b \in R$ si definisce un operatore $a = a \sqcap b$ o equivalentemente $b = a \sqcup b$. Si deve verificare che $a \leq b \iff a = a \sqcap b$ sia tale che $\leq$ sia riflessiva, antisimmetrica e transitiva, dalla quale deriva che $R$ è un poset. Per concludere che $R$ sia in particolare un poset-reticolo bisogna mostrare che per ogni coppia $a, b \in R$ esiste inf e sup. Sia allora $c \leq a$ e $c \leq b$: bisogna mostrare che $c \leq \inf\{a, b\}$, in genere $c \leq a \sqcap b$. Ma $c = a \sqcap c = a \sqcap (b \sqcap c)$ per la definizione di $\leq$; si può allora scrivere

$$\text{per associatività } (a \sqcap b) \sqcap c \iff c \leq a \sqcap b$$

per definizione di $\leq$. Per il sup si ragiona in modo analogo.

### 4.5.3 Definizione di Algebra Booleana

A questo punto, siamo finalmente pronti per definire cosa sia l'Algebra Booleana.

**Definizione.** *Algebra Booleana L'Algebra Booleana è un sistema* $(S, \sqcap, \sqcup, 0, 1, ()^c)$ *operazioni tali che* $(S, \sqcup, \sqcap)$ *sia un reticolo limitato, complementato, distributivo:*

- ***limitato*** *in quanto per ogni* $a \in S$ *si ha* $a \sqcap 0 = a$ *e* $a \sqcap 1 = a$ *(*$a \geq 0$ *e* $a \leq 1$*);*
- ***distributivo*** *in quanto per ogni* $a, bc \in S$ *si ha*

$$a \sqcap (b \sqcup c) = (a \sqcap b) \sqcup (a \sqcap c)$$

*e*

$$a \sqcup (b \sqcap c) = (a \sqcup b) \sqcap (a \sqcup c)$$

- ***complementato*** *in quanto* $a \sqcap a^c = 0$ *e* $a \sqcup a^c = 1$.

Un esempio è il seguente: dato un insieme $A$ si consideri $(P(A), \cup, \cap, \emptyset, A, ()^c)$ e si ha $(P(A), \subseteq)$ tale che $A \subseteq B \iff A = A \cap B$.

**Esercizio** Sia $A$ un insieme infinito e sia $F(A)$ l'insieme dei suoi sottoinsiemi finiti o complementi di insiemi finiti. Si dimostri che $(F(A), \cup, \cap, \emptyset, A, ()^c)$ è un algebra booleana. Un altro esempio di algebra booleana è quella dei valori di verità $(\{0,1\}, \min, \max, 0, 2, 1 - \_)$.

#### Algebra di Lindenbaum

L'algebra delle formule, chiama anche **Algebra di Lindenbaum delle Formule**, è definito $(F_L / \equiv, \wedge, \vee, \bot, \top, \neg)$. Dato che $\equiv$ è una congruenza, si può quindi procedere a definire

$$A, B \in F_L [A]_\equiv \wedge [B]_\equiv = [A \wedge B]_\equiv$$
$$A, B \in F_L [A]_\equiv \vee [B]_\equiv = [A \vee B]_\equiv$$
$$A, B \in F_L [A]_\equiv \to [B]_\equiv = [A \to B]_\equiv$$
$$A \in F_L \neg [A]_\equiv = [\neg A]_\equiv$$
$$[\top]_\equiv = \text{ tautologie}$$
$$[\bot]_\equiv = \text{ contraddizioni}$$

La cardinalità dell'insieme delle classi di equivalenza scrivibili con $n$ variabili è

$$|F^{(n)} / \equiv| = 2^{2^n} = |\{F : \{0,1\}^n \to \{0,1\}\}|$$

### 4.5.4 Funzioni Termine

Al fine di arrivare ad un discorso completo sulle Forme Normali, si continua a ragionare sulle classi d'equivalenza delle formule e la loro struttura, ora da un punto di vista che sembra essere diverso ma che in conclusione si riunirà con quanto detto precedentemente.

Si rifletta sul significato di

$$\models F \iff \forall \mathfrak{v} : L \to \{0,1\} \quad \mathfrak{v}(F) = 1$$

in termini "computazionali" significa *tenere fermo* l'argomento ($F$) e far cambiare le funzioni. Sarebbe comodo scambiare i ruoli, nel senso che si vorrebbe che la formula diventasse una funzione, in qualche modo, e gli assegnamenti degli argomenti. Sia ora $A \in F_L$ una formula. Si chiama $Var(A) \subseteq L$ l'insieme delle lettere proposizionali che occorrono in $A$ tale che $Var(A) \subseteq \{p_1, p_2, \cdots, p_n\}$ per un qualche naturale $n$. Sia ora $\hat{A}$ una funzione definita come

$$\hat{A} : \{0,1\}^n \to \{0,1\}$$

tale che

$$\forall \mathfrak{v} : L \to \{0,1\} \quad \hat{A}(\mathfrak{v}(p_1), \mathfrak{v}(p_2), \cdots, \mathfrak{v}(p_n)) = \tilde{\mathfrak{v}}(A)$$

la fuzione $\hat{A}$ si chiama **funzione termine** e, per definirla formalmente, si procede per induzione.

**Caso base:** Sia $A = p_i$, con $p_i \in \{p_1, p_2, \cdots, p_n\}$ un letterale. Allora

$$\hat{A} := \hat{p}_i : \{0,1\}^n \to \{0,1\}$$

dove, per ogni $(t_1, \cdots, t_n)$ si ha

$$\hat{p}_i(t_1, \cdots, t_n) = t_i$$

ossia l'$i$-esima proiezione. Quindi, nel caso base, $\forall \mathfrak{v} : L \to \{0,1\}$ si ha che

$$\hat{A} = \hat{p}_i(\mathfrak{v}(p_1), \cdots, \mathfrak{v}(p_n)) = \mathfrak{v}(p_i)$$

**Passo induttivo:**

- Se $A = \neg B$ allora $\hat{A}(t_1, \cdots, t_n) = 1 - \hat{B}(t_1, \cdots, t_n)$.
- Se $A = B \wedge C$ allora $\hat{A}(t_1, \cdots, t_n) = min\{\hat{B}(t_1, \cdots, t_n), \hat{C}(t_1, \cdots, t_n)\}$.
- Se $A = B \vee C$ allora $\hat{A}(t_1, \cdots, t_n) = max\{\hat{B}(t_1, \cdots, t_n), \hat{C}(t_1, \cdots, t_n)\}$.
- Se $A = B \to C$ allora $\hat{A}(t_1, \cdots, t_n) = max\{1 - \hat{B}(t_1, \cdots, t_n), \hat{C}(t_1, \cdots, t_n)\}$.

Ora si può trasformare la riflessione iniziale in termini di Funzioni Termine:

$$\models F \iff \hat{F}(t_1, \cdots, t_n) = 1 \forall (t_1, \cdots, t_n)$$

e l'equivalenza logica:

$$[F]_{\equiv} = [G]_{\equiv} \iff F \equiv G \iff \hat{F}(t_1, \cdots, t_n) = \hat{G}(t_1, \cdots, t_n) \forall t_i$$

A questo punto, potrebbo avere una definizione diversa di Algebra Booleana da esporre, che in realtà è la stessa Algebra di Lindenbaum scritta in maniera diversa

**Definizione** (Algebra Booleana su Funzioni Termine)**.**

$$\hat{F}_L = (\{\hat{A} : \{0,1\}^n \to \{0,1\}, A \in F_L\}, \wedge, \vee, \neg, \bot, \top)$$

*dove i significati degli operatori sono quelli espressi precedentemente.*

Eccetto la natura "fisica", l'Algebra di Lindenbaum e l'Algebra Booleana su Funzioni Termine sui primi $n$ termini sono **isomorfe**.

### 4.5.5 Teorema di Completezza Funzionale

Concentriamoci sull'Algebra appena espressa

$$\hat{F_L} = (\{\hat{A} : \{0,1\}^n \to \{0,1\}, A \in F_L\}, \wedge, \vee, \neg, \bot, \top)$$

e costruiamo un'altra Algebra definita come

$$Funz^{(n)} = (\{F : \{0,1\}^{(n)} \to \{0,1\}\}, \wedge, \vee, \neg, \bot, \top)$$

Che rapporto sussiste tra le due Algebre? La prima ha come universo l'insieme di tutte le Funzioni Termine per ogni formula $A \in F_L$, mentre la seconda ha come universo tutte le funzioni

$$F : \{0,1\}^n \to \{0,1\}$$

per ogni $n \in \mathbb{N}$. Sicuramente, data la restrizione, si può affermare che

$$\hat{F_L}^{(n)} \subseteq Funz^{(n)}$$

Si dimostra, ora, che vale anche

$$\hat{F_L}^{(n)} \supseteq Funz^{(n)}$$

Questo fatto prende il nome di **Teorema di Completezza Funzionale**. A parole, ciò che si afferma è che ogni funzione

$$F : \{0,1\}^n \to \{0,1\}$$

è esprimibile come una formula $F \in F_L$. Chiaramente, dimostrando questo, si ha che le due Algebre esposte sono in realtà la stessa, ossia sono **isomorfe**.

**Teorema** (di Completezza Funzionale)**.** *Per ogni* $n \in \mathbb{N}$

$$\hat{F_L}^{(n)} \supseteq Funz^{(n)}$$

La dimostrazione consiste nel fornire un metodo per ottenere una Funzione Termine

$$\hat{F} = F$$

per ogni formula $F : \{0,1\}^n \to \{0,1\}$.

*Dimostrazione.* Sia data una formula

$$F : \{0,1\}^n \to \{0,1\}$$

La cardinalità del dominio di $F$ è

$$|Dom(F)| = 2^n$$

e per ogni combinazione degli $n$ argomenti esiste un certo risultato associato, che definiamo $b_i \;\; i = 0, \cdots, 2^n - 1$. Costruiamo una formula $F \in F_L$ in modo tale che $\hat{F} = F$. Se $F(t_1, \cdots, t_n) = 0$ per ogni $(t_1, \cdots, t_n)$, allora si può semplicemente scegliere $F = \bot$, analogamente nel caso in cui sia sempre vera.

Altrimenti, sia

$$I \subseteq \{0, 1, \cdots, 2^n - 1\}$$

tale che

$$i \in I \iff b_i = 1$$

(intuitivamente, $I$ contiene i numeri delle "righe" della tabella di verità della funzione $F$ in cui l'immagine è il numero 1). Per ogni $i \in I$ si considera la $i$-esimo elemento del dominio di $F$, che è composto come

$$(a_{i1}, a_{i2}, \cdots, a_{in})$$

e per ogni $j \in \{1, \cdots, n\}$ si definisce

$$A_{ij} = \begin{cases} p_j & a_{ij} = 1 \\ \neg p_j & a_{ij} = 0 \end{cases}$$

Si noti che $A_{ij} \in F_L$. Si pone, ora

$$F = \vee_{i \in I} \wedge_j A_{ij}$$

E si può mostrare $\hat{F} = F$: chiaramente,

$$\wedge_{j=1}^{n} \hat{A}_{ij}(t_1, \cdots, t_n) = 1$$

se e solo se

$$(t_1, \cdots, t_n) = (a_{i1}, \cdots, a_{in})$$

Altrettanto ovviamente, considerando

$$(A_{i1} \wedge A_{i2} \wedge \cdots \wedge A_{in})$$

nel "punto" $(a_{i1}, a_{i2}, \cdots, a_{in})$ si sa che il suo valore di verità è 1, ed è l'unico caso in cui sia così; in definitiva,

$$\hat{F} = \vee_{i \in I} \wedge_j A_{ij}(t_1, \cdots, t_n) = 1 \iff i \in I$$

e $\hat{F} = F$. □

**Considerazioni sul Teorema di Completezza Funzionale**

Per come è stato provato il Teorema di Completezza Funzionale, la funzione può essere espressa

$$F = \vee_{i \in I} \wedge_{j=1}^{n} A_{ij}$$

e

$$G = \wedge_{i \in J} \vee_{j=1}^{n} B_{ij}$$

definendo

$$J \subseteq 0, \cdots, 2^n - 1 \quad j \in J \iff b_j = 0$$

e

$$B_{ij} = \begin{cases} \neg p_j & a_{ij} = 1 \\ p_j & a_{ij} = 0 \end{cases}$$

Si ha che $\hat{F} = F = \hat{G}$[2], è pertanto una dimostrazione dell'esistenza delle **forme normali** e si può concludere che il Teorema di Completezza Formale sia un Teorema di Forma Normale.

Sebbene il Teorema possa sembrare banale, in quanto la sua dimostrazione è semplice, il suo contenuto non è nullo né scontato. Un esempio di mancanza funzionale è già presente nella Logica Proposizionale stessa, nonostante la dimostrazione precedente, rimuovendo il vincolo su $n$ e definendo

$$Funz^{\omega} := \{f : \{0,1\}^{\omega} \to \{0,1\}\}$$

[2]Ciò che è stato fatto è esattamente uguale alla costruzione dei minterm e maxterm per una data tavola di verità.

con $\omega = |\mathbb{N}|$, in altre parole l'insieme di tutte le successioni infinite di valori $\{0,1\}$. Se valesse il Teorema di Completezza Funzionale anche in questo caso, si dovrebbe poter affermare che tale insieme è incluso nelle classi di equivalenza scritte nelle classi di equivalenza delle formule scritte con $\omega$ variabili:

$$Funz^{\omega} \subseteq \hat{F}_L^{\omega} = F_L^{\omega} / \equiv$$

Ma in realtà

$$|Funz^{\omega}| = |\mathbb{R}| \geq |\mathbb{N}| = |F_L^{\omega} / \equiv|$$

e pertanto non è valido.

**Forme Normali Canoniche** Le forme normali che abbiamo trovato non sono desiderabili per gli scopi futuri, in quanto sono molto prolisse: ognuno degli $A_{ij}$ menziona tutte le lettere proposizionali $p_1, \cdots, p_n$ (analogamente per le forme congiuntive) ed è infatti chiamata Forma Normale Disgiuntiva (or di and) **Canonica** o **Completa**. Per esempio, siano date tre variabili $p_1$, $p_2$ e $p_3$, si vuole costruire la formula

$$F(t_1, t_2, t_3) = t_1$$

è chiaro che la formula più "corta" è $\hat{F} = t_1$, ma con la forma canonica il risultato è

$$\hat{F} = (p_1 \wedge \neg p_2 \wedge \neg p_3) \vee (p_1 \wedge \neg p_2 \wedge p_3) \vee (p_1 \wedge p_2 \wedge \neg p_3) \vee (p_1 \wedge p_2 \wedge p_3)$$

molto più lunga.

**Insieme funzionalmente completo di connettivi** C'è un'altra nozione, legata alla Completezza Funzionale, che è molto interessante, ossia la nozione di **insieme funzionalmente completo di connettivi**.

**Definizione** (Insieme funzionalmente completo di connettivi)**.** *Un insieme*

$$C \subseteq \{\wedge, \vee, \rightarrow, \neg, \iff, \bot, \top\}$$

*è definito funzionalmente complete se per ogni $F \in F_L$ esiste una $G \in F_L$ scritto usando solo i connettivi in C tale che $F \equiv G$ e $\hat{F} = \hat{G}$, ossia $[F]_{\equiv} = [G]_{\equiv}$.*

Un insieme di connettivi funzionalmente completo è dato dal Teorema di Completezza Funzionale stesso, ossia $C = \{\wedge, \vee, \neg\}$, che è funzionalmente completo grazie al Teorema di Completezza Funzionale. Si può mostrare facilmente che anche $\{\neg, \wedge\}$ e $\{\neg, \vee\}$ sono completi (in quanto, grazie alle leggi di De Morgan, $(A \vee B) = \neg(\neg A \wedge \neg B)$). Questi non sono gli unici insiemi funzionalmente completi: $\{\neg, \rightarrow\}$ (in quanto $(A \wedge B) \equiv \neg(A \rightarrow \neg B)$ e $(A \vee B) \equiv ((\neg A) \rightarrow B)$ e $\{\rightarrow, \bot\}$. In particolare, un insieme composto da un solo connettivo, il `nand`, è anch'esso funzionalmente completo, in quanto ogni altro connettivo è ricostruibile utilizzando solo

$$A \mathbin{\overline{\wedge}} B = \neg(A \wedge B)$$

infatti $\neg A \equiv A \mathbin{\overline{\wedge}} A$ e, ovviamente, $A \wedge B \equiv \neg(A \mathbin{\overline{\wedge}} B)$ per definizione.

## 4.6 Forme Normali

Dopo averle introdotte col Teorema di Completezza Funzionale, possiamo passare ad un esame più approfondito delle Forme Normali. Le Forme Normali Congiuntive e Disgiuntive Canoniche sono esageratamente prolisse. Sarebbe utile conservare l'idea delle Forme Normali con formule più succinte.

### 4.6.1 Forma Normale Negata

Una terza forma normale, chiamata **Negation Normal Form** (NNF) che, per definizione, è l'insieme delle formule che rispettano le seguenti proprietà:

- non contiene $\rightarrow$
- $\neg$ occorre solo applicato a lettere proposizionali

quindi, per esempio

$$p_1 \wedge \neg(p_2 \vee p_3)$$

non è una NNF, mentre

$$p_1 \wedge (\neg p_2 \vee \neg p_3)$$

è una NNF.

**Lemma.** *Ogni formula $F \in F_L$ è equivalente a una $F^N \in NNF$.*

*Dimostrazione.* Per provare il lemma, descriviamo come trasformare $F$ in $F^N$ in un numero finito di passi, dove ogni passo produce una formula equivalente alla precedente. Per ottenere la sequenza si applica ad ogni passo una delle seguenti trasformazioni:

1. $C \rightarrow D$ in $(\neg C) \wedge D$
2. $\neg\neg C$ in $C$
3. $\neg(C \vee D)$ in $\neg C \wedge \neg D$
4. $\neg(C \wedge D)$ in $\neg C \vee \neg D$

Un'osservazione preliminare su questo algoritmo è che si può mostrare che applicando le quattro regole in un ordine qualsiasi si ottiene sempre $F^N \in NNF$. Si mostrerà che partendo da $(1)$ e applicando le altre trasformazioni si possa arrivare a $F^N$: sia data $F \in F_L$. In un numero finito di applicazioni di $(1)$ si ottiene una $F'$ in IFNF (Implication-Free Normal Form), equivalente a $F$.

Si introduce una misura di *distanza*: si definisce $i(E)$ come il numero di connettivi $\rightarrow$ nella formula $E$. Si ha, quindi, che $E \in IFNF \iff i(E) = 0$. Inoltre, se $E \notin IFNF$, allora $E'$ ottenuta applicando $(1)$ su $E$ deve avere $i(E) > i(E')$. È chiaro che $F \equiv F'$ poiché l'applicazione di $(1)$ è semplicemente l'applicazione di una equivalenza eseguita grazie al Teorema di Sostituzione. Quindi, in un certo numero di passi si ha

$$F = F_0 \equiv \cdots \equiv F'(\in IFNF) \equiv F_i \equiv \cdots \equiv F_n(\in NNF)$$

e si nota che le applicazioni di $(2)$, $(3)$ e $(4)$ trattano tutte equivalenze logiche. Ciò che bisogna dimostrare è che la sequenza di trasformazioni termini, ossia l'algoritmo non è infinito ($n \in \mathbb{N}$). Per dimostrare che l'algoritmo termina, sia ora $A \in F_L$ e si definisce

$$l(A) = \text{ numero di occorrenze di connettivi in } A$$

e

$$m(E) = \sum_{\neg A \in E} l(A)$$

dove la notazione $\neg A \in E$ indica che $\neg A$ sia una sottoformula. Si noti che

$$m(E) = 0 \iff E \in NNF$$

in quanto

$$\begin{cases} E \in NNF & \neg A \in E \iff A \in L \iff l(A) = 0 \iff m(E) = 0 \\ m(E) = 0 & \sum_{\neg A \in E} l(A) = 0 \iff l(A) = 0 \iff A \in L \end{cases}$$

In un passaggio eseguito con una delle $(2), (3), (4)$ generando

$$E \rightsquigarrow E'$$

si ha che $m(E) > m(E')$. Se si riesce a dimostrare quest'ultima affermazione, allora si dimostra automaticamente che il procedimento termina in un numero finito di passi.

**(2)**

$$\neg\neg C \rightsquigarrow C$$

In questo caso

$$\begin{aligned} m(\neg\neg C) = \sum_{\neg A \in \neg\neg C} l(A) &= l(\neg C) + \sum_{\neg A \in \neg C} l(A) \\ &= l(C) + 1 + l(C) + \sum_{\neg A \in C} l(A) \end{aligned}$$

mentre

$$m(C) = \sum_{\neg A \in C} l(A) < 1 + 2 * l(c) + \sum_{\neg A \in C} l(A)$$

**(3)**

$$\neg(C \wedge D) \rightsquigarrow \neg C \wedge D$$

In questo caso

$$m(\neg(C \wedge D)) = 1 + \sum_{\neg A \in (C \wedge D)} = l(C) + l(D) + 1 + \sum_{\neg A \in C} l(A) + \sum_{\neg A \in D} l(A)$$

mentre

$$m(\neg C \vee \neg D) = l(C) + l(D) + \sum_{\neg A \in C} l(A) + \sum_{\neg A \in D} l(A) < 1 + \sum_{\neg A \in C} l(A) + \sum_{\neg A \in D} l(A)$$

Analogamente per $(4)$. Data una formula $F \in F_L$, si può costruire una successione finita

$$F = F_0 \equiv \cdots \equiv F'(\in IFNF) \equiv F_i \equiv \cdots \equiv F_n(\in NNF)$$

e ovviamente $F \equiv F_n$, per un qualche $n \in \mathbb{N}$. □

### 4.6.2 Forma Normale Congiuntiva e Disgiuntiva

Le FNC, Forme Normali Congiuntive o CNF, sono un superset delle CNF complete o canoniche.

**Definizione** (Letterale)**.** *Un **letterale** è una formula nella forma $p$ o $\neg p$ per qualche lettera proposizionale $p$ - o è una lettera proposizionale o è la sua negata. La definizione di **letterale opposto** a uno dato è, dato $A$ letterale, l'opposto di $A$*

$$\bar{A} = \begin{cases} \neg p & A = p \\ p & A = \neg p \end{cases}$$

**Definizione** (Clausola)**.** *Una **clausola** è una disgiunzione di un numero finito di letterali come*

$$l_1 \vee l_2 \cdots \vee l_k$$

*dove ogni $l_i$ è un letterale.*

**Definizione** (FNC)**.** *Una **forma normale congiuntiva** è una congiunzione di un numero finito di clausole, come*

$$c_1 \wedge c_2 \wedge \cdots \wedge c_h$$

*dove ogni $c_j$ è una clausola.*

**Definizione** (Clausola vuota)**.** *Una **clausola vuota** $\square$ è definita come la disgiunzione di zero letterali.*

Questo concetto è importante e non va confuso col concetto di **CNF vuota** $\emptyset$, definita come la congiunzione di zero clausole. Prendendo una sequenza di letterali e formule

$$\begin{aligned} \square \\ l_1 \\ l_1 \vee l_2 \\ l_1 \vee l_2 \vee l_3 \\ l_1 \vee l_2 \vee l_3 \vee l_4 \end{aligned}$$

possiamo affermare che la catena di implicazioni dall'alto verso il basso è una catena di tautologie, infatti ogni assegnamento che soddisfa una formula soddisfa anche la successiva, verso il basso - seguendo questa definizione la clausola vuota è insoddisfacibile e pertanto, per noi, è $\square \equiv \bot$.

Per l'insieme di CNF vuote, si ha

$$\begin{aligned} \emptyset \\ c_1 \\ c_1 \wedge c_2 \\ c_1 \wedge c_2 \wedge c_3 \\ c_1 \wedge c_2 \wedge c_3 \wedge c_4 \end{aligned}$$

qui, al contrario, ogni assegnamento che soddisfa una clausola soddisfa anche la precedente, verso l'alto - seguendo questa definizione la CNF vuota è per definizione soddisfacibile ed è, in realtà, soddisfatta da ogni assegnamento e $\emptyset \equiv \top$.

**Definizione** (Forma Normale Disgiuntiva)**.** *Le FND, Forme Normali Disgiuntive o DNF sono definite come disgiunzione di un numero finito di formule*

$$D_1 \vee D_2 \vee \cdots \vee D_n$$

*dove ogni $D_i$ è una congiunzione chiamata* ***co-clausola*** *di un numero finito*

$$l_1 \wedge l_2 \wedge \cdots \wedge l_v$$

*di letterali.*

si tratta, quindi, di una struttura duale rispetto alle CNF. Si noti la terminologia speciale per le CNF: questa terminologia è riservata alle CNF in quanto la dualità tra CNF e DNG verrà spezzata.

**Lemma.** *Ogni formula $F \in F_L$ è logicamente equivalente a una formula $F^c \in CNF$ e a una formula $F^d \in DNF$.*

Questo è già stato dimostrato grazie al teorema di completezza funzionale, che tuttavia utilizzava le CNF e DNF complete: una versione alternativa della dimostrazione è la seguente, per induzione strutturale su $F$.

**Dimostrazione esistenza DNF/CNF**

- **base induttiva:** se $F = p, p \in L$ allora $p = p^c = p^d$.
- **passo induttivo:**

  1. se $F = \neg G$ per ipotesi induttiva si ha $G^c \equiv G$, $G^d \equiv G$. Allora, se

     $$G = C_1 \wedge C_2 \wedge \cdots \wedge C_n$$

     con

     $$C_i = l_{i1} \vee \cdots \vee l_{ik_i}$$

     si ha che

     $$\begin{aligned} \neg G^c &\equiv \neg(C_1 \wedge \cdots \wedge C_n) \\ \text{(DeMorgan)} &\equiv \neg C_1 \vee \neg C_2 \vee \cdots \vee \neg C_n \\ \text{(DeMorgan)} &\rightarrow \neg C_i \equiv \neg l_{i1} \wedge \cdots \wedge \neg l_{ik_i} \\ &\rightarrow \neg G^c \in DNF \end{aligned}$$

     e similmente

     $$\begin{aligned} \neg G^d &\equiv \neg(C_1 \vee \cdots \vee C_n) \\ \text{(DeMorgan)} &\equiv \neg C_1 \wedge \neg C_2 \wedge \cdots \wedge \neg C_n \\ \text{(DeMorgan)} &\rightarrow \neg C_i \equiv \neg l_{i1} \vee \cdots \vee \neg l_{ik_i} \\ &\rightarrow \neg G^d \in CNF \end{aligned}$$

     Quindi, "scambiandosi", esistono sia $(\neg G)^c \equiv (\neg G^d)$ e $(\neg G)^d \equiv (\neg G^c)$.

2. se $F = (G \wedge H)$, per ipotesi induttiva si hanno $G^c \equiv G$, $G^d \equiv G$, $H^c \equiv H$ e $H^d \equiv H$. Si definisce, allora

$$F \equiv F^c \equiv G^c \wedge H^c$$

banalmente, mentre

$$\begin{aligned} F &\equiv F^d \\ &\equiv (G_1 \vee \cdots \vee C_n) \wedge (H_1 \vee \cdots \vee H_k) \\ \text{(Dist. gen.)} &\equiv \vee_{i,j}(G_i \wedge H_j) \\ &\rightarrow \vee_{i=1}^{n} \vee_{j=1}^{k} (G_i \wedge H_i) \in DNF \end{aligned}$$

3. se $F = (G \vee H)$, per ipotesi induttiva si hanno $G^c \equiv G$, $G^d \equiv G$, $H^c \equiv H$ e $H^d \equiv H$. Si può definire, allora

$$F \equiv F^d \equiv G^d \wedge H^d$$

banalmente, mentre

$$\begin{aligned} F &\equiv F^c \\ &\equiv (G_1 \wedge \cdots \wedge C_n) \vee (H_1 \wedge \cdots \wedge H_k) \\ \text{(Dist. gen.)} &\equiv \wedge_{i,j}(G_i \vee H_j) \\ &\rightarrow \wedge_{i=1}^{n} \wedge_{j=1}^{k} (G_i \vee H_i) \in CNF \end{aligned}$$

Si nota, tuttavia, che muoversi tra DNF e CNF causa un notevole allungamento delle formule: per esempio, avendo

$$\vee_{i=1}^{n}(p_i \wedge q_i) \equiv \wedge_{i=1}^{2^n}(l_{i1} \vee \cdots \vee l_{in})$$

quindi c'è una dilatazione esponenziale. Questo è ovviamente un bel problema. Ci sono algoritmi più intelligenti per generare formule DNF o CNF più corte? Purtroppo no.

CAPITOLO 5

# Complessità Computazionale e Deduzione Automatica

In questo momento, per decidere se una formula $F \in F_L$ che menziona $n$ lettere proposizionali diverse sia soddisfacibile o meno, l'algoritmo che conosciamo computa la sua tabella di verità, analizzando quindi $2^n$ possibilità, ossia un numero esponenziale.

Cominciamo definendo cosa sia un problema di decisione:

**Definizione** (Problema di Decisione)**.** *Dato un alfabeto finito* $\Sigma$ *e un sottoinsieme non necessariemente finito di stringhe finite*

$$L \subseteq \Sigma^*$$

*il problema di decisione consiste nel affermare se una precisa stringa finita appartenga o meno a L, che viene chiamato Linguaggio, dove* $\Sigma^*$ *è l'insieme di tutte le stringhe di lunghezza finita, chiamate anche parole, costruite coi simboli di* $\Sigma$*.*

Ad esempio, fissato

$$\Sigma = \{\wedge, \vee, \neg, \rightarrow, (,), p, |\}$$

si possono definire alcuni problemi, per esempio

$$F_L \subseteq \Sigma^*$$

dove il problema risiede nel capire se una certa formula di lunghezza finita risiede in $F_L$ oppure no. Vi sono dei problemi principalmente di natura sintattica, come appunto $F_L$ ma anche $NNF$, $CNF$ e $DNF$. Vi sono problemi in cui la natura semantica è decisiva, per esempio $SAT$, dove

$$w \in \Sigma^* : w \in SAT \iff w \in F_L \wedge w \text{ è soddisfacibile}$$

oppure $TAUTO$, dove una stringa finita appartiene a $TAUTO$ se e solo se è una formula ed è una tautologia, ossia

$$w \in \Sigma^* : w \in TAUTO \iff w \in F_L \wedge \models w$$

e analogamente $UNSAT$, il problema di decidere se una certa formula è insoddisfacibile.

## 5.1 Complessità Computazionale

### 5.1.1 Efficienza

Alcuni dei problemi elencati precedentemente possono essere risolti (decisi) in maniera efficiente, ossia data una *parola w* decidere se appartiene ad un certo linguaggio: esempi di soluzioni

efficienti sono quelle che riguardano tutti i problemi sintattici, risolvibili in tempo al massimo quadratico e in realta anche lineare con alcune tecniche di parsing.

I problemi che riguardano la semantica, invece, sono tipicamente più complessi da risolvere e infatti, per quanto conosciamo fino ad ora, si possono risolvere solo (nel caso peggiore) con un analisi dell'intera tabella di verità della formula, quindi con un numero esponenziale di calcoli. È importante rimarcare nuovamente che benché sussista questa difficoltà, verificare che un singolo assegnamento verifichi la formula è invece un calcolo semplice che richiede un tempo decisamente più contenuto rispetto al problema di decidibilità.

Quindi, vi sono dei problemi decidibili efficientemente, dei problemi decidibili molto difficilmente e verificabili facilmente e dei problemi decidibili molto difficilmente e verificabili difficilmente; da qui parte una gerarchia di classi di complessità infinita, ma a noi interesseranno solo le prime due.

Per essere più precisi, si fissa un modello astratto di computazione, che fornisca la nozione di *passo elementare*, in modo da poter ragionare precisamente sull'efficienza dei problemi. Qualunque modello di computazione che costituisca un modello realistico di calcolatore, con associata una nozione di passo elementare per costruire algoritmi, classifica i problemi nello stesso modo.

**Osservazione** (Tesi di Church-Turing)**.** *Ogni modello ragionevole di computazione è equivalente.*

Uno dei modelli di computazione è quello delle Macchine di Turing (MdT): un'idea della Macchina di Turing è immaginarla come una macchina che lavora su un nastro infinito in lettura e scrittura, mantenendo uno stato e operando su una singola porzione di nasto in lettura utilizzando un programma per muoversi tra gli stati e scrivere sul nastro.

Si può passare ora alla definizione formale delle classi dei problemi:

**Definizione** (Classe $\mathbb{P}$)**.** *La classe dei problemi "efficientemente decidibili" è definita, con una certa ideologia sottesa, come tutti quei problemi che possono essere risolti in tempo polinomiale, ossia esiste una certa Macchina di Turing $T$ e un polinomio $p : \mathbb{N} \to \mathbb{N}$ per i quali per ogni $w \in \Sigma^*$ la computazione della MdT sull'input $w$, denotato $T(w)$ termina entro $p(||w||)$ passi e per ogni $w \in L$ si ha che $T(w)$ accetta il problema e per ogni $w \notin L$ si ha che $T(w)$ non accetta, ossia risolve il problema.*

**Definizione** (Classe $\mathbb{NP}$)**.** *La classe dei problemi "verificabili efficientemente" è definita come tutti quei problemi tali per cui esiste una Macchina di Turing deterministica e due polinomi $p, q : \mathbb{N} \to \mathbb{N}$ tali che per ogni $w \in \Sigma^*$ e ogni $z \in \Gamma^*$, ossia un **certificato**, che si può immaginare prodotto oracolarmente, si ha che $T(w,z)$ termina entro $p(||w||)$ passi e si ha che per ogni $w \in L$ esiste $z \in \Gamma^*$ tale che $||z|| \leq q(||w||)$ (ossia il certificato è sufficientemente corto) e $T(w,z)$ accetta e per ogni $w \notin L$ si ha che per ogni possibile $z \in \Gamma^*$ sufficientemente corto $T(w,z)$ rifiuta, ossia "valida" il certificato.*

**Definizione** (Riducibilità)**.** *Un problema $L_1 \subseteq \Sigma^*$ è **riducibile in tempo polinomiale** a un altro problema $L_2 \subseteq \Gamma^*$ se e solo se esistono una Macchina di Turing $T_{L_1,L_2}$ e un polinomio $p : \mathbb{N} \to \mathbb{N}$ tale che per ogni $w \in L_1$ $T(w)$ trasfroma $w$ in $w' \in \Gamma^*$ in un numero di passi minore o uguale a $p(||w||)$.*

Per indicare la relazione di riducibilità tra due problemi si indica la notazione $L_1 \preceq_p L_2$ per indicare che il primo problema è riducibile polinomialmente al secondo; la nozione di riducibilità è utile poiché rende possibile risolvere istanze del problema $L_1$ "riscrivendole" come se

fossero istanze di $L_2$ (chiaramente questa utilità si verifica quando è più facile risolvere $L_2$ di $L_1$). Un esempio di riducibilità polinomiale è

$$TAUTO \preceq_p UNSAT$$

poiché la trasformazione $w \to \neg w$ è semplice e si sa che $w$ è tautologica se e solo se $\neg w$ è insoddisfacibile.

**Definizione** (Problemi $\mathbb{NP}$-completi)**.** *Un problema appartenente alla classe $\mathbb{NP}_c$ è un problema $L \subseteq \Sigma^*$ se e solo se*

- $L \in \mathbb{NP}$
- *ogni $L' \in \mathbb{NP}$ è tale che $L' \preceq_p L$*

*La seconda proprietà si chiama $\mathbb{NP}$-hardness.*

Come corollario della definizione dei problemi $\mathbb{NP}_c$, si ha che risolvendo polinomialmente un problema di tale classe si dimostra che

$$\mathbb{P} = \mathbb{NP}$$

**Teorema** (di Cook-Levin)**.** *$SAT \in \mathbb{NP}$-completo. $CNFSAT \in \mathbb{NP}$-completo.*

Per dimostrare che $CNFSAT \preceq SAT$, basta realizzare che una formula in CNF è comunque ancora una formula e pertanto la trasformazione è ovviamente polinomiale, essendo la funzione identità. Dimostrare che $SAT \preceq CNFSAT$ è tutt'altra questione benché sia ovvio dal Teorema di Cook; tuttavia, questa proprietà che invece non vale per le DNF, ossia $SAT \not\preceq DNFSAT$, è una buona motivazione per concentrarsi sulle CNF. Si mostra ora, esplicitamente, che SAT si riduce polinomialmente a CNFSAT, conservando non l'equivalenza logica ma la relazione di equisoddisfacibilità.

### 5.1.2 Equisoddisfacibilità

**Definizione** (Equisoddisfacibilità)**.** *Siano $A, B \in F_L$. Le due formule sono equisoddisfacibili se e solo se A è soddisfacibile se e solo se B è soddisfacibile, ossia se A e B sono entrambe soddisfsacibili o entrambe insoddisfacibili.*

#### Esempi

**1** Date le due formule $\neg(A \wedge B)$ e $\neg A \vee \neg B$, è noto che sono equivalenti grazie alle leggi di De Morgan e sono, di conseguenza, anche equisoddisfacibili.

**2** $\neg(A \wedge B)$ e $(\neg A \wedge \neg B)$ non sono equivalenti, infatti l'assegnamento $A = 1$ e $B = 1$ soddisfa solo una delle due, tuttavia sono equisoddisfacibili.

**3** $A \wedge \neg A$ e $\neg A \neg B$ non sono né equivalenti né equisoddisfacibili.
Da questi esempi si può chiaramente notare che l'equivalenza implica l'equisoddisfacibilità mentre il contrario non è affatto verificato.

**4** L'equisoddisfacibilità è un tipo di relazione d'equivalenza, in quanto valgono le proprietà di simmetria, transitività e riflessività.

**5** L'equisoddisfacibilità è una congruenza rispetto ai connettivi, pensati come operazioni? Non lo è. Sia, per esempio $\models A$, quindi $A \in [\top]$ e $B$ soddisfacibile ma non tautologica; sono equisoddisfacibili, in quanto sono entrambi chiaramente soddisfacibili. Se fosse una congruenza, rispetto alle varie operazioni l'equisoddisfacibilità dovrebbe essere mantenuta, invece $\neg A$ è $\bot$, mentre $\neg B$ è ancora soddisfacibile, pertanto non sono equisoddisfacibili.

**6** Le classi di equivalenza delle formule, per esempio su due variabili, sono costruite nelle forme come $F_L^{(2)} / \equiv$, ossia 16 diverse classi ($2^{2^2}$). Per l'equisoddisfacibilità vi saranno unicamente due classi, ossia l'insieme delle formule insoddisfacibili ($[\bot]$) e tutte le rimanenti, ossia tutte quelle almeno soddisfacibili.

### 5.1.3 Riduzione $SAT \preceq CNFSAT$

Sia $A \in F_L$ tale che $A$ sia in Negation Normal Form, ossia $A \in NNF$. Se $A \notin CNF$, allora contiene almeno una sottoformula del tipo $C \vee (D_1 \wedge D_2)$ oppure $(D_1 \wedge D_2) \vee C$. Questo si dimostra semplicemente confermando che $A \in NNF$ ma non $A \in CNF$, quindi la parte che non rispetta la CNF deve essere una disgiunzione di congiunzioni. Trattiamo, d'ora in poi, solo il primo dei due casi, ossia quello in cui nella formula appare la sottoformula $C \vee (D_1 \wedge D_2)$, senza perdita di generalità. Sia data $A \in NNF$ e sia $B = C \vee (D_1 \wedge D_2)$ una sua violazione; per ogni violazione si introduce una nuova lettera proposizionale $a \in L$ che ancora non è presente in $A$. Si definisce ora

$$B' := B[a/D_1 \wedge D_2] \wedge (\neg a \vee D_1) \wedge (\neg a \vee D_2)$$

dove

$$B'' := B[a/D_1 \wedge D_2]$$

è la formula ottenuta rimpiazzando ogni occorrenza di $D_1 \wedge D_2$ con $a$ in $B$. Come prima osservazione, si ha che $(\neg a \vee D_1) \wedge (\neg a \vee D_2)$ è equivalente a $a \rightarrow (D_1 \wedge D_2)$. Si mostra che $B'$ e $B$ sono equisoddisfacibili, ma in genere non sono equivalenti:

$B \in SAT \rightarrow B' \in SAT$. Per ipotesi, dato che $B$ è soddisfacibile, sia $\mathfrak{v} : L \rightarrow \{0,1\}$ tale che $\mathfrak{v}(B) = 1$, ossia $\mathfrak{v} \models B$. Si definisce

$$\mathfrak{v}_a : L \rightarrow \{0,1\} = \begin{cases} \mathfrak{v}_a(p) = \mathfrak{v}(p) & \forall p \in L, p \neq a \\ \mathfrak{v}_a(a) = \mathfrak{v}(D_1 \wedge D_2) \end{cases}$$

L'assegnamento $\mathfrak{v}(a)$ è ben definito, nel senso che non da alla stessa lettera proposizionale due assegnamenti diversi. Si nota che $\mathfrak{v}_a \models a \rightarrow (D_1 \wedge D_2)$, dato che per definizione $\mathfrak{v}_a(a) = \mathfrak{v}(D_1 \wedge D_2)$ e per interpretazione dell'implicazione $0 \rightarrow 0 = 1$ e $1 \rightarrow 1 = 1$; dunque, $\mathfrak{v}_a(B') = \mathfrak{v}_a(B'')$, in quanto la "coda" di $B'$, ossia $(\neg a \vee D_1) \wedge (\neg a \vee D_2)$, ha un assegnamento uguale a 1, ossia $\mathfrak{v}_a(B') = \mathfrak{v}_a(B'') \wedge 1$.

$B$ si può riscrivere come

$$B = B''[D_1 \wedge D_2/a]$$

dato che $a$ non appare in $B$, e quindi

$$B'' = (B[a/D_1 \wedge D_2])[a/a]$$

e grazie al Lemma di Sostituzione si può affermare che i due valori di verità di $B$ e $B''$ sono uguali in quanto i valori di verità di $a$ e $D_1 \wedge D_2$ sono uguali. Ora si può scrivere

$$\begin{aligned} 1 &= \mathfrak{v}(B) \text{ per ipotesi} \\ &= \mathfrak{v}_a(B) \text{ poiché } a \text{ non occore in } B \\ &= \mathfrak{v}_a(B'') \text{ per il lemma di sostituzione} \\ &= \mathfrak{v}_a(B') \end{aligned}$$

ossia l'assegnamento che soddisfa $B$ soddisfa anche $B'$, ossia sono equisoddisfacibili. □

$B' \in SAT \rightarrow B \in SAT$. La sequenza di derivazioni utilizzata per la dimostrazione precedente è ancora buona, tuttavia c'è un vincolo all'inizio, ossia l'assunzione che se $\mathfrak{v}(B) = 1$ allora $\mathfrak{v}_a(B) = 1$; per definizione

$$\mathfrak{v}_a : L \rightarrow \{0,1\} = \begin{cases} \mathfrak{v}_a(p) = \mathfrak{v}(p) & \forall p \in L, p \neq a \\ \mathfrak{v}_a(a) = \mathfrak{v}(D_1 \wedge D_2) \end{cases}$$

che non è la stessa cosa di dire che $B'$ è soddisfacibile, ossia $\mathfrak{v}_a \models B'$, poiché $B'$ potrebbe essere soddisfatto da assegnamenti che non sono nella forma $\mathfrak{v}_a$. Il ragionamento è un po' più complicato.

Si supponga che $B'$ sia soddisfacibile da un assegnamento della forma $\mathfrak{v}_a$, ossia tale che $\mathfrak{v}_a(a) = \mathfrak{v}(D_1 \wedge D_2)$; allora, in questo caso si può tranquillamente ribaltare la catena di uguaglianze precedente:

$$\begin{aligned} 1 &= \mathfrak{v}_a(B') \\ &= \mathfrak{v}_a(B'') \text{ per il lemma di sostituzione} \\ &= \mathfrak{v}_a(B) \text{ poiché } a \text{ non occore in } B \end{aligned}$$

dunque $\mathfrak{v}_a \models B$. Si supponga, ora, che $B'$ sia soddisfatto solo da assegnamenti $\mathfrak{w}$ che non sono della forma $\mathfrak{v}_a$, ossia $\mathfrak{w}(a) \neq \mathfrak{v}(D_1 \wedge D_2)$. Vi sono allora due casi: nel primo $\mathfrak{w}(a) = 0 \neq \mathfrak{v}(D_1 \wedge D_2) = 1$, nel secondo accade il contrario, ossia $\mathfrak{w}(a) = 1 \neq \mathfrak{v}(D_1 \wedge D_2) = 0$ tuttavia quest'ultimo caso è impossibile, poiché $\mathfrak{w}(a \rightarrow D_1 \wedge D_2) = \mathfrak{w}(1 \rightarrow 0) = 0$ e quindi non può soddisfare $B'$. Quindi rimane il primo caso e bisogna mostrare che anche tale assegnamento effettivamente soddisfa $B'$ e non è un assurdo; come informazione di carattere generale utile per risolvere questo problema, sia E un'espressione formata solo da $0, 1, \wedge, \vee$ interpretati come min e max. Il valore di $E$ è 0 o 1. Sia $E'$ ottenuta da $E$ rimpiazzando nessuna o più occorrenze del simbolo 0 con 1. Allora $E \leq E'$. Questo si dimostra affermando che $0, 1, \min, \max$ sono tutte funzioni non decrescenti e $E$ ed $E'$ sono composizioni di funzioni non decrescenti, pertanto sono non decrescenti (un'altra prova è per induzione strutturale su $E$).

Ora, tornando al problema principale, si ha che $\mathfrak{w}(B') = 1$, $\mathfrak{w}(a) = 0$ e $\mathfrak{w}(D_1 \wedge D_2) = 1$. Dato che la formula iniziale era in $NNF$, anche $B'$ e $B$ sono in NNF. Dunque, considerando $\mathfrak{w}(B)$ e $\mathfrak{w}(B'')$ si possono esprimere entrambe come funzioni termine di $B''$, ossia

$$\hat{B}''(\mathfrak{w}(p_1), \cdots, \mathfrak{w}(p_n), \mathfrak{w}(D_1 \wedge D_2))$$

e

$$\hat{B}''(\mathfrak{w}(p_1), \cdots, \mathfrak{w}(p_n), \mathfrak{w}(a))$$

rispettivamente. Come prima osservazione, $\mathfrak{w}(B'')$ e $\mathfrak{w}(B)$ sono considerabili come espressioni costruite su $0, 1, \wedge, \vee$, poiché sono entrambi in $NNF$. Si può concludere, quindi, che

$$\mathfrak{w}(B'') \leq \mathfrak{w}(B) \rightarrow 1 \leq \mathfrak{w}(B) \rightarrow \mathfrak{w}(B) = 1$$

applicando le sostituzioni sui valori di verità di $a$ e $D_1 \wedge D_2$. □

**Osservazione** (Nota finale)**.** *Si sarebbe potuto definire $B'$, alternativamente, come segue:*

$$\begin{aligned} B' &:= B[a/D_1 \wedge D_2] \wedge (a \iff (D_1 \wedge D_2))^c \\ &= B[a/D_1 \wedge D_2] \wedge (\neg a \vee D_1) \wedge (\neg A \vee D_2) \wedge ((D_1 \wedge D_2) \to a)^c \\ &= B[a/D_1 \wedge D_2] \wedge (\neg a \vee D_1) \wedge (\neg A \vee D_2) \wedge (a \vee \neg D_1 \vee \neg D_2) \end{aligned}$$

*Adottando tale definizione per $B'$, allora nella prova che $B'$ soddisfacibile implica $B$ soddisfacibile si sarebbe potuto mostrare che $B'$ è soddisfatto solo da assegnamenti nella forma $\mathfrak{v}_a$, ossia $\mathfrak{v}_a(a) = \mathfrak{v}(D_1 \wedge D_2)$.*

Nel passaggio da $B$ a $B'$ si osserva una dilatazione nella lunghezza della formula, ossia $B'$ è *più lungo* di $B$; data la formula generale

$$B' \to B[a/D_1 \wedge D_2] \wedge (\neg a \vee D_1) \wedge (\neg a \vee D_2)$$

la parte della formula $B$ viene al massimo accorciata, però sia aggiunge una decina di simboli: si può affermare che la lunghezza di $B'$ sia $|B'| = |B| + k$, con $k$ una costante dipendente dal modo in cui si conta la lunghezza (si contano le parentesi come simboli eccetera). Per ogni "violazione" nella formula originale, la formula risultante equisoddisfacibile senza tale violazione è più lunga di $k$ caratteri e, se vi sono $v$ violazioni, dopo $v$ passi la formula risultante sarà CNF e avrà lunghezza $|B| + v * k$, non esponenziale rispetto alla lunghezza iniziale.

La tecnica usata per ridurre $SAT \preceq_p CNFSAT$ che consiste nel rimpiazzare $B$ con $B'$ equisoddisfacibile è ispirata alla riduzione

$$SAT \preceq 3CNFSAT$$

dovuta a Karp. Il passaggio $B \to B'$ è un esempio del cosiddetto "Tseytin's Trick", che si usa per ridurre $F \in F_L$ a una in $3CNFSAT$ ad essa equisoddisfacibile.

**Definizione** (Problemi $3CNFSAT$)**.** $3CNFSAT \subseteq CNFSAT \subseteq F_L$ *è costituito da tutte e sole le CNF dove ogni clausola contiene o esattamente* 3 *letterali.* $3CNFSAT \in \mathbb{NP}$ *ed è addirittura* $\mathbb{NP}$*-completo.*

**Osservazione** (Problemi $2CNFSAT$)**.** $2CNFSAT \in \mathbb{P}$.

### Algoritmo di riduzione a CNF equisoddisfacibili

Sia data $F \in F_L$ e sia $A \in NNF$ $A \equiv F$, che si ottiene in tempo polinomiale rispetto alla lunghezza di $F$. Se $A \in CNF$, allora l'algoritmo termina, altrimenti è necessario individuare la violazione $B$ sottoformula di $A$ e si sostituisce con $B'$ e si ritorna al check su $A \in CNF$.

### Esempi

**1** Sia $A := p \vee (q \wedge r)$. Si trasforma ora in una formula equisoddisfacibile in CNF:

$$A' := (p \vee a) \wedge (\neg a \vee q) \wedge (\neg a \vee r)$$

Queste due formule non sono equivalenti, infatti vi sono assegnamenti che portano a risultati diversi; tuttavia sono equisoddisfacibili.

**2** Sia $A := (p_1 \wedge p_2) \vee (p_2 \wedge q_2) \vee (p_3 \wedge q_3)$. Si ha $A'$ uguale a

$$((p_1 \wedge q_1) \vee (p_2 \wedge q_2) \vee a_3) \wedge (\neg a_2 \vee p_3) \wedge (\neg a_3 \vee q_3)$$

e, applicando di nuovo la trasformazione, si ottiene

$$((p_1 \wedge q_1) \vee a_2 \vee a_3) \wedge (\neg a_3 \vee p_3) \wedge (\neg a_3 \vee q_3) \wedge (\neg a_2 \vee p_2) \wedge (\neg a_2 \vee p_2)$$

e si conclude definendo $A'''$

$$(a_1 \vee a_2 \vee a_3) \wedge (\neg a_3 \vee p_3) \wedge (\neg a_3 \vee q_3) \wedge (\neg a_2 \vee p_2) \wedge (\neg a_2 \vee q_2) \wedge (\neg a_1 \vee p_1) \wedge (\neg a_1 \vee q_1)$$

Data una formula con $n$ letterali si generano $2 * n + 1$ clausole, di cui una con $n$ letterali e $2n$ con due letterali, laddove utilizzando la distributività se ne generavano $2^n$ ognuna con $n$ letterali.

## 5.2 Deduzione Automatica

Vogliamo studiare i problemi $CNFSAT$, che è $\mathbb{NP}$-completo, e il suo complemento $CNFUNSAT$, che è un problema $\mathbb{NP}$-completo, in quanto la riduzione polinomiale $SAT \preceq_p CNFSAT$ è valida e pertanto studiare $CNFSAT$ è uguale a studiare $SAT$. Analogamente si può studiare $TAUTO \preceq_p SAT^c \preceq_p CNFUNSAT$. In realtà siamo nella posizione di studiare $\Gamma \models A$ tramite la risoluzione di $CNFSAT$ e $CNFUNSAT$. Il motivo per cui non studiare invece $DNFSAT$ e $DNFUNSAT$ è che non c'è un algoritmo per "tradurre" una formula arbitraria equisoddisfacibile in DNF in modo che sia sufficientemente corta: i metodi conosciuti allungano esponenzialmente la formula.

**Definizione** (Variazione notazionale)**.** *Date $F_i \in F_L$, le formule*

$$F_1 \wedge F_2 \wedge \cdots \wedge F_k$$

*e*

$$F_1 \vee F_2 \vee \cdots \vee F_k$$

*si possono scrivere senza operatori grazie all'associatività, e inoltre le formule interne nelle formule esterne si possono scambiare ($F_1 \wedge F_2 \equiv F_2 \wedge F_1$) grazie alla commutatività e si possono espandere le singole formule ($F_1 \wedge F_1 \equiv F_1$) grazie all'idempotenza.*

*In questo frangente si possono anche "tralasciare" gli operatori $\wedge$ e $\vee$, chiaramente nel caso in cui ci sia un utilizzo uniforme. La notazione può diventare puramente insiemistica, ossia*

$$\{F_1, F_2, \cdots, F_k\}$$

*per indicare entrambe le formule, specificando il connettivo che le unisce. D'ora in poi una CNF sarò un **insieme** di clausole, dove ogni clausola sarà un insieme di letterali. Per esempio*

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

*diventa*

$$\{\{p, q, \neg r\}, \{q, r, a\}, \{p\}\}$$

*ossia una CNF in forma insiemistica, che contiene clausole in forma insiemistica. Una teoria costruita da CNF è l'insieme di tutte le clausole appartenenti a qualche CNF della teoria. La CNF vuota è $\emptyset$, mentre la clausola vuota $\square$ e la CNF che contiene la clausola vuota avrà la forma $\{\cdots, \square, \cdots\}$.*

Il problema della conseguenza logica di una teoria

$$\Gamma \models A$$

si può ridurre a calcolare la soddisfacibilità di

$$S := \{\Gamma^c \cup \{\neg A\}^c\} \text{ dove } \Gamma^c := \{\gamma^c | \gamma \in \Gamma\}$$

unione di CNF. Si è quindi ridotto il problema $\Gamma \models A$ al problema $S \in CNFSAT$, dove $S$ è un insieme di clausole considerate come insiemi di letterali, eventualmente infinito.

### 5.2.1 Metodi refutazionali

$S$ è soddisfacibile se esiste $\mathfrak{v} : L \to \{0,1\}$ tale che $\mathfrak{v} \models C$ per ogni clausola $C \in S$, in altre parole in ogni $C \in S$ esiste un letterale $l \in C$ tale che $\mathfrak{v} \models l$.

Se l'obiettivo è dimostrare che $S$ è insoddisfacibile, una strategia può essere ampliare $S$ in un nuovo insieme $S \subseteq S'$ in modo tale che $S'$ è logicamente equivalente o almeno equisoddisfacibile a $S$. Ad esempio, se ampliando iterativamente $S$ in $S'$, $S''$ fino a $S^{(u)}$ e alla fine la clausola vuota $\square$ appartiene a $S^{(u)}$ allora quest'ultimo è insoddisfacibile, e quindi anche $S$ lo è. Questa idea può essere sfruttata disegnando **metodi refutazionali**, ossia metodi che hanno l'obiettivo di provare l'insoddisfacibilità di un insieme di clausole, in questo frangente, ma più in generale anche di una formula o una teoria, i quali sono basati sulla regola di inferenza chiamata **principio di risoluzione**, il quale è utile per un tipo di calcolo particolarmente adatto a essere automatizzato, ossia SAT solver e Theorem Prover.

**Definizione** (Principio di Risoluzione)**.** *Date due clausole $C_1$ e $C_2$ si dice che $D$ è la **risolvente** di $C_1$ e $C_2$ sul pivot $\ell$ se e solo se*

- $\ell \in C_1$
- $\bar{\ell} \in C_2$
- $D := (C_1 \setminus \{\ell\}) \cup (C_2 \setminus \{\bar{\ell}\})$

*e si scriverà $D = \mathbb{R}(C_1, C_2; \ell, \bar{\ell})$.*

#### Esercizio

Mostrare che $(C_1 \setminus \{\ell\}) \cup (C_2 \setminus \{\bar{\ell}\})$ può essere diverso da $(C_1 \cup C_2) \setminus \{\ell, \bar{\ell}\}$.

#### Esempio

Sia $C_1 := \{x, y, \neg t\}$ e $C_2 := \{u, \neg y, t\}$; si ha $D_1 = \mathbb{R}(C_1, C_2; y, \neg y) = \{x, \neg t, u\}$ e $D_2 = \mathbb{R}(C_1, C_2; \neg t, t) = \{x, y, y\}$.

**Lemma** (di correttezza della risoluzione)**.** *Sia $D = \mathbb{R}(C_1, C_2; \ell, \bar{\ell})$, allora*

$$\{C_1, C_2\} \models D$$

*Dimostrazione.* Bisogna dimostrare che ogni $\mathfrak{v}$ tale che $\mathfrak{v} \models C_1$ e $\mathfrak{v} \models C_2$ implica $\mathfrak{v} \models D$. Sia allora $\mathfrak{v}$ tale che $\mathfrak{v} \models C_1$ e $\mathfrak{v} \models C_2$, allora $\mathfrak{v} \models m$ e $\mathfrak{v} \models n$ per due letterali $m \in C_1$ e $n \in C_2$; se fosse il caso che $m = \ell$ e $n = \bar{\ell}$ allora $\mathfrak{v}(\ell) = 1$ e $\mathfrak{v}(\bar{\ell}) = 1$, impossibile. Allora,almeno uno tra $m$ e $n$ è tale che $m \neq \ell$ o $n \neq \ell$ e assumendo senza perdita di generalità $m \neq l$ si ha $m \in D$, dunque $\mathfrak{v}(D) = 1$. $\square$

**Corollario.** *Se* $\mathbb{R}(C_1,C_2;\ell,\bar{\ell}) = \square$ *allora* $\{C_1,C_2\}$ *è insoddisfacibile, in quanto* $\{C_1,C_2\} \models \square$.

**Corollario.** *Sia S un insieme di clausole e sia* $S := S_0,S_1,\cdots,S_k$ *una successione di insiemi tale che per ogni* $i = 0,\cdots,k-1$ *si ha che* $S_{i+1}$ *si ottiene unendo a* $S_i$ *una o più risolventi di clausole in* $S_i$ *e che* $\square \in S_k$. *S è insoddisfacibile.*

L'obiettivo è mostare che i calcoli refutazionali basati su applicazioni ripetute della risoluzione sono corretti e completi (refutazionalmente). In genere, per un calcolo logico si studiano infatti le due seguenti proprietà:

**Definizione** (Correttezza)**.** *Un calcolo si definisce* ***corretto*** *se i certificati che produce testimoniano il vero, ossia sono corretti, in altre parole non produce certificati fasulli.*

Per esempio, nel frangente attuale $S_0,S_1,\cdots,S_k \ni \square$ è un certificato corretto dell'insoddisfacibilità di $S = S_0$.

**Definizione** (Completezza)**.** *Un calcolo è completo se non omette alcun certificato.*

Nella situazione attuale vuol dire che se $S$ è insoddisfacibile, allora esiste $S_0,S_1,\cdots,S_k$ tale che si produce $\square \in S_k$.

Prima di mostrare che il calcolo refutazionale è sia corretto (anche se il lemma di correttezza è già stato esplicitato) e completo, è necessario prepararsi la strada, poiché non sarà banale.

**Teorema** (Teorema di Completezza del principio di risoluzione (o di Robinson))**.** *Un insieme S di clausole è insoddisfacibile se e solo se* $\square \in R^*(S)$, *dove* $R^*(S)$ *è definito come segue:*

$$
\begin{aligned}
R(S) &:= S \cup \{D : D = \mathbb{R}(C_1,C_2;\ell,\bar{\ell}) C_1,C_2 \in S \wedge \ell \in C_1 \wedge \bar{\ell} \in C_2\} \\
R^2(S) &= R(R(S)) \\
&\cdots \\
R^{t+1}(S) &:= R(R^t(S)) \\
&\cdots \\
R^*(S) &= \cup_{i\in\omega} R^i(S)
\end{aligned}
$$

*dato* $R^0(S) := S$.

Il calcolo $R^*$ è un metodo *a forza bruta*, applica il Principio di Risoluzione calcolando ciecamente tutte le risoluzioni possibili e non c'è da sperare che faccia molto meglio delle tavole di verità.

$\square \in R^*(S) \rightarrow S$ *insoddisfacibile (Correttezza del calcolo R).* Si supponga $\square \in R^*(S)$, allora c'è un $i \in \omega$ tale per cui, per definizione, $\square \in R^i(S)$ e pertanto $R^i(S)$ è insoddisfacibile, per definizione di clausola vuota; $R^i(S)$ è equivalente a $R^{i-1}(S)$ per il lemma di correttezza, arrivando fino a $R^0(S) = S$, che è pertanto insoddisfacibile. $\square$

$S$ *insoddisfacibile* $\rightarrow \square \in R^*(S)$ *(Completezza Refutazionale del calcolo R).* Dato che $S$ è insoddisfacibile, per il Teorema di Compattezza esiste un $S_{fin} \subseteq_\omega S$ finito e $S_{fin}$ è insoddisfacibile; bisogna capire come sia fatto $S_{fin}$: l'insieme finito $S_{fin}$ contiene un numero finito di lettere proposizionali e si definisce, come è stato fatto precedentemente, $Var(S_{fin}) \subseteq \{p_1,p_2,\cdots,p_n\}$ per qualche $n \in \omega$, $p_i \in L$. La notazione $C_L^n$ indica l'insieme di tutte le clausole scrivibili sulle prime $n$ lettere proposizionali, ossia $\{p_1,p_2,\cdots,p_n\}$ e si ha $S_{fin} \subseteq C_L^n$. Si osserva che $C_L^0 = \{\square\}$.

A fortiori, dato che $S_{fin} \subseteq C_L^n$ si ha $S_{fin} \subseteq (C_L^n \cap S) \subseteq (C_L^n \cap R^*(S))$ e pertanto $C_L^n \cap R^*(S)$ è insoddisfacibile, dato che $S_{fin}$ è insoddisfacibile.

Se si riesce a dimostrare che per ogni $k = n, \cdots, 1, 0$ si ha

$$C_L^k \cap R^*(S) \text{ è insoddisfacibile}$$

allora si ha che per $k = 0$

$$C_L^0 \cap R^*(S) \text{ è insoddisfacibile}$$

poichè se si dimostra che $C_L^0 \cap R^*(S)$ insoddisfacibile, allora $\square \in R^*(S)$:

$$C_L^0 \cap R^*(S) = \begin{cases} \emptyset & \rightarrow C_L^0 \cap R^*(S) \text{soddisfacibile, assurdo.} \\ \square & \text{unico caso possibile} \end{cases}$$

dunque $\square \in R^*(S)$. Per dimostrare che per ogni $k = n, \cdots, 1, 0$

$$C_L^k \cap R^*(S) \text{ è insoddisfacibile}$$

si usa l'induzione decrescente su $k$, ossia l'induzione aritmetica su $t = n - k$; se $k = n \rightarrow t = 0$ e se $k = 0 \rightarrow t = n$. La base dell'induzione è "regalata" dal Teorema di Compattezza, ossia che $C_L^k \cap R^*(S)$ sia insoddisfacibile è dovuto all'esistenza $S_{fin}$ insoddisfacibile. Si assume l'asserto vero per $n-0, n-1, \cdots, n-k-1$ e si prova per $t = n-k$, ossia

$$C_L^n \cap R^*(S), C_L^{n-1} \cap R^*(S), \cdots, C_L^{k+1} \cap R^*(S)$$

sono insoddisfacibili e si vuole dimostrare per $C_L^k \cap R^*(S)$ sia insoddisfacibile.

Per assurdo, si assume $C_L^k \cap R^*(S)$ soddisfacibile, dunque esiste $\mathfrak{v} \models C$ per ogni $C \in C_L^k \cap R^*(S)$; si definiscono ora

$$\mathfrak{v}^+ : L \rightarrow \{0,1\} \quad \mathfrak{v}^+(p_{k+1}) = 1$$

$$\mathfrak{v}^- : L \rightarrow \{0,1\} \quad \mathfrak{v}^-(p_{k+1}) = 0$$

mentre per ogni altra $i$ $\mathfrak{v}(i) = \mathfrak{v}^+(i) = \mathfrak{v}^-(i)$; si noti che $p_{k+1} \notin C_L^{(k)}$. Per ipotesi induttiva esiste $C_1 \in C_K^{k+1} \cap R^*(S)$ tale che $\mathfrak{v}^+(C_1) = 0$; analogamente esiste $C_2 \in C_L^{k+1} \cap R^*(S)$ tale che $\mathfrak{v}^-(C_2) = 0$, in quanto è insoddisfacibile.

La lettera proposizionale $p_{k+1}$ occorre in $C_1$ e più precisamente come letterale $\neg p_{k+1}$, infatti $p_{k+1}$ non può occorrere come letterale positivo poiché $\mathfrak{v}^+(p_{k+1}) = 1$ per definizione e dunque $\mathfrak{v}^+(C_1) = 1$, assurdo. Deve, però, apparire nel letterale $\neg p_{k+1}$, poiché altrimenti si ottiene che $p_{k+1}, \neg p_{k+1} \notin C_1$ dunque $C_1 \in C_L^k \cap R^*(S)$ e dunque $\mathfrak{v}(C_1) = 1$. Analogamente, $p_{k+1}$ occorre in $C_2$ e più precisamente come letterale $p_{k+1}$ e la prova induttiva è identica, mutandis mutandis.

Dunque, esiste $D = R(C_1, C_2; \neg p_{k+1}, p_{k+1})$, ossia

$$D = (C_1 \setminus \{\neg p_{k+1}\}) \cup (C_2 \setminus \{p_{k+1}\})$$

e $p_{k+1}, \neg p_{k+1} \notin D$ e $D \in C_L^k \cap R^*(S)$ e $\mathfrak{v}(D) = 1$. Vi sono due casi: il primo è che $\mathfrak{v}$ soddisfi qualche letterale in $C_1 \setminus \{p_{k+1}\}$, ma allora $\mathfrak{v}(C_1) = 1$ e $\mathfrak{v}^+(C_1) = 1$, assurdo. Il secondo caso è che $\mathfrak{v}$ soddisfi qualche letterale in $C_2 \setminus \{p_{k+1}\}$, e allora $\mathfrak{v}^-(c_2) = 1$, assurdo. Non essendoci altri casi, si è raggiunta la contraddizione che conclude la dimostrazione per assurdo, dunque $C_L^k \cap R^*(S)$ è insoddisfacibile per ogni $k$, che chiude la prova per induzione; dunque, per ogni $k = n, n-1, \cdots, 1, 0$, $C_L^k \cap R^*(S)$ è insoddisfacibile e $C_L^0 \cap R^*(S)$ anche, pertanto $C^0 \cap R^*(S) = \{\square\}$ e $\square \in R^*(S)$. $\square$

**Osservazione.** *Se S è un insieme finito di clausole, la costruzione di $R^*(S)$ costituisce una procedura di decisione, cioè sia che S sia insodd. sia che S sia sodd. termina in tempo finito, dando la risposta corretta. Infatti, se S è finito menziona solo un numero finito $n = |Var(S)|$ di lettere proposizionali diverse e i letterali scrivibili su $\{p_1, \cdots, p_n\}$ sono $2 * n$, dunque in S occorrono al più $2 * n$ letterali diversi; le clasuole scrivibili su $2 * n$ letterali sono al più $2^{2*n}$ dunque in S occorrono al più $2^{2*n}$ clausole. Si nota, ora, che la risoluzione non introduce mai nuovi letterali, dunque la sequenza $R^{(0)}(S) \subseteq R(S) \subseteq \cdots R^k(S) \subseteq \cdots$ è tale che che ogni $R^i(S)$ è un sottoinsieme delle $2^{2*n}$ clausole scrivibili su $\{p_1, \cdots, p_n\}$, dunque esiste un t tale che $R^t(S) = R^{t+1}(S)$, poiché la sequenza non può crescere all'infinito; pertanto*

$$R^t(S) = R^{t+1}(S) = \cdots = R^*(S)$$

*Se si trova $\square \in R^i(S)$, si può concludere che S sia insoddisfacibile. Se, al contrario, $R^t(S) = R^{t+1}(S)$ e $\square \notin R^t(S)$, si può concludere che S sia soddisfacibile.*

**Osservazione.** *Dato che $S_{fin}$ è finito, esiste $t \in \omega$ tale che $R^*(S_{fin}) = R^t(S_{fin}) = R^{t+1}(S_{fin})$, tuttavia questo non fornisce una procedura di decisione per gli S infiniti, ma esclusivamente di semidicesione, in quanto in genere non si sa scegliere $S_{fin}$ e il meglio che si può fare è descrivere una successione infinita*

$$S_1 \subseteq S_2 \subseteq \cdots \subseteq S_k \subseteq \cdots$$

*di sottoinsiemi finiti di S tali che $\bigcup S_i = S$ e per ogni i si calcola $R^*(S_i)$: se $\square \in S_i$ allora S insoddisfacibile, altrimenti si aumenta i e si procede al passo successivo.*

La Completezza Refutazionale non è la Completezza *tout-court*. È qualcosa che è più debole, e in realtà proprio per questo è una proprietà desiderabile dal punto di vista computazionale.

**Definizione** (Deduzione per Risoluzione)**.** *Una deduzione per risoluzione di una clausola C da un insieme di clausole S, indicato*

$$S \vdash_R C$$

*è una sequenza finita di clausole*

$$C_1, C_2, \cdots, C_n$$

*tale che*

- $C_n = C$
- $\forall C_i \quad C_i \in S$ *oppure* $C_i = R(C_j, C_k, \ell, \bar{\ell})$

In particolare, una deduzione per risoluzione della clausola vuota ($S \vdash_R \square$) è detta **refutazione** di *S*.

**Teorema** (Teorema di Completezza Refutazionale)**.** *Un insieme di clausole S è insoddisfacibile se e solo se $S \vdash_R \square$.*

Al momento, la refutazione al momento la sappiamo costruire solo tramite il metodo $R^*(S)$.

**Esempio**

$$\{\{a,b,\neg c\},\{a,b,c\},\{a,\neg b\}\} \models \{a\}?$$

Si costruisce una refutazione: inizialmente, si trasforma il problema in un problema di insoddisfacibilità, ossia

$$\{\{a,b,\neg c\},\{a,b,c\},\{a,\neg b\},\{\neg a\}\} \text{ è soddisfacibile?}$$

Non è soddisfacibile, poiché

$$(\{a,b,\neg c\},\{a,b,c\}) \vdash_R \{a,b\}$$
$$(\{a,b\},\{a,\neg b\}) \vdash_R \{a\}$$
$$(\{a\},\{\neg a\}) \vdash_R \square$$

Genericamente, chiedersi se $\Gamma \models A$ è uguale a chiedersi se $\Gamma, \neg A$ sia insoddisfacibile, che è uguale a $\Gamma^c, (\neg A)^c \vdash_R \square$ per Completezza Refutazionale, ma che è diverso da $\Gamma^c \vdash_R A^c$; è questa la debolezza della completezza refutazionale rispetto alla Completezza tout court ($\Gamma \models A \iff \Gamma^c \vdash_c A$). Quindi, per un calcolo refutazionale $R$ per il quale vale la Completezza Refutazionale si ha

$$\Gamma \models A \iff \Gamma, \neg A \text{ è insoddisfacibile} \iff \Gamma^C, (\neg A)^C \vdash_R \square \rightarrow\!\!\!\!\leftarrow \Gamma^C \vdash_R A^C$$

Si consideri, per esempio, l'insieme di clausole

$$\{\{b\},\{\neg b\},\{\neg a\}\} \vdash_R \square$$

ma per il fatto che la Risoluzione non introduce mai nuovi letterali, non si potrà mai a provare

$$\{\{b\},\{\neg b\}\} \vdash_R \{a\}$$

### 5.2.2 Sistemi Assiomatici (Calcoli alla Hilbert)

I Sistemi Assiomatici sono un tipo di calcolo tradizionale completo tout court. Hanno una formalizzazione tra le più semplici, ma non sono in particolar modo adatti alla ricerca *human-oriented* e nemmeno alla ricerca di prove tramite algoritmi.

**Definizione** (Calcoli alla Hilbert)**.** *Dato un insieme di assiomi (che sono tautologie della Logica), come per esempio il seguente, che è corretto e completo per la Logica Proposizionale classica*

$$\begin{cases} A \rightarrow (B \rightarrow A) \\ (A \rightarrow (B \rightarrow C)) \rightarrow ((A \rightarrow B) \rightarrow (A \rightarrow C)) \\ (\neg B \rightarrow \neg A) \rightarrow (A \rightarrow B) \end{cases}$$

*con $A,B,C \in F_L$ e avendo regole di inferenza come il modus ponens*

$$\frac{A \qquad A \rightarrow B}{B}$$

*una prova di A da una teoria $\Gamma$ nel calcolo definito Calcolo H o Calcolo alla Hilbert, ossia*

$$\Gamma \vdash_H A$$

*è una successione finita di formule*

$$A_1, A_2, \cdots, A_u$$

*tale che:*

1. $A_u = A$
2. *ogni* $A_i$ *per* $i = 1, \cdots,$ *u è tale che:*
   (a) $A_i \in \Gamma$
   (b) $A_i$ *è un'istanza di assioma*
   (c) *esistono* $j,k < i$ *tali che* $A_j = A_k \to A_i$, *quindi*

$$\frac{A_k \qquad A_k \to A_i}{A_i}$$

**Teorema** (Completezza Forte del Calcolo H)**.** $\Gamma \models A \iff \Gamma \vdash_H A$, *anche per teorie* $\Gamma$ *infinite.*

Il Calcolo H non è particolarmente adatto alla deduzione automatica, come sottolineato precedentemente. I vari tipi di calcolo corrispondono ad esigenze diverse: quando la necessità è la deduzione automatica, vi sono ottime ragioni per preferire il Calcolo Refutazionale. Diamo ora qualche evidenza del perché non sia così saggio utilizzare il Calcolo H. Quest'ultimo è semplice dal punto di vista concettuale, ma tirar fuori prove è più complicato.

### Differenze tra i due Calcoli

Verificare se $\neg\neg A \models A$ è vera.

**Calcolo R** Bisogna inizialmente trasportare $A \in F_L$ in lettere proposizionali:

$$\neg\neg p \models p$$

con $p \in L$; nel Calcolo H questo passaggio non è necessario, si potrà tranquillamente ragionare sulle metavariabili. Si studia l'insoddisfacibilità di

$$\{\neg\neg p, \neg p\} \models \bot$$

che si traduce, in refutazione, in

$$\{\{p\}, \{\neg p\}\} \vdash_R \square?$$

si ha: $\mathbb{R}(\{p\}, \{\neg p\}; p, \neg p) = \square$, pertanto la prima formula è vera.

**Calcolo H** Per il Calcolo di Hilbert bisogna "inventarsi" una dimostrazione; si comincia a scrivere

$$\begin{aligned} \neg\neg A \to (\neg\neg\neg\neg A \to \neg\neg A), \neg\neg A (\in \Gamma) \\ \therefore \neg\neg\neg\neg A \to \neg\neg A, (\neg\neg\neg\neg A \to \neg\neg A) \to (\neg A \to \neg\neg\neg\neg A) \\ \therefore \neg A \to \neg\neg\neg\neg A, (\neg A \to \neg\neg\neg\neg A) \to (\neg\neg A \to A) \\ \therefore \neg\neg A \to A, \neg\neg A \\ \therefore A \end{aligned}$$

Il calcolo H non presenta un calcolo estremamente meccanico.

**Esercizio** Dimostrare che $\models A \to A$

Si prenda una formula $B$ soddisfacibile, già provata, ossia $\models_H B$.

$$\frac{B \qquad B \to (A \to B)}{A \to B}$$

e

$$\frac{A \to (B \to A) \qquad (A \to (B \to A)) \to (A \to A)}{(A \to B) \to (A \to A)}$$

da cui si può concludere

$$(A \to A)$$

### 5.2.3 Procedura refutazionale di Davis-Putnam

Si supponga che sia stato definito un insieme di clausole

$$S = \{\{p,q\},\{p,\neg q\},\{\neg p,q\},\{\neg p,\neg q\}\}$$

e ci si chiede se sia refutabile. Si ha

$$\dfrac{\dfrac{p,q \qquad p,\neg q}{p} \qquad \dfrac{\neg p,q \qquad \neg p,\neg q}{\neg p}}{\square}$$

Utilizzando il calcolo refutazionale $R^*(S)$ si utilizzano molti più passaggi: il problema che si vuole affronare e risolvere, per quanto possibile, è designare una tecnica che permetta di mantenere la completezza refutazionale selezionando, in qualche modo, i risolventi da calcolare.

**Definizione** (Sussunzione)**.** *Una clausola $C_1$ **sussume** $C_2$ se e solo se $C_1 \subset C_2$, ossia è un insieme proprio di $C_2$. In termini logici, si ha $\models C_1 \to C_2$, ossia $\{C_1, C_2\} \equiv \{C_1\}$.*

**Definizione** (Regola di Sussunzione)**.** *Sia $S$ un insieme di clausole e sia $S'$ ottenuto da $S$ eliminando tutte le clausole sussunte. Allora si può concludere che $S \equiv S'$.*

**Osservazione.** *$S'$ non contiene sussunte, ossia non è ulteriormente riducibile per sussunzione.*

**Definizione** (Clausola Banale)**.** *Una clausola $C$ è detta **banale** (trivial) se contiene sia $L$ che il sua opposto $\bar{L}$ per qualche letterale $l$, ossia $C \equiv \top$ o $\models C$.*

**Definizione** (Regola di rimozione banali)**.** *Rimuovendo da $S$ tutte le banali costruendo $S'$, si ha $S \equiv S'$, e $S'$ non contiene banali.*

**Osservazione.** *Risolvendo tra loro le clausole non banali $C_1$ e $C_2$, allora in $D = R(C_1, C_2; \ell, \bar{\ell})$ si ha che $\ell \notin D$ e $\bar{\ell} \notin D$, cioè $(C_1 \setminus \{\ell\}) \cup (C_2 \setminus \{\bar{\ell}\}) = (C_1 \cup C_2) \setminus \{\ell, \bar{\ell}\}$*

**Osservazione.** $R(\{a,\neg a\},\{a,\neg a\}; a, \bar{a}) = (\{a,\neg a\} \setminus \{a\}) \cup (\{a,\neg a\} \setminus \{\neg a\}) = \{\neg a\} \cup \{a\} = \{a,\neg a\}$

**Definizione** (Passo di DPP)**.** *Dato un input $S$ finito, già ripulito di banali e sussunte, un passo di David-Putnam procedure si articola nei seguenti micropassi:*

1. *Scegliere una lettera proposizionale $p \in L$ tra quelle che appaiono in $S$. $p$ sarà detto il pivot del passo.*

2. *$S'$ è l'insieme delle p-esonerate, ossia l'insieme delle clausole in S in cui non occorre p come lettera proposizionale, quindi né p né $\neg p$.*

3. *$S''$ è l'insieme delle p-risolventi ed è l'insieme delle clausole ottenute per risoluzione sul pivot p in $S \setminus S'$.*

4. *$S'''$ è l'insieme $S' \cup S''$ ripulito, che è l'output del passo di DPP.*

**Esempi di DPP**

**1** Sia $S_0 = \{\{a,b,\neg c\},\{a,\neg b,\neg d\},\{a,\neg b,\neg c,\neg d\},\{\neg a,d\},\{\neg a,\neg c,\neg d\},\{c\}\}$.

- $S_0$ è ripulito. Si sceglie, come pivot, $c$. Le $c$-esonerate sono la seconda e la terza clausola. Le $c$-risolventi sono, per esempio, $\{a,b\},\{a,\neg b,\neg d\},\{\neg a,\neg d\}$. L'output ripulito è
$$\{\{a,\neg b,d\},\{\neg a,d\},\{a,b\},\{a,\neg b,\{d\},\{\neg a,\neg d\}\}$$
- Per il secondo passo, si sceglie come pivot $a$. Le $a$-esonerate sono $\emptyset$, mentre le $a$-risolventi sono
$$\{\neg b,d\},\{\neg b,d,\neg d\},\{b,d\},\{\neg b,d\},\{b,\neg d\},\{\neg b,\neg d\}$$
L'output ripulito è:
$$\{\{\neg b,d\},\{b,d\},\{b,\neg d\},\{\neg b,\neg d\}\}$$
- Come terzo passo si prende come pivot $b$. Le $b$-esonerate sono
$$\emptyset$$
e le $b$-risolventi sono
$$\{d\},\{d,\neg d\},\{\neg d,d\}\{\neg d\}$$
mentre l'output ripulito è
$$\{\{d\},\{\neg d\}\}$$
- Per l'ultimo passo si è obbligati a scegliere come pivot $d$. Si ha che le $d$-esonerate è nuovamente l'insieme vuoto $\emptyset$, mentre le $d$-risolventi sono
$$\square$$
e l'output è
$$\{\square\}$$

Dato che è stata ottenuta la clausola vuota, si dichiara la clausola iniziale $S_0$ insoddisfacibile.

**2** Sia $S_0 = \{\{a,\neg b,c\},\{b,\neg c\},\{a,c,\neg d\},\{\neg b,\neg d\},\{a,b,d\},\{\neg a,d,b\}\{b,\neg c,d\}\{b,c,d\}\}$.

- Ripulito, si trovano le clausole
$$\{\{a,\neg b,c\},\{b,\neg c\},\{a,c,\neg d\},\{\neg b,\neg d\},\{a,b,d\},\{\neg a,d,b\}\}$$
Si sceglie, come pivot, $b$. le $b$- esonerate sono
$$\{a,c,\neg d\}$$
e le $b$-risolventi sono:
$$\{a,c,\neg c\},\{a,c,d\},\{a,\neg a,c,d\},\{\neg c,\neg d\},\{a,\neg d,d\},\{\neg a,\neg d,d\}$$
e l'output è
$$S_1 := \{a,c,d\},\{\neg c,\neg d\},\{a,c,\neg d\}$$

- Al secondo passo si sceglie, come pivot, $c$. Non vi sono $c$-esonerate, mentre le $c$-risolventi sono
$$\{a, d, \neg d\}, \{a, \neg d, \neg d\}$$
e l'output è
$$S_2 := \{a, \neg d\}$$
- Si sceglie, come pivot, $a$. Non vi sono $a$-esonerate e non vi sono $a$-risolventi; L'output di questo passaggio è l'insieme vuoto $\emptyset$ e si dimostra che questo implica che $S_0$ è soddisfacibile. La lettera proposizionale $d$ non è mai stata scelta come lettera pivot, quindi è una *fuoriuscita*.

**Definizione.** *In un'esecuzione di DPP le lettere che non sono mai pivot si definiscono fuoriuscite.*

Delineamo, prima di affrontare la dimostrazione, la strategia che con questi dati costruisce un assegnamento che soddisfa $S$. Sia $\mathfrak{v} : L \to \{0,1\}$ tale che $\mathfrak{v} \models S_0$. Si definisce

$$\mathfrak{v} := \begin{cases} a & \mathfrak{v}(a) = 1 \\ b & \mathfrak{v}(b) = 1 \\ c & \mathfrak{v}(c) = 1 \\ d & \mathfrak{v}(d) = 0 \text{ per convenzione per le fuoriuscite, arbitrario} \end{cases}$$

Si definisce $\mathfrak{v}(a)$ come un valore che soddisfi $S_2$, ossia $\{a, \neg d\}$. Si definisce $\mathfrak{v}(c)$ come un valore che soddisfi $S_1$ (si è liberi, in quanto tutte le clausole sono già soddisfatte dagli altri assegnamenti). Si definisce $\mathfrak{v}(b)$ come un valore che soddisfi $S_0$.

### Teorema di completezza (e correttezza) refutazionale di DPP

**Teorema.** *Un insieme **finito** $S$ di clausole nelle lettere proposizionali $p_1, p_2, \cdots, p_n$ $(:= Var(S))$ è insoddisfacibile se e solo se entro al più n passi si ottiene la clausola vuota $\square$. $S$ è soddisfacibile se e solo se entro al più n passi si ottiene l'insieme vuoto di clausole $\emptyset$. Non vi sono altri modi di terminare.*

*Terminazione.* Sia $DPP(S) := S_0, S_1, \cdots, S_k$ una sequenza di clausole, dove $S_0$ è la ripulitura di $S$ e $S_i$ è ottenuto da $S_{i-1}$ attraverso un passo di DPP sul pivot $q_i \in \{p_1, \cdots, p_n\}$. Si osserva che $S_i$ non conterrà più alcuna occorrenza di $q_i$ e, quindi, dopo al più $k \leq n$ passi si ottiene $Var(S_k) = \emptyset$, dunque o $S_k = \{\square\}$ oppure $S_k = \emptyset$. $\square$

*Correttezza e Completezza Refutazionale di DPP.* La correttezza di DPP si può specificare in due modi:

$$\begin{cases} S_k = \{\square\} \implies S_0 \text{ insoddisfacibile} \\ S_0 \text{ soddisfacibile} \implies S_k = \emptyset \end{cases}$$

La prima delle due versioni è "gratuitamente" provata dal Lemma di Correttezza, ancora valido, poiché il calcolo dei risolventi non è cambiato (pertanto la correttezza è provata.)

La completezza di DPP si può specificare nei due modi rimanenti:

$$\begin{cases} S_0 \text{ insoddisfacibile} \implies S_k = \{\square\} \\ S_k = \emptyset \implies S_0 \text{ soddisfacibile} \end{cases}$$

Ci basta dimostrare quest'ultima parte, ossia che se $S_k = \emptyset$ allora $S_0$ è soddisfacibile: questo fatto è chiamato anche *Lemma di Model Building*, poiché la dimostrazione si occupa di mostrare come costruire l'assegnamento che soddisfa $S$.

Si dimostra per induzione decrescente su $k$, ossia induzione crescente su $t = k - i$:

**base**

$$S_k = \emptyset \rightarrow S_k \text{ soddisfacibile.}$$

**passo induttivo** si assume che $S_{i+1}$ soddisfacibile e si mostra $S_i$ soddisfacibile, ossia esiste $\mathfrak{v} : L \rightarrow \{0,1\}$ tale che $\mathfrak{v} \models S_i$. Dato $\mathfrak{v} : L \rightarrow \{0,1\}$ tale che $\mathfrak{v} \models S_{i+1}$ si definiscono le due varianti

$$\mathfrak{v}^+ : L \rightarrow \{0,1\} \quad \mathfrak{v}^+(p) = 1$$

$$\mathfrak{v}^- : L \rightarrow \{0,1\} \quad \mathfrak{v}^-(p) = 0$$

dove $p$ è il pivot di $S_i \rightsquigarrow S_{i+1}$, mentre

$$\mathfrak{v}(q) = \mathfrak{v}^+(q) = \mathfrak{v}^-(q) \forall q \in Lq \neq p$$

Si mostrerà che una delle due varianti soddisfa $S_i$. Per assurdo, si assume $\mathfrak{v}^+, \mathfrak{v}^-$ non verifichino $S_i$.

Allora esistono $C_1, C_2 \in S_i$ tali che $\mathfrak{v}^+(C_1) = 0$ e $\mathfrak{v}^-(C_2) = 0$. Si assume che $\neg p \in C_1$ (senza perdita di generalità), poiché altrimenti se $p \in C_1$ allora $\mathfrak{v}^+(C_1) = 1$, assurdo, e se $p \notin C_1 e \neg p_1 \notin C_1$, allora $C_1$ sarebbe $p$-esonerata, dunque $C_1 \in S_{i+1}$ e $\mathfrak{v} \models C_1$ e $\mathfrak{v}^+ \models C_1$, assurdo. Analogamente per $p \in C_2$.

Si conclude che $D = R(C_1, C_2; \neg p, p)$ e $D \in p$-risolventi implica $D \in S_{i+1}$, che implica $\mathfrak{v} \models D$ e allora

$$\mathfrak{v} \models D \rightarrow \mathfrak{v}^+ \models D \text{ oppure } \mathfrak{v}^- \models D \rightarrow \mathfrak{v}^+ \models C_1 \text{ oppure } \mathfrak{v}^- \models C_2$$

che è assurdo, dunque $S_i$ è soddisfacibile, e pertanto $S_0$ è soddisfacibile. □

**Osservazione.** *Sia $S$ un insieme finito di clausole tale che $Var(S) \subseteq \{p_1, \cdots, p_n\}$. Allora $DPP(S)$ termina in $k \leq n$ passi. Se $k \leq n \leq ||S||$, si potrebbe concludere che si ha una procedura che funziona in tempo polinomiale e decide se $S$ è soddisfacibile o meno, e quindi $P = NP$.*

Se si potesse garantire che $||S_i||$ sia sempre polinomiale rispetto a $||S||$ allora davvero $k \leq n$ passi di DPP porterebbero a un tempo di esecuzione complessivo che è polinomiale rispetto a $||S||$ e dunque $SAT \in P$, e dunque $P = NP$, ma purtroppo questo non si può garantire.

Un risultato, dovuto ad Haken, afferma che ogni prova per refutazione del *Principio della Piccionaia* su $n$ "piccioni" richiede tempo almeno $T(n) = \Omega(2^n)$.

**Definizione** (Principio della Piccionaia)**.** *Il Principio della Piccionaia, notazionalmente espresso su un numero naturale n come*

$$PHP(n)$$

*è esresso affermando che $n+1$ piccioni non possono occupare n posti nella piccionaia in modo che ogni posto abbia al più un piccione.*

Come formalizzare in clausole il $PHP(n)$? Definendo $P$ i "piccioni" e $H$ i "posti", si può definire

$$\bigwedge_{i \in P} \bigvee_{j \in H} p_{ij} \text{ (ogni piccione ha trovato posto}$$

In altre parole, per ogni piccione, il piccione ha trovato almeno un posto, ossia il piccione $i$ sta nel posto $j$. Questo, però va unito al fatto che ogni piccione ha al più un posto:

$$\bigwedge_{i\neq j}\ \bigwedge_{i,j\in P}\bigwedge_{k\in H}(\neq p_{ik}\vee\neq p_{jk})\ \text{nessuna coppia di piccioni sta nel posto } k$$

in altre parole non c'è nessun posto che contenga contemporaneamente due piccioni. Quindi, la formalizzazione finale sarà

$$PHP(n) := \bigwedge_{i\in P}\bigvee_{j\in H} p_{ij}\ \ \wedge\ \ \bigwedge_{i\neq j}\ \bigwedge_{i,j\in P}\bigwedge_{k\in H}(\neq p_{ik}\vee\neq p_{jk})$$

Mostrare che il Principio della Piccionaia è vero consiste nel mostrare che $PHP(n)$ è insoddisfacibile, ossia che $PHP(n)\in CNFUNSAT\forall n\in\mathbb{N}$. Haken ha dimostrato che la risoluzione di questo problema impiega, per ogni $n$, tempo esponenziale.

### Complessità di DPP

Dato che i passi di DPP sono "pochi", $k\leq|Var(S)|$, anche se nel caso peggiore DPP richiede tempo esponenziale, nella pratica è un algoritmo molto più efficiente di $R^*$ e delle tavole di verità. Alcuni frammenti, ossia sottolinguaggi di CNFSAT, sono noti avere tempo di decisione, attraverso la DPP, polinomiale, come KROMSAT e HORNSAT; una CNF è HORN se e solo se ogni sua clausola è una clausola di Horn, ossia una clausola che ha la forma della clausola vuota o è un'unità $\{p\}$ o $\{p,\neg p_1,\neg p_2,\cdots\}$ o più genericamente contiene al più un letterale positivo; una clausola è di Krom se ha al più due letterali (quindi $KROMSAT=2CNFSAT$).

### Davis Putnam Logemann Loveland Procedure

La DPLL è un metodo di "reingegnerizzazione" della DPLL. Su un insieme $S$ di clausole finito, DPLL cerca di costruire un assegnamento che soddisfi $S$. L'idea è di costruire un assegnamento parziale che viene esteso ad ogni passo ad una nuova lettera proposizionale $p$, chiamata pivot. Se si giunge ad assegnare tutte le lettere in $Var(S)$ si è ottenuto un assegnamento che mostra $S$ soddisfacibile: se non si giunge a tal punto, il teorema di completezza e correttezza di DPLL dimostra che $S$ è insoddisfacibile. Si propaga, ad ogni passo, tutta l'informazione che si guadagna estendendo l'assegnamento al pivot. Sostanzialmente, quello che era il passo finale della DPP si utilizza come azione fondamentale nella DPLL.

Può accadere che non vi siano informazioni sfruttabili per quanto riguarda la scelta del pivot ad un certo passo codificata nelle regole che verranno date, allora si "spezza" la procedura in due parti: in una si assegna al pivot il valore di verità 1 e alla seconda si assegna al pivot il valore di verità 0 e la prova risulta dunque in una struttura ad albero i cui rami si visitano in "backtracking", la cui implementazione (anche nel caso di backtracking non cronologico) è soggetto di ampie ricerche; se in un ramo si raggiunge $\emptyset$ si è costruito un assegnamento che soddisfa l'insieme iniziale e se su *tutti* i rami si raggiunge la clausola vuota, allora $S$ è insoddisfacibile.

Le regole utilizzate dalla DPPL sono quelle della DPP adattate a questo contesto; il punto iniziale è l'assegnamento vuoto.

**Definizione** (Assegnamento Parziale)**.** *Un assegnamento parziale è una funzione parziale*

$$\mathfrak{v}: L\to\{0,1,?\}$$

*Un assegnamento vuoto è un assegnamento parziale tale che* $\mathfrak{v}(p_i)=?\forall p_i\in\{p_1,\cdots,p_n\}$*. Un assegnamento completo o totale sulle prime n lettere proposizionali è una mappa* $\mathfrak{v}:\{p_1,\cdots,p_n\}\to\{0,1,?\}$ *tale che* $\mathfrak{v}(p_i)\neq?\forall p_i\in\{p_1,\cdots,p_n\}$.

**Regole di DPLL**

- Regola iniziale: $\emptyset \vdash S$ è la radice della prova.
- Sussunzione: se $\mathfrak{v}(p_i) = 1$ allora si possono cancellare da $S$ tutte le clausole che contengono il letterale $p_i$

$$\frac{\mathfrak{v}, \mathfrak{v}(p_i) = 1 \vdash S \cup \{\{p_i\} \cup C\}}{\mathfrak{v}, \mathfrak{v}(p_i) = 1 \vdash S}$$

  analogamente, se $\mathfrak{v}(p_i) = 0$, allora si possono cancellare da $S$ tutte le clausole che contengono il letterale $\neg p_i$.
- Risoluzione unitaria: se $\mathfrak{v}(p_i) = 0$ si cancella da ogni clausola di $S$ il letterale $p_i$

$$\frac{\mathfrak{v}(p_i) = 0 \qquad (\{\neg p_i, \{p_i, \cdots\})}{C}$$

  Ossia, in termini di "nodi" DPLL

$$\frac{\mathfrak{v}, \mathfrak{v}(p_i) = 0 \vdash S \cup \{\{p_i\} \cup C\}}{\mathfrak{v}, \mathfrak{v}(p_i) = 0 \vdash S \cup \{C\}}$$

  mentre se $\mathfrak{v}(p_i) = 1$, allora si elimina da ogni clausola di $S$ il letterale $\neg p_i$.
- Asserzione: se $S$ contiene la clausola $\{p_i\}$, allora si estende $\mathfrak{v}$ ponendo $\mathfrak{v}(p_i) = 1$, cancellando $\{p_i\}$ da $S$

$$\frac{\mathfrak{v} \vdash S \cup \{\{p_i\}\}}{\mathfrak{v}, \mathfrak{v}(p_i) = 1 \vdash S}$$

  mentre si fa al contrario se $S$ contiene $\{\neg p_i\}$.
- Letterale puro: Se il letterale $p_i$ occorre in $S$ e $\neg p_i$ non occorre, allora si estende $\mathfrak{v}$ ponendo $\mathfrak{v}(p_i) = 1$,

$$\frac{\mathfrak{v} \models S}{\mathfrak{v}, \mathfrak{v}(p_i) = 1 \vdash S}$$

  (chiaramente se $p_i \in C \in S, \neg p_i \notin C \forall C \in S$), mentre si fa contrario se $\neg p_i$ occorre in $S$ e $p_i$ non occorre in $S$, ponendo $\mathfrak{v}(p_i) = 0$.
- Spezzamento: in ogni momento, si può biforcare la prova in due sottoprove, dando origine ad una struttura ad albero, in cui in una si pone per il pivot scelto $\mathfrak{v}(p) = 1$ e nell'altra si pone $\mathfrak{v}(p) = 0$.

$$\frac{\mathfrak{v} \vdash S}{\mathfrak{v}, \mathfrak{v}(p_i) = 0 \vdash S \quad || \quad \mathfrak{v}, \mathfrak{v}(p_i) = 1 \vdash S}$$

- Terminazione: se in un ramo si ottiene $\mathfrak{v} \vdash \emptyset$ allora si prova che $\mathfrak{v}$ è completo su $Var(S)$ e $\mathfrak{v} \models S$. Al contrario, se su tutti i rami si ottiene $\mathfrak{v} \vdash \square$, allora $S$ è insoddisfacibile.
- Ogni ramo o ha come foglia $\mathfrak{v} \vdash \emptyset$ oppure $\mathfrak{v} \vdash \square$.

# Parte II

# Logica dei Predicati

CAPITOLO 6

# Introduzione e Sintassi

Si può immaginare la Logica del Primo Ordine come "costruita" sulla base della logica proposizionale. Il sillogismo aristotelico

$$\frac{\text{Ogni uomo è mortale} \qquad \text{Socrate è un uomo}}{\text{Quindi Socrate è mortale}}$$

espresso come possibile nella Logica proposizionale diventa

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

Ci si chiede se sia vero, quindi: è forse

$$p, q \models r$$

Con DPLL ci si chiede se $\{p, q, \neg r\}$ sia insoddisfacibile. Il primo passo di DPLL afferma

$$\dfrac{\dfrac{\dfrac{\emptyset \vdash \{\{p\},\{q\},\{\neg r\}\}}{\mathfrak{v}(p) = 1 \vdash \{\{q\},\{\neg r\}\}}}{\mathfrak{v}(p) = 1, \mathfrak{v}(q) = 1 \vdash \{\{\neg r\}\}}}{\mathfrak{v}(r) = 0, \mathfrak{v}(p) = 1, \mathfrak{v}(q) = 1 \vdash \emptyset}$$

e pertanto

$$p, q \models r \text{ è falso}$$

che va ovviamente contro la nostra intuizione. Per gestire argomentazioni razionali come il sillogismo aristotelico è necessario dotarsi di un linguaggio più *espressivo* rispetto alla Logica Proposizionale, arricchendone la Sintassi con nuovi operatori, variabili, costanti e la Semantica assegnando un modo di interpretare i nuovi "ingredienti" del linguaggio in modo da poter rappresentare situazioni più raffinate che nella Logica Proposizionale.

Si arriverà a scrivere

$$\forall x(U(x) \to M(x)), U(s) \models M(s)$$

per indicare il sillogismo aristotelico. Oltre ad andare a vedere come mettere in piedi, di primo acchito a livello sintattico, tutta questa struttura, si studierà anche il modo per *risolvere* delle asserzioni, riutilizzando le tecnologie introdotte per la logica proposizionale. In particolare, per fare un esempio introduttivo, si tornerà a chiedersi se il sillogismo aristotelico sia valido ponendosi il quesito

$$\{\{\neg U(x), M(x)\}, \{U(s)\}, \{\neg M(s)\}\} \text{ è soddisfacibile?}$$

Si consideri nuovamente

Ogni uomo è mortale

I modi di designare direttamente dell'*Universo* sono un ingrediente importante della Logica dei Predicati, fornendo per esempio il modo di affermare che Socrate sia mortale, riferendosi ad un preciso elemento. Si necessita un modo per designare un elemento non preciso, in maniera indiretta:

Il padre di ogni uomo è un uomo

Si può concludere, quindi, che il padre di Socrate sia un mortale, nonostante sia una designazione indiretta di un individuo dell'Universo; si può inoltre tradurre quanto detto come

$$\frac{\forall x(U(x) \to M(x) \qquad \forall(U(x) \to U(p(x)) \qquad U(s)}{M(p(s))}$$

Per decidere questa *deduzione*, ci sarà qualcosa come

$$\{\{\neg U(x), M(x)\}, \{\neg U(x), U(p(x))\}, \{U(s)\}, \{\neg M(p(s))\}\}$$

Quindi, scopriremo che non basterà sostituire al posto di $x$ la "lettera" $s$, ma sarà anche necessario considerare $p(s)$. A questo punto, sarà immediato arrivare alla conclusione che la situazione sia in realtà un po' più complicata: come si considera $p(s)$, si dovrebbe considerare anche $p(p(s))$, $p(p(p(s)))$...

Nel nostro caso, una possibile refutazione è la seguente:

$$\neg U(p(s)), M(p(s))$$

istanziando la $x$ su $p(s)$, in questo modo si può risolvere rispetto a $\neg M(p(s))$:

$$\{\{\neg U(p(s))\}, \{\neg U(x), U(p(x))\}, \{U(s)\}\}$$

e si prosegue istanziando $U(x)$ a $U(s)$, arrivando all'insieme vuoto. Vedremo, quindi, perché e quando si possono utilizzare queste istanzianzioni.

## 6.1 Sintassi della Logica del Primo Ordine

Partiamo, dunque, fissando i dettagli sintattici, cioè il Linguaggio della Logica del Primo Ordine nella sua natura grammaticale.

A livello proposizionale, si fissa $L$ come insieme infinito di lettere proposizionali, e si conclude il tutto, poiché da questo insieme si arriva direttamente a costruire ogni formula ammissibile nella Logica Proposizionale.

Innanzitutto, nella Logica del Primo Ordine vi sono diversi linguaggi detti **linguaggi elementari**; la situazione è simile alla situazione generica dei linguaggi formali, anche se tecnicamente $L$ sarebbe un alfabeto per e $F_L$ sarebbe il linguaggio formale, mentre in logica $L$ stesso è chiamato Linguaggio.

Allo stesso modo, nella logica del Prim'Ordine ci sono dei linguaggi elementari, che specificano gli "ingredienti" per costruire le formule della Logica dei Predicati, le quali sarebbero il *vero* linguaggio nel senso formale. In altri termini, in Logica dei Predicati il termine *linguaggio elementare* indica l'insieme di elementi che andranno a costruire l'insieme delle formule.

**Definizione** (Linguaggio Elementare)**.** *I linguaggi elementari sono definiti come* $L = (\mathscr{P}, F, \alpha, \beta)$ *dove P è un insieme di simboli, detti* ***predicati*** *(o simboli di predicato), che deve essere diverso dall'insieme vuoto* $\mathscr{P} \neq \emptyset$*; F è l'insieme di simboli detti* ***di funzione****, disgiunto da* $\mathscr{P}$*:* $\mathscr{P} \cap F = \emptyset$*;* $\alpha$ *è una funzione* $P \to \mathbb{N}$ *assegna l'arità a ogni* $\mathscr{P} \in P$ *e, infine,* $\beta$ *è l'arità di ogni* $f \in F$*.*

Ogni elemento del linguaggio varia in base al linguaggio stesso, tuttavia vi sono "ingredienti" intrinsecamente comuni a tutti i linguaggi elementari: l'insieme $V$ (o $Var$) infinito di simboli detti **variabili individuali**, chiamate così perché la loro interpretazione sarà quella di un *individuo generico* dell'universo del discorso e l'insieme dei connettivi $\wedge, \vee, \neg, \to, \bot, \top, \iff, \cdots$; differentemente dalla logica proposizionale, i linguaggi elementari contengono anche i **quantificatori** $\forall, \exists$.

Il linguaggio $F_L$ delle formule sul linguaggio elementare $L$ sarà definito a partire dai simboli in $L$ e dai possibili connettivi.

**Osservazione.** *Esiste un predicato speciale nella logica del primo ordine che a livello sintattico è uguale agli altri, ma la sua interpretazione è fissata. Se* $\mathscr{P}$ *contiene il simbolo* $'='$*, la sua arità sarà fissata* $\alpha('=') = 2$ *e* $L = (\mathscr{P}, F, \alpha, \beta)$ *è detto linguaggio con identità.*

**Definizione** (Formule)**.** *La definizione di* ***Formula di*** *L (*$F \in F_L$*) è data per strati e induttivamente: il primo strato è quello dei termini, i quali* ***non sono formule*** *ma si usano per costruire formule.*

**Definizione** ($L$-termini)**.** *Sia* $L = (\mathscr{P}, F, \alpha, \beta)$*.*

*Allora l'insieme* $T_L$ *degli L-termini è definito come segue:*

**Base**

- *Ogni* $x \in Var(L)$ *è un termine.*
- *Per ogni* $c \in F$ *tale che* $\beta(c) = 0$*, c è un termine, detto* ***costante****.*

**Passo Induttivo** *Se* $f \in F$ *è tale che* $\beta(f) = n$ *e* $t_1, t_2, \cdots, t_n \in T_L$ *(sono termini già costruiti), allora* $f(t_1, t_2, \cdots, t_n)$ *è un L-termine.*

*Nient'altro è un L-termine.*

*L'insieme* $F_L$ *delle L-Formule è definito come segue.*

**Base** *Se* $p \in \mathscr{P}$ *e* $\alpha(p) = n$ *e* $t_1, \cdots, t_n \in T_L$ *allora* $P(t_1, \cdots, t_n)$ *è una L-Formula detta* ***formula atomica****. Le formule atomiche sono il corrispettivo*

**Passo Induttivo** *Se* $A, B \in F_L$ *allora anche* $A \wedge B$*,* $A \vee B$*,* $\neg A$*,* $A \to B$ *sono L-Formule. Se* $A \in F_L$ *e* $x \in V$ *allora anche* $\forall xA$ *e* $\exists xA$ *sono L-Formule. Se in A non occorre x,* $\forall xA$ *e* $\exists xA$ *sono comunque formule, come anche* $\forall x(\forall xA)$ *e* $\exists x(\forall xA)$*.*

*Nient'altro è una L-Formula.*

**Esercizio** Dare una nozione di *certificato* per $L$-termini e per $L$-Formule analoga alla $L$-costruzione in logica proposizionale.

### 6.1.1 Terminologia

Introduciamo alcune nozioni terminologiche per potersi riferire alla sintassi della Logica dei Predicati. Si userà liberamente la scrittura semplificata di formule omettendo coppie di parentesi quando questo non causa ambiguità, ossia invece di $(\forall xA)$ si scriverà $\forall xA$.

**Definizione** (Termine Ground)**.** *Un termine è detto **chiuso** o **ground** se è costruito senza utilizzare variabili.*

**Definizione** (Variabile vincolata)**.** *Una occorrenza di una variabile x in una Formula $A \in F_L$ è detta **vincolata** se occorre all'interno di una sottoformula di A (ossia una formula che appare in ogni certificato della L-formula A) del tipo $\forall xB$ o $\exists xB$.*

**Definizione** (Variabile libera)**.** *Una variabile $x \in V$ occorre libera in $A \in F_L$ se e solo se qualche sua occorrenza è libera. Questa definizione permette di dare definizioni anche nelle situazioni come l'osservazione precedente.*

Per esempio, nella formula

$$A = (\forall xR(x,y)) \vee P(x)$$

$x$ è sia vincolata che libera, mentre $y$ è libera. Nella formula

$$A' = \forall x(\forall yR(x,y)) \vee P(x)$$

Sia $x$ che $y$ occorrono sempre vincolate.

Questo ci permette di introdurre il concetto fondamentale sul quale lavoreremo: infatti, non ragioneremo più su *formule* quando definiremo la semantica della Logica dei Predicati, ma si ragionerà su un particoalre tipo di formula:

**Definizione** ($L$-sentence o Enunciato)**.** *Un L-**enunciato** o L-**sentence**, detto anche formula chiusa, è una L-formula senza occorrenze di variabili libere (o senza variabili libere).*

**Definizione** (Sostituzione)**.** *Con la notazione $A[t/x]$ con A una formula, x variabile e t un termine, si intende la formula ottenuta rimpiazzando simultaneamente tutte e sole le occorrenze **libere** di x con t.*

CAPITOLO 7

# Semantica della Logica del Primo Ordine

## 7.1 Semantica di Tarski

La semantica esprime *come e quando* un $L$-enunciato è vero (o falso) in una data circostanza. È necessario fissare e formalizzare la nozione di *circostanza* o "mondo possibile". Nel livello Proposizionale, la circostanza è un *assegnamento*. Una volta fatto ciò, sarà necessario formalizzare l'*interpretazione* degli enunciati (e quindi dei termini) in ogni circostanza formalizzata.

Un'idea per la formalizzazione del concetto di **circostanza**, dovuta al logico polacco Alfred Tarski, è che la *verità* è una corrispondenza con lo stato di *Fatto*.

Per la **semantica tarskiana**, nel linguaggio naturale l'enunciato "La neve è bianca" è un enunciato vero se e solo se la neve è bianca. Pertanto, ci si può immaginare di matematizzare questo concetto tramite la teoria degli insiemi, affermando che esista un certo insieme chiamato *Universo del discorso*, composto da vari individui, tra i quali vi è un certo elemento che rappresenta la neve e, rispecchiando la nostra esperienza, si vuole che effettivamente sia vero che la neve sia bianca. Pertanto, "si forza" l'appartenenza della neve all'insieme degli oggetti del discorso che sono bianchi.

In termini matematici, e per il concetto stesso della semantica tarskiana, è necessario considerare Universi in cui vi è la possibilità che la neve non sia bianca, ossia l'enunciato è falso.

A priori, le opzioni *esistono* tutte (la neve è bianca, la neve non è bianca), ma si può utilizzare la formalizzazione degli enunciati anche per modellizzare l'universo, ossia asserendo che il fatto che la neve sia bianca sia vera, e questo sappiamo già farlo con l'uso delle *teorie*.

Per esempio, rielaboriamo il solito sillogismo aristotelico: L'insieme di enunciati

$$\Gamma := \{\{\forall x(U(x) \to M(x))\}, \{U(s)\}\}$$

benché non sia in FNC, è un *teorema*, mentre

$$A := M(s)$$

è la deduzione. Per ottenere un'infrastruttura sulla quale arrivare a compiere il calcolo deduttivo

$$\Gamma, \neg A \text{ è soddisfacibile?}$$

è necessario "concentrarsi" sugli Universi in cui $\Gamma$ è vero. Si possono considerare diversi **Universi del discorso**, composti da vari *individui*: un enunciato caratterizza alcuni universi in cui un enunciato è vero a scapito di altri.

Si consideri, ora, un universo composto da altri universi: chiamando ciò che abbiamo considerato fino ad ora un *universo* una $L$-Struttura, si costruisce l'insieme di tutte le $L$-Strutture; data

una teoria $\Gamma$ si potrà identificare, nell'insieme di tutte le $L$-Strutture, tutte quelle che modellano $\Gamma$.

Definiamo formalmente le $L$-Strutture:

**Definizione** ($L$-Struttura)**.** *Sia dato un linguaggio elementare $L = (\mathscr{P}, F, \alpha, \beta)$. Una L-struttura è una coppia $\mathscr{A} = (A, I)$, dove A è un insieme detto "universo del discorso" o, semplicemente, universo (o dominio), che deve rispettare*

$$A \neq \emptyset$$

*mentre I è una funzione detta interpretazione, definita tale che per ogni simbolo di predicato $P \in \mathscr{P}$ con arità $\alpha(P) = n$ si ha*

$$I(P) \subseteq A^n$$

*ossia $I(P)$ è un insieme di n-ple di elementi di A; inoltre, per ogni $f \in F$, $\beta(f) = n$ si ha*

$$I(f) : A^n \to A$$

*ossia $I(F)$ è una funzione dall'insieme delle n-ple di A verso elementi di A.*

La definizione di $L$-struttura data formulizza a livello del Primo ordine la nozione intuitiva di "circostanza" o "mondo possibile". A livello proposizionale, essa è formalizzata dalla nozione di assegnamento.

**Osservazione** (Costanti)**.** *Se $c \in F$ e $\beta(c) = 0$*

$$I(c) : A^0 \to A$$

*e $A^0 = \{f : \emptyset \to A\}$, la cui cardinalità è*

$$|A^0| = |\{f : \emptyset \to A\}| = |\{\emptyset\}| = 1$$

*Pertanto*

$$I(c) : \{*\} \to A$$

*(la notazione $\{*\}$ rappresenta l'insieme che contiene l'insieme vuoto), che equivale a "scegliere" un elemento di A. Identifichiamo una funzione*

$$g : \{*\} \to A$$

*con l'elemento scelto da $g(*)$. In altre parole, le costanti sono funzioni zerarie.*

**Osservazione** (Predicati zerari)**.** *Se $p \in \mathscr{P}, \alpha(P) = 0$ è un simbolo di predicato zerario, si ha*

$$I(P) \subseteq A^0 \quad I(P) \subseteq \{*\}$$

*I cui sottoinsiemi sono sé stesso $\{*\}$ e l'insieme vuoto $\emptyset$. E quindi l'interpretazione di un predicato zerario svolge le funzioni di una vecchia lettera proposizionale, in quanto l'interpretazione di una proposizione può essere identificata tramite la sua funzione caratteristica*

$$\chi_P : A^n \to \{0, 1\}$$

*definita come*

$$\bar{a} \in A^n \in I(P) \iff \chi_P(\bar{a}) = 1$$

**Osservazione** (Interpretazione fissa del predicato di Uguaglianza)**.** *Se* $=\in \mathscr{P}$ *allora L è un linguaggio con identità e* $\alpha(=) = 2$*. L'interpretazione di "=" è fissata: se* $\mathscr{A} = (A, I)$ *è una L-struttura, allora*

$$I(=) \subseteq A^2$$

*è fissato e definito come*

$$I(=) := \{(a,a) : a \in A\}$$

*e, in altre parole,* $x = y$ *è vero se* $(I(x), I(y)) \in I(=)$.

L'eccezione sull'interpretazione del predicato di uguaglianza, fissato per ogni linguaggio elementare che lo contiene, è giustificato dal fatto che con la potenza espressiva della Logica dei Predicati, si riesce a dire che una relazione binaria gode della proprietà riflessiva, simmetrica e transitiva.

Ci si può anche spingere a dire che per ogni altro simbolo, sia di predicato che di funzione, quella relazione si comporta in maniera *congruenziale*. Anche questo non è sufficiente per inchiodare l'identità, in quanto una congruenza su un insieme che non è l'identità rispetta gli stessi assiomi.

In altre parole, o l'identià si codifica a forza, come accade ora nella Logica del Prim'Ordine, oppure se ne ha una versione molto indebolita, in quanto non si ha modo di distinguere con formule della Logica dei Predicati l'identità da una congruenza sullo stesso insieme.

### 7.1.1 Semantica degli enunciati in ogni *L*-struttura

Abbiamo dato la nozione di *L*-Struttura ma non si è ancora detto come si interpretano gli enunciati, formalmente, cioè quando un enunciato è vero o meno in tale *L*-Struttura. Intuitivamente, quello che si vuole fare è quanto anticipato, ossia scegliere le *L*-Strutture che verificano una certa teoria. Tuttavia, questo deve essere eseguito seguendo una definizione rigorosa e sistematicamente.

Si procederà per strati, ossia definendo prima come interpretare i termini e successivamente formule ed enunciati, induttivamente. Si deve catturare l'idea intuitiva: "la neve è bianca" verrà formalizzato in qualcosa come $B(n)$, dove $B$ è un predicato unario e $n \in A$ è una costante (quindi esiste una funzione zeraria il cui risultato è $n$). Si vorrà formalizzare il tutto in modo tale da avere una qualche $L$ struttura tale che $I(n)$ sia contenuta in $I(B)$; quindi si avrà che $B(n)$ è vera in $\mathscr{A}$ se e solo se $I(n) \in I(B)$:

$$\mathscr{A} \in B(n) \iff I(n) \in I(B)$$

Il passo più ostico nella definizione di interpretazione di termini e formule (ed enunciati) sarà nell'interpretazione dei quantificatori; per esempio, si supponga che si vuole stabilire se

$$\mathscr{A} \models \exists x A$$

che significa: se esiste un $a \in A$ tale che $\mathscr{A} \models A[a/x]$ e analogamente per l'altro quantificatore

$$\mathscr{A} \models \forall x A$$

che è vero se e solo se per ogni $a \in A$ vale $\mathscr{A} \models A[a/x]$.

Purtroppo $a$ non è un elemento sintattico, ma è un elemento semantico, ossia non è un termine e pertanto $A[a/x]$ non è definita. Non c'è nessuna ragione per cui ogni elemento di una *L*-struttura debba avere, a priori, un nome: piuttosto, è vero il contrario. Se si vuole parlare ad esempio dei reali, si vorrebbe farlo con un linguaggio elementare $L$ che contiene solo finitamente molti simboli.

**Definizione** (Espansione di un $L$-Struttura)**.** *Dato un linguaggio $L$ e una $L$-struttura $\mathscr{A}(A,I)$, si definisce una* ***espansione*** *di $L$ in un nuovo linguaggio $L_{\mathscr{A}}$ come segue: per ogni $a \in A$ arricchisci l'insieme delle costanti di $L$ con un nuovo simbolo $\bar{a}$ (nome di $a$) che viene interpretato in $a$.*

$$L_{\mathscr{A}} := (\mathscr{P}, F^{\mathscr{A}}, \alpha, \beta^{\mathscr{A}})$$

*con*

$$F^{\mathscr{A}} := F \cup \{\bar{a} : a \in A\}$$

*e*

$$\begin{cases} \beta^{\mathscr{A}} = B & f \in F \\ \beta^{\mathscr{A}}(\bar{a}) = 0 & a \in A \end{cases}$$

*E inoltre $I(\bar{a}) := a$ per ogni $a \in A$.*

**Definizione** (Interpretazione degli $L_{\mathscr{A}}$-termini ground)**.** *Dati $L = (\mathscr{P}, F, \alpha, \beta)$ e $\mathscr{A} = (A, I)$ e la $L_{\mathscr{A}}$-struttura espansa, si definisce induttivamente:*

- ***Base*** *ogni $c \in F$ tale che $\beta(c) = 0$ ha un'interpretazione $I(c)$ già definita, dalla definizione di $L$-struttura. Invece, $\bar{a} \in F^A = F \cup \{\bar{a} : a \in A\}$ ha come interpretazione $I(\bar{a}) = a \in A$.*
- ***Passo Induttivo*** *Sia $f \in F$ un simbolo di funzione con arità $\beta(f) = n$ e siano $t_1, t_2, \cdots, t_n$ $L_{\mathscr{A}}$-termini ground. Allora $I(F(t_1, \cdots, t_n)) := I(f)(I(t_1), \cdots, I(t_n))$.*

**Definizione** (Interpretazione degli Enunciati)**.** *Sia data $\mathscr{A} = (A, I)$ una $L$-struttura; allora, si definisce induttivamente la nozione $\mathscr{A} \models A$ per ogni $L_{\mathscr{A}}$-enunciato $A$, ossia $A$ è vera in $\mathscr{A}$.*

**Base** *Sia $A = P(t_1, \cdots, t_n)$ per $P \in \mathscr{P}$ con $\alpha(P) = n$ e $t_1, \cdots, t_n$ $L_{\mathscr{A}}$-termini ground. Allora*

$$\mathscr{A} \models P(t_1, \cdots, t_n) \iff I(t_1), \cdots, I(t_n) \in I(P) \subseteq A^n$$

**Passo induttivo**

- $\mathscr{A} \models A_1 \wedge A_2 \iff \mathscr{A} \models A_1 \wedge \mathscr{A} \models A_2$
- $\mathscr{A} \models A_1 \vee A_2 \iff \mathscr{A} \models A_1 \vee \mathscr{A} \models A_2$
- $\mathscr{A} \models \neg A_1 \iff \mathscr{A} \not\models A_1$
- $\mathscr{A} \models A_1 \to A_2 \iff \mathscr{A} \models A_1 \wedge \mathscr{A} \models A_2$
- $\mathscr{A} \models \forall x A \iff \mathscr{A} \models A[\bar{a}/x]$ *per ogni $a$*
- $\mathscr{A} \models \exists x A \iff \mathscr{A} \models A[\bar{a}/x]$ *per almeno un $a$*

*Si noti che $\bar{a}$ è un termine, mentre $x$ è una variabile individuale, quindi $A$ diventa un $L_{\mathscr{A}}$-enunciato.*

Abbiamo quindi definito induttivamente $\mathscr{A} \models A$ per gli $L_{\mathscr{A}}$-enunciati $A$. A questo punto ci si può "dimenticare" degli $L_{\mathscr{A}}$-enunciati che non sono $L$-enunciati, ossia quelle formule in cui vi è almeno una variabile libera.

**Definizione** (Chiusura universale di una formula con variabili libere)**.** *Se $A$ è una formula ben formata, ossia $A \in F_L$ ed eventualmente, non un enunciato, quindi $FV(A) \neq \emptyset$, dove $FV$ è definito come l'insieme di variabili libere che appaiono in $A$, ci si può chiedere se*

$$\mathscr{A} \models A$$

*Definito, quindi,*

$$FV(A) = \{x_1, \cdots, x_n\}$$

*si definisce la chiusura universale di $A$*

$$\forall[A] := \forall x_1 \forall x_2 \cdots \forall x_n A$$

*e si postula che*

$$\mathscr{A} \models A \iff \mathscr{A} \models \forall[A]$$

ossia $\mathscr{A}$ modella una formula ben formata $A$ se e solo se $\mathscr{A}$ rende vera la sua chiusura universale, che è un $L$-enunciato.

**Osservazione.** *Se $L$ è un linguaggio con identità (o con uguaglianza), ossia $L = (\mathscr{P}, F, \alpha, \beta)$ e $= \in \mathscr{P}$, $\alpha(=) = 2$, allora, ricordando che in ogni $L$-struttura $\mathscr{A} = (A, I)$, si deve avere per definizione $I(=) = \{(a,a) : a \in A\}$. Pertanto $\mathscr{A} \models t_1 = t_2$ (che, più correttamente, andrebbe scritto in forma postfissa "$= (t_1, t_2)$" se e solo se $I(t_1) = I(t_2)$.*

## 7.1.2 Definizione alternativa di $\mathscr{A} \models A$

In questa definizione alternativa, le variabili sono oggetti di interesse; tuttavia le due definizioni, quella su $L \to L_{\mathscr{A}}$ e quella che andiamo a dare, sono equivalenti.

**Definizione** (Ambiente)**.** *Un **ambiente di interpretazione** in una $L$-struttura $\mathscr{A} = (A, I)$ è una funzione*

$$\mathfrak{v} : Var \to A$$

*ossia $\mathfrak{v}(x) \in A$ è un elemento di $A$.*

**Definizione** (Variante di ambiente)**.** *La **variante di ambiente** servirà a dare la semantica alle espressioni quantificate ed è indicato $\mathfrak{v}[a/x](y)$ e si definisce*

$$\mathfrak{v}[a/x](y) := \begin{cases} \mathfrak{v}(y) & y \neq x \\ a & y = x \end{cases}$$

**Definizione** (Interpretazione degli $L$-termini)**.** *L'interpretazione degli $L$-termini, anche aperti, per ogni $x \in Var$ e $\mathfrak{v} : Var \to A$ è definita come*

$$I_{\mathfrak{v}}(x) := \mathfrak{v}(x)$$

*e si può ora passare a definire*

$$I_{\mathfrak{v}}(F(t_1, \cdots, t_n)) := (I(F))(I_{\mathfrak{v}}(t_1), \cdots, I_{\mathfrak{v}}(t_n))$$

*Si dirà*

$$\mathscr{A} \models_{\mathfrak{v}} A$$

*se $\mathscr{A}$ rende vera $A$ nell'ambiente $\mathfrak{v} : Var \to A$, e si dirà*

$$\mathscr{A} \models_{\mathfrak{v}} P(t_1, \cdots, t_2) \iff (I_{\mathfrak{v}}(t_1), \cdots, I_{\mathfrak{v}}(t_n)) \in I(P)$$

*e quindi*

- $\mathscr{A} \models_{\mathfrak{v}} A_1 \wedge A_2 \iff \mathscr{A} \models A_1 \wedge \mathscr{A} \models A_2$
- $\mathscr{A} \models_{\mathfrak{v}} A_1 \vee A_2 \iff \mathscr{A} \models A_1 \vee \mathscr{A} \models A_2$
- $\mathscr{A} \models_{\mathfrak{v}} \neg A_1 \iff \mathscr{A} \not\models A_1$
- $\mathscr{A} \models_{\mathfrak{v}} A_1 \to A_2 \iff \mathscr{A} \models A_1 \wedge \mathscr{A} \models A_2$
- $\mathscr{A} \models_{\mathfrak{v}} \forall x A \iff \mathscr{A} \models_{\mathfrak{v}} A[a/x]$ *per ogni* $a$
- $\mathscr{A} \models_{\mathfrak{v}} \exists x A \iff \mathscr{A} \models_{\mathfrak{v}} A[a/x]$ *per almeno una* $a$

*A questo punto si può dire che* $\mathscr{A} \models A$ *se e solo se per ogni* $\mathfrak{v} : Var \to A$ *si ha* $\mathscr{A} \models A$.

Si esprime, ora, un'idea intuitiva di come avvenga il processo di interpretazione di un $L$-enunciato. Nella prima definzione si ha il linguaggio $L$ e delle $L$-strutture che vengono associate.

$$L \rightsquigarrow \begin{cases} \mathscr{A} \rightsquigarrow L_{\mathscr{A}} \\ \mathscr{B} \rightsquigarrow L_{\mathscr{B}} \\ \mathscr{C} \rightsquigarrow L_{\mathscr{C}} \\ \dots \end{cases}$$

Per ogni $L$-struttura si crea l'espansione associata. Nella seconda definizione si ha nuovamente il linguaggio $L$ e delle $L$-strutture, tuttavia esse non si espandono ma vi sono un numero di ambienti da considerare:

$$L \rightsquigarrow \begin{cases} \mathscr{A} \begin{cases} \mathscr{A}, \mathfrak{v}_1 \\ \mathscr{A}, \mathfrak{v}_2 \\ \mathscr{A}, \mathfrak{v}_3 \\ \dots \end{cases} \\ \mathscr{B} \begin{cases} \mathscr{B}, \mathfrak{v}_1 \\ \mathscr{B}, \mathfrak{v}_2 \\ \mathscr{B}, \mathfrak{v}_3 \\ \dots \end{cases} \\ \dots \end{cases}$$

## 7.2 Terminologia e Nozioni

**Definizione** ($L$-Teoria)**.** *Una L-Teoria è un insieme di L-enunciati. Se* $\Gamma$ *è una L-Teoria e* $\mathscr{A}$ *è una L-struttura, si dice che* $\mathscr{A}$ *è modello di* $\Gamma$ *e si scrive* $\mathscr{A} \models \Gamma$ *se e solo se* $\mathscr{A} \models \gamma$ *per ogni* $\gamma \in \Gamma$.

*La L-teoria* $\Gamma$ *è soddisfacibile e solo se esiste almeno un* $\mathscr{A}$ *tale che* $\mathscr{A} \models \Gamma$*, altrimenti* $\Gamma$ *è insoddisfacibile.*

*Sia* $\Gamma$ *una L-teoria e A un L-enunciato. Allora A è conseguenza logica di* $\Gamma$ *e si scrive* $\Gamma \models A$ *se e solo se A è vera in ogni modello di* $\Gamma$*, ossia per ogni L-struttura* $\mathscr{A}$*, se* $\mathscr{A} \models \Gamma$*, allora* $\mathscr{A} \models A$.

**Definizione** (Verità Logica)**.** *Un L-enunciato A è detto* ***verità logica*** *se e solo se per ogni L-struttura* $\mathscr{A}$*,* $\mathscr{A} \models A$.

Il concetto di Verità Logica è analogo al concetto di tautologia nella Logica Proposizionale. Un esempio di verità logica è $\forall x(x = x)$.
Lo scopo della nostra indagine sarà, d'ora in poi, cercare di stabilire se

$$\Gamma \models A$$

dati $\Gamma$ una $L$-teoria e $A$ un $L$-enunciato. Se $\Gamma$ è finito, ossia $\Gamma = \{\gamma_1, \cdots, \gamma_n\}$, varrà nuovamente il fatto

$$\Gamma \models A \iff \gamma_1 \wedge \cdots \wedge \gamma_n \wedge \{\neg A\} \text{ insodd.}$$

ossia ci si riduce all'analisi di una singola formula, mentre invece se $\Gamma$ è infinito, possiamo solo chiederci se

$$\Gamma \models A \iff \Gamma \cup \{\neg A\} \text{ insodd.}$$

Anche nel caso in cui $\Gamma$ è finito, ci saranno casi in cui il processo di deduzione sarà solo *semi-decidibile*, mentre nella Logica Proposizionale il problema di soddisfacibilità era decidibile.

### 7.2.1 Esempi di $L$-Teorie

**Congruenze su un $L$-Linguaggio $L$**

Data una relazione $R$, si vorrebbe modellarla come congruenza.

$$\Gamma := \begin{cases} \forall x \ R(x,x) \text{ riflessiva} \\ \forall x \forall y \ R(x,y) \to R(y,x) \text{ simmetrica} \\ \forall x \forall y \forall z \ R(x,y) \wedge R(y,z) \to R(x,z) \text{ transitiva} \\ \forall x_1 \cdots \forall x_n \forall y_1 \cdots \forall y_n \begin{cases} (R(x_1,y_1) \wedge \cdots \wedge R(x_n,y_n)) \to R(f(x_1,\cdots,x_n), f(y_1,\cdots y_n)) \\ (R(x_1,y_1) \wedge \cdots R(x_n,y_n)) \to (P(x_1,\cdots x_n) \to P(y_1,\cdots,y_n)) \end{cases} \end{cases}$$

I primi tre $L$-enunciati (assiomi di $\Gamma$) assiomatizzano una relazione d'equivalenza, ossia

$$\text{se } \mathscr{A} \models \Gamma \text{ allora } I(R) \text{ è una relazione d'equivalenza}$$

Il quarto $L$-enunciato assiomatizza la congruenza $R$ rispetto ad una generica $f \in F$ con $\beta(f) = n$, mentre il quinto fa la stessa cosa rispetto un generico predicato $P \in \mathscr{P}$.

**Relazione d'ordine (parziale)**

$$\Gamma_{po} := \begin{cases} \forall x R(x,x) & \text{riflessiva} \\ \forall xy(R(x,y) \wedge R(y,x)) \to = (x,y) & \text{simmetrica} \\ \forall x \forall y \forall z(R(x,y) \wedge R(y,z)) \to R(x,z) & \text{transitiva} \end{cases}$$

Si noti l'utilizzo del predicato $=$ d'uguaglianza, pertanto il Linguaggio è dotato di tale predicato. Un esempio, l'insieme dei naturali con la relazione di minore o uguale modellano $\Gamma$:

$$(\mathbb{N}, \leq) \models \Gamma$$

(la notazione utilizzata significa che $I(R) = \leq$) così come i naturali con l'ordine di divisibilità:

$$(\mathbb{N}, |) \models \Gamma$$

**Relazione d'ordine (totale)** Esiste un modo per costruire una $L$-Teoria che "allarga" $\Gamma$ che è modello di uno e non dell'altra? Effettivamente, il primo è un ordine totale, mentre il secondo non lo è, ossia vi sono elementi tali che né $m|n$ né $n|m$; se si riesce ad esprimere la totalità con un enunciato, si riesce a discernere uno dall'altro:

$$\Gamma_{to} := \Gamma_{po} \cup \{\forall x \forall y (R(x,y) \vee R(y,x))\}$$

Aggiungendo nuovi assiomi si possono caratterizzare altri tipi di ordine: ad esempio (esercizio), si può caratterizzare un ordine che abbia un elemento minimo.

### Teoria dei gruppi

Un gruppo è una struttura algebrica. Un gruppo è un modello dell'insieme di assiomi $\Gamma_G$ sul linguaggio $L = (\mathscr{P}, F, \alpha, \beta)$.

$$\Gamma_G := \begin{cases} \forall x \forall y \forall z \ \ ((x*y)*z) = (x*(y*z)) & \text{associatività} \\ \forall x (x*e) = x \wedge (e*x) = x & \text{elemento neutro} \\ \forall x (x*x^{-1}) = e \wedge (x^{-1}*x = e) & \text{invertibilità} \end{cases}$$

con $\mathscr{P} = \{=\}$, $F = \{*, e, ()^{-1}\}$ e relative arità. $(G, \cdot, ^{-1}, 0) \models \Gamma_G$ se e solo se $G$ è un gruppo; in altri termini un gruppo è una struttura con un'operazione associativa in cui ogni elemento è invertibile. Per esempio, $(\mathbb{Z}, +, -, 0) \models \Gamma_G$ e $(\mathbb{Q} \setminus 0, \cdot, ^{-1}, 1) \models \Gamma_G$.

**Formulazione alternativa** Una seconda formulazione è la seguente

$$\Gamma_{G_2} := \begin{cases} \forall x \forall y \forall z \ \ ((x*y)*z) = (x*(y*z)) & \text{associatività} \\ \forall x (x*e) = x \wedge (e*x) = x & \text{elemento neutro} \\ \forall x \exists y (x*y) = e \wedge (y*x) = e & \text{esistenza inverso} \end{cases}$$

Questa $L$-Teoria rimuove la specifica dell'inverso. Se si ottiene un gruppo che modella l'altra formulazione, esso andrà bene anche per questo, ma non è detto il viceversa. Ora, infatti, non è più necessario specificare l'operazione inverso, e si può quindi affermare $(\mathbb{Z}, +, 0) \models \Gamma_{G_2}$, $(Q, +, 0) \models \Gamma_{G_2}$, eccetera. Come anticipato, questo vale anche per i gruppi esplicitati prima: $(\mathbb{Q} \setminus 0, \cdot, ^{-1}, 1) \models \Gamma_{G_2}$ e così via, anche se l'inverso non viene utilizzato come simbolo esplicito. Potremmo quindi utilizzare, al posto del simbolo dell'inverso, qualsiasi cosa e rimarrà comunque un modello di $\Gamma_{G_2}$:

$$(\mathbb{Z}, +, +1, 0) \models \Gamma_{G_2}$$

Ma ovviamente non è vero il contrario, ossia

$$(\mathbb{Z}, +, +1, 0) \not\models \Gamma_G$$

in questo senso, abbastanza sottile, le due formulazioni non sono equivalenti.

### Tipi di dato: stack

Definiamo il linguaggio $L = (\mathscr{P}, F, \alpha, \beta)$ con

$$\mathscr{P} = \{Stack, Elem, =\}$$

e

$$F = \{push, pop, top, nil\}$$

$$\Gamma_{Stack} := \begin{cases} \forall x(Stack(x) \vee Elem(x)) \\ \forall x(\neg Stack(x) \vee \neg Elem(x)) \\ \forall x \forall y((Stack(x) \wedge Elem(y)) \rightarrow top(push(x,y)) = y) \\ \forall x \forall y((Stack(x) \wedge Elem(y)) \rightarrow pop(push(x,y)) = x) \\ \forall x(Stack(x) \rightarrow push(pop(x), top(x)) = x) \end{cases}$$

**Aritmetica di Peano**

Sia $L_{P_A} = (\mathscr{P}, F, \alpha, \beta)$, con $\mathscr{P} = \{=\}$, $\alpha(=) = 2$, $F = \{+, *, s, 0, \}$ $\beta(+) = \beta(*) = 2$, $\beta(s) = 1$, $\beta(0) = 0$.

L'aritmetica di Peano afferma:

- $\forall x \neg(= (0, s(x)))$, ossia nessun numero ha come successore 0.
- $\forall x, \forall y(s(x) = s(y)) \rightarrow (x = y)$
- $\forall x(x + 0 = x)$
- $\forall x \forall y(x + s(y)) = s(x + y)$
- $\forall x(x * 0 = 0)$
- $\forall x \forall y(x * s(y)) = (x + (x * y))$
- $(P[0/x] \wedge \forall x(P[x/x] \rightarrow P[s(x)/x])) \rightarrow \forall x P[x/x]$

L'ultimo postulato è una codifica dell'induzione sui numeri naturali.

CAPITOLO 8

# Risoluzione Automatica

Lo scopo del nostro corso è analizzare come sia possibile capire se una certa teoria $\Gamma \models A$ in modo automatico. Se $\Gamma = \{\gamma_1, \cdots, \gamma_n\}$, $\Gamma \models A$ se e solo se $\gamma_1 \wedge \gamma_2 \wedge \cdots \wedge \gamma_n \wedge \neg A$ è insoddisfacibile. Quindi, informalmente, il nostro scopo è, dato $A$ un $L$-enunciato, stabilire se $A$ è insoddisfacibile.

## 8.1 Forme Normali

Cercheremo di seguire il più possibile ciò che è stato fatto anche per la Logica Proposizionale, ossia riportare il tutto in CNF per poi applicare qualcosa di simile al calcolo refutazionale - tuttavia, nella Logica dei Predicati, questo passaggio è leggermente più complicato.

Prima di esaminare questa procedura, diamo delle definizioni utili.

**Definizione** (Equivalenza Logica)**.** *Siano A e B L-enunciati. A è logicamente equivalente a B ($A \equiv B$) se e solo se A e B hanno gli stessi modelli, cioè se $\mathscr{A} \models A \iff \mathscr{A} \models B$ per ogni L-struttura $\mathscr{A}$, equivalentemente si afferma*

$$\models (A \to B) \wedge (B \to A)$$

*è una verità logica.*

**Definizione** (Equisoddisfacibilità)**.** *Siano A e B due L-enunciati. A è equisoddisfacibile a B se e solo se A ha un modello se e solo se B ha un modello, anche se sono modelli diversi.*

**Definizione** (Forma Normale Prenessa (PNF))**.** *Un L-enunciato è in PNF se e solo se è nella forma*

$$Q_1 x_1, \ldots, Q_n x_n M$$

*dove $Q_i \in \{\forall, \exists\}$ e $M \in FBF$ è priva di quantificatori. M è detta matrice, mentre la parte restante è chiamata prefisso.*

Ogni $L$-enunciato è logicamente equivalente ad un $L$-enunciato in PNF, ossia

$$A \equiv A^P \quad A^P \in PNF$$

$A^P$ si costruisce efficientemente da $A$ utilizzando la ripetuta riscrittura tramite **equivalenze logiche notevoli**, ossia partendo da un certo $A_0$, si arriva in un numero limitato di passi ad $A_N$ che è in $PNF$:

$$A := A_0 \rightsquigarrow A_1 \rightsquigarrow A_2 \ldots \rightsquigarrow A_n = A^P \in PNF$$

### 8.1.1 Equivalenze Notevoli

Sia $P$ una FBF e sia $z$ una variabile che **non occorre** (né libera né vincolata) in $P$.

**Rinomina**

$$\begin{cases} \exists xP \equiv \exists zP[z/x] \\ \forall xP \equiv \forall zP[z/x] \end{cases}$$

**De Morgan**

$$\begin{cases} \neg\forall xP \equiv \exists x\neg P \\ \neg\exists xP \equiv \forall x\neg P \\ \forall xP \equiv \neg\exists x\neg P \\ \exists xP \equiv \neg\forall x\neg P \end{cases}$$

*Dimostrazione.* di (1)

$$\begin{aligned} \mathscr{A} \models \neg\forall xP \iff \mathscr{A} \not\models \forall xP \iff \neg\forall a \in A\mathscr{A} \models P[\bar{a}/x] \\ \text{per almeno un } a \in \mathscr{A}\ \mathscr{A} \not\models P[\bar{a}/x] \\ \text{esiste almeno un } a \in \mathscr{A} \models \neg P[\bar{a}/x] \\ \exists x\neg P \end{aligned}$$

□

**Quantificazione Ridondante**

$$\begin{cases} \forall x\forall yP \equiv \forall y\forall xP \\ \exists x\exists yP \equiv \exists y\exists xP \forall xP \equiv P & x \notin FV(P) \\ \exists xP \equiv P & x \notin FV(P) \end{cases}$$

È importante notare che invece non vale, in genere, $\forall x\exists yP \equiv \exists y\forall xP$.

**Quantificatore rispetto a $\wedge$ e $\vee$**

$$\begin{cases} \forall x(P_1 \wedge P_2) \equiv \forall xP_1 \wedge \forall xP_2 \\ \exists x(P_1 \vee P_2) \equiv \exists xP_1 \vee \exists xP_2 \\ \forall x(P_1 \vee P_2) \equiv \forall xP_1 \vee P_2 & x \notin FV(P_2) \\ \exists x(P_1 \wedge P_2) \equiv \exists xP_1 \wedge P_2 & x \notin FV(P_2) \end{cases}$$

Si noti che, per esempio, non valgono

$$\forall x(P_1 \vee P_2) \neq \forall xP_1 \vee \forall xP_2$$

e

$$\exists x(P_1 \wedge P_2) \neq \exists xP_1 \wedge \exists xP_2$$

Per esempio, infatti

$$\forall x(P(x) \vee D(x))$$

dove $P$ è l'insieme dei numeri pari mentre $D$ è l'insieme dei numeri dispari. Mentre è vero che

$$\mathbb{N} \models \forall x(P(x) \vee D(x))$$

ma ovviamente non è vero che

$$\mathbb{N} \not\models \forall x P(x) \vee \forall x D(x)$$

Analogamente, mentre non vale

$$\mathbb{V} \not\models \exists x(P(x) \wedge D(x))$$

vale

$$\mathbb{N} \models \exists x P(x) \wedge \exists x D(x)$$

**Rinomina**

Supponiamo di voler mettere in Forma Normale Prenessa la seguente formula:

$$\exists x P(x) \wedge \exists x D(x)$$

utilizzando le equivalenze logiche che conosciamo.

$$\begin{aligned} \exists x P(x) \wedge \exists x D(x) &\equiv \exists x P_1 \wedge P_2 \text{ se } x \notin FV(P_2) \\ &\equiv \exists x(P(x) \wedge \exists x D(x)) \\ &\equiv \exists x(P(x) \wedge \exists y D(y)) \text{ rinomina} \\ &\equiv \exists x \exists y(P(x) \wedge D(y)) \end{aligned}$$

**Altre Equivalenze Notevoli**

Dati due quantificatori $Q_1, Q_2 \in \{\forall, \exists\}$, si può affermare

$$\begin{cases} Q_1 x P_1 \vee Q_2 x P_2 \equiv Q_1 x Q_2 z(P_1 \vee P_2[z/x]) \\ Q_1 x P_1 \wedge Q_2 x P_2 \equiv Q_1 x Q_2 z(P_1 \wedge P_2[z/x]) \end{cases}$$

con $z \notin FV(P_1) \cup FV(P_2)$, o in generale una nuova variabile. Inoltre, se $x \notin FV(Q)$ (o con rinomina):

$$\begin{cases} \forall x P \to Q \equiv \exists x(P \to Q) \\ Q \to \forall x P \equiv \forall x(Q \to P) \\ Q \to \exists x P \equiv \exists x(Q \to P) \end{cases}$$

Dato un $L$-enunciato $A$, c'è una sequenza

$$A := A_0 \rightsquigarrow A_1 \rightsquigarrow A_2 \ldots \rightsquigarrow A_n = A^P \in PNF$$

e ogni passaggio $A_i \rightsquigarrow A_{i+1}$ è ottenuto applicando una delle equivalenze notevoli elencate, dunque $A \equiv A^P$.

**Osservazione.** *Il numero di equivalenze notevoli da utilizzare per la trasformazione è*

$$n \leq p(||A||)$$

*per un qualche polinomio $p$.*

### 8.1.2 Forma Normale di Skolem

Sia $A$ un $L$-enunciato, e sia $A^P \in PNF$ la sua forma normale prenessa. Si vuole ora eliminare un tipo di quantificatore, ossia quello esistenziale "$\exists$": un enunciato $A \in PNF$ si dice in Forma Normale di Skolem se e solo se non contiene occorrenze del quantificatore $\exists$. La motivazione dietro questa necessità è che vogliamo arrivare nuovamente al calcolo refutazionale, dove si va a dimostrare che una certa formula è insoddisfacibile. Questa operazione porta gli enunciati in una Forma Normale chiamata Forma Normale di Skolem.

**Definizione** (Forma Normale di Skolem)**.** *Un L-enunciato $A \in PNF$ si dice in Forma Normale di Skolem se e solo se non contiene occorrenze del quantificatore esitenziale.*

$$A := Q_1 x_1 Q_2 x_2 \cdots Q_n x_n M$$

$A \in SNF \iff Q_i = \forall$

Vi sono altre tradizioni di calcolo, che invece mirano alla prova diretta (come i calcoli alla Hilbert per la Logica Proposizionale) in cui si vuole eliminare il quantificatore universale.

#### Skolemizzazione

Vi sarà dunque un procedimento, chiamato **skolemizzazione**, che porta un generico $L$-enunciato $A$ in Forma Normale Prenessa ad un enunciato **equisoddisfacibile** in Forma Normale di Skolem.

**Esempio di Skolemizzazione** Riprendiamo le due formulazioni date della teoria dei gruppi:

$$\Gamma_G := \begin{cases} \forall x \forall y \forall z \ ((x*y)*z) = (x*(y*z)) & \text{associatività} \\ \forall x (x*e) = x \wedge (e*x) = x & \text{elemento neutro} \\ \forall x (x*x^{-1}) = e \wedge (x^{-1}*x = e) & \text{invertibilità} \end{cases}$$

$$\Gamma_{G_2} := \begin{cases} \forall x \forall y \forall z \ ((x*y)*z) = (x*(y*z)) & \text{associatività} \\ \forall x (x*e) = x \wedge (e*x) = x & \text{elemento neutro} \\ \forall x \exists y (x*y) = e \wedge (y*x) = e & \text{esistenza inverso} \end{cases}$$

La seconda formulazione non è equivalente alla prima, tuttavia sono perlomeno equisoddisfacibili. Si può notare che nella seconda formulazione vi è un assioma con un quantificatore esistenziale. Si può pensare che la prima formulazione sia la Forma Skolemizzata della seconda formulazione.

Nella seconda formulazione, il terzo assioma è descritto come

$$\forall x \exists y (x*y) = e \wedge (y*x) = e \text{ esistenza inverso}$$

Questo sarà vero in una struttura (che sarà un gruppo) $\mathscr{A} = (A, I)$ se e solo se per ogni $a \in A$ esiste $b \in A$ tale che $\bar{a} * \bar{b} = e$ e $\bar{b} * \bar{a} = e$. L'idea sottesa per portare questo assioma nella sua Forma Skolemizzata è che si può associare all'esistenza di $b$ una certa funzione di $a$, in questo caso *l'inverso* di $a$. In altre parole, ad ogni $a$ si associa un elemento $f(a)$ tale che l'assioma sia verificato. Nell'esempio concreto, si arriva dunque a definire una nuova $L$-struttura $\mathscr{A}'$ in modo tale che

$$\mathscr{A}' \models \forall x (x*x^{-1} = e \wedge x^{-1}*x = e)$$

La funzione inverso è esattamente il "gioco" sotteso alla Skolemizzazione: ora, l'assioma legge che $\mathscr{A}'$ è un modello per l'assioma se e solo se per ogni $a \in A$ si ha che $\bar{a} * \bar{a}^{-1} = e$ e $\bar{a}^{-1} * \bar{a} = e$, postulando in qualche senso che la funzione $f(a) = b$ esiste ed è chiamata inverso.

È chiaro che questo processo non può preservare l'equivalenza logica per due motivi: il primo, superficiale, è che i due $L$-enunciati sono scritti in due linguaggi diversi. Il secondo, più profondo, è che quando si interpreta il nuovo simbolo $f$, c'è *un* modo per interpretarlo bene, ma esistono anche tutte le altre interpretazioni che possono far fallire la formula; l'unica cosa che si sa è che c'è *almeno una* interpretazione che possa far funzionare le cose.

**Processo di Skolemizzazione** Questo discorso è assolutamente generale ed è il cuore della Skolemizzazione:

$$\mathscr{A} \models A = \forall x_1 \cdots \forall x_n \exists y B$$

Senza perdità di generalità, si assume che in $A$ non occorrano quantificazioni ridondanti, cioè in $B$ non occorrono cose come $\forall x$, $\exists x$, in altre parole ogni occorrenza di $x_1, \cdots, x_n, y \in B$ è libera. In secondo luogo, in $B$ possono occorre altri quantificatori, pertanto $\exists y$ è solo l'occorrenza "più esterna".

Applicando la definizione di semantica di Tarski si legge che quanto è scritto è vero se e solo se per ogni $(a_1, \cdots, a_n) \in A^n$ esiste un $b \in A$ tale che

$$\mathscr{A} \models A = B[\bar{a}_1/x_1] \cdots [\bar{a}_n/x_n][\bar{b}/y]$$

il concetto di esistenza (di $b$ in questo caso) viene riletto come un modo per associare un immagine ad un argomento di una qualche funzione: esisterà quindi

$$f(a_1, \cdots, a_n) = b$$

Ciò che la Skolemizzazione "fa" è introdurre un simbolo di funzione $n$-ario nel linguaggio $f$ interpretandolo come la funzione $f$; dunque, la $L$-struttura iniziale viene modificata in una $L$-struttura estesa che deve interpretare anche $f$:

$$\mathscr{A}' \models A^s = \forall x_1 \cdots \forall x_n B[f(x_1, \cdots, x_n)/y]$$

creando quindi $A^s$, la Forma Skolemizzata di $A$. Allo stesso modo si passa da un linguagggio

$$L \rightsquigarrow L^s = (\mathscr{P}, F \cup \{f\}, \alpha, \beta^s)$$

con

$$\beta^s(f) = n$$

Sia $\mathscr{A} = (U, I)$ una $L$-struttura; allora scriviamo $(\mathscr{A}, h) = (U, I_h)$ per identificare la $L^s$-struttura tale che $I_h$ coincide con $I$ tranne che per $I_h(f)$, quindi $h : U^n \to U$ è una funzione $n$-aria.

**Equisoddisfacibilità** Definiamo

$$A := \exists x P(x)$$

e la sua forma Skolemizzata

$$A^s := P[c/x] = P(c)$$

e la $L$-struttura

$$\mathscr{A} = (U, I)$$

con

$$U = \{e_1, e_2\} \qquad I(P) = \{e_2\}$$

Chiaramente, si ha che

$$\mathscr{A} \models A$$

e che

$$\begin{aligned}(\mathscr{A}, e_2) \models A &\iff (\mathscr{A}, e_2) \models \exists x P(x) \\ &\iff \text{esiste } a \in U \text{ tale che } \mathscr{A} \models P(x)[\bar{a}/x] \\ &\iff \text{esiste } a \in U \text{ tale che } \mathscr{A} \models P(a)\end{aligned}$$

il quale esiste, poiché basta scegliere $a = e_2$ per verificare il tutto. Parimenti, si ha anche che

$$\begin{aligned}(\mathscr{A}, e_1) \models A &\iff (\mathscr{A}, e_1) \models \exists x P(x) \\ &\iff \text{esiste } a \in U \text{ tale che } \mathscr{A} \models P(x)[\bar{a}/x] \\ &\iff \text{esiste } a \in U \text{ tale che } \mathscr{A} \models P(a)\end{aligned}$$

poiché non è importante come si interpreta la costante $e_1$; l'utilizzo dell'esitenziale ci "salva" nonostante l'interpretazione della funzione di Skolem (in questo caso la costante $c$) sia erronea.

Cercando di interpretare, ora

$$\begin{aligned}(\mathscr{A}, e_1) \not\models A^s &\iff (\mathscr{A}, e_1) \models P(C) \\ &\iff I_{e_2}(c) \notin I_{e_2}(P) \\ &\iff e_1 \notin \{e_2\}\end{aligned}$$

Ossia, abbiamo dimostrato che

$$A \not\equiv A^s$$

e non possono pertanto essere logicamente equivalenti, ma saranno invece equisoddisfacibili. Ciò che però vale è questo:

$$\models A^s \to A$$

nonostante abbiamo appena dimostrato che non è vero

$$\not\models A \to A^s$$

*Dimostrazione di* $\models A^s \to A$. Sia data una formula in PNF

$$A := \forall x_1 \cdots \forall x_n \exists y B$$

Allora, si ha che la sua forma Skolemizzata è

$$A^s := \forall x_1 \cdots \forall x_n B[f(x_1, \cdots, x_n)/y]$$

Si suppone che esista una $L^s$-struttura che soddisfa $A^s$, definita

$$(\mathscr{A}, h) \quad \mathscr{A} = (U, I)$$

Allora,

$$\begin{aligned}
(\mathscr{A},h) \models A^s &\iff \text{per ogni } (a_1,\cdots,a_n) \in U^n (\mathscr{A},h) \models B[f(x_1,\cdots,x_n)/y][\bar{a_1},x_1]\cdots[\bar{a_n}/x_n] \\
&\iff \text{per ogni } (a_1\cdots,a_n) \in U^n (\mathscr{A},h) \models B[\bar{a_1}/x_1]\cdots[\bar{a_n}/x_n][f(x_1,\cdots,x_n)/y] \\
&\quad \text{si pone } b := h(a_1,\cdots,a_n) \quad \text{ricordandosi che } I_h(f) = f \\
&\iff \text{per ogni } (a_1,\cdots,a_n) \in U^n (\mathscr{A},h) \models B[\bar{a_1}/x_1]\cdots[\bar{a_n}/x_n][\bar{b}/y] \\
&\iff (\mathscr{A},h) \models \forall x_1 \cdots \forall x_n \exists y B \\
&\iff (\mathscr{A},h) \models A \\
&\therefore \models A^s \to A
\end{aligned}$$

□

Questo non basta a dimostrare che siano equisoddisfacibili. Mostreremo ora che $A$ e $A^s$ sono equisoddisfacibili: si deve mostrare che se $(\mathscr{A},h) \models A$ allora esiste $g : U^n \to U$ tale che $(\mathscr{A},g) \models A^s$.

*Dimostrazione.* Supponiamo che sia

$$\begin{aligned}
(\mathscr{A},h) \models A \iff & \\
&\text{per ogni } (a_1,\cdots,a_n) \in U^n (\mathscr{A},h) \models \text{ esiste } b \in U \\
&\text{tale che } (\mathscr{A},h) \models B[\bar{a_1}/x_1]\cdots[\bar{a_n}/x_n][\bar{b}/y]
\end{aligned}$$

Si pone ora

$$g(a_1,\cdots,a_n) = b$$

dato che $g$ sarà definito in questo modo per ogni $n$-pla $(a_1,\cdots,a_n) \in U^n$, allora questa definizione definisce una funzione $g : U^n \to U$. Ci si chiede ora quando vale

$$\begin{aligned}
(\mathscr{A},g) \models A^s &\iff \text{per ogni } (a_1,\cdots,a_n) \in U^n (\mathscr{A},g) \models B[f(x_1,\cdots,x_n)/y][\bar{a_1}/x_1]\cdots[\bar{a_n}/x_n] \\
&\iff \text{per ogni } (a_1,\cdots,a_n) \in U^n (\mathscr{A},g) \models B[\bar{a_1}/x_1]\cdots[\bar{a_n}/x_n][f(\bar{a_1},\cdots,\bar{a_n})/y] \\
&\iff \text{per ogni } (a_1,\cdots,a_n) \in U^n (\mathscr{A},g) \models B[\bar{a_1}/x_1]\cdots[\bar{a_n}/x_n][\bar{b}/y]
\end{aligned}$$

pertanto

$$\text{se } (\mathscr{A},h) \models A \to \text{ esiste } g : U^n \to U \text{ tale che } (\mathscr{A},g) \models A^s$$

□

### 8.1.3 Forma Normale Congiuntiva

Sia $A$ un $L$-enunciato. Siamo passati da $A$ ad $A^p \in PNF$ mantenendo l'equivalenza logica per poi passare a $A^s \in SNF$ mantenendo, in genere, l'equisoddisfacibilità. Ciò che manca per ricalcare il lavoro fatto riguardo la Logica Proposizionale, è portare il tutto in CNF:

$$A \xrightarrow{\equiv} A^p \xrightarrow{\text{equisod.}} (A^p)^s \rightsquigarrow ((A^p)^s)^c$$

Quindi, $A^{p^s}$ è nella forma $\forall x_1 \cdots \forall x_n M$, dove in $M$ non occorrono quantificatori. Inoltre, $A^{p^s}$ è un $L$-enunciato, pertanto $FV(M) \subseteq \{x_1,\cdots,x_n\}$. Consideriamo ora ogni occorrenza di formula atomica $P(t_1,\cdots,t_n)$ in $M$ come una lettera proposizionale; $M$ può dunque essere pensata come

una formula della Logica Proposizionale e il processo di trasformazione verso la CNF si ottiene applicando uno degli algoritmi visti, ottenendo

$$M \xrightarrow[\text{equisod.}]{\equiv} M^C$$

Concludendo, quindi che il processo di "traduzione" è

$$A \xrightarrow{\equiv} A^p \xrightarrow{\text{equisod.}} (A^p)^s \xrightarrow[\text{equisod.}]{\equiv} ((A^p)^s)^c$$

ovviamente mantenendo l'equisoddisfacibilità.

## 8.2 Risoluzione Automatica

### 8.2.1 Preprocessamento

Siamo partiti dal chiederci se

$$\Gamma \models A? \underset{0}{\rightarrow} \Gamma, \neg A \text{ insod.?} \xrightarrow[1]{\text{se } \Gamma \text{ finito}} \gamma_1 \wedge \cdots \wedge \gamma_n \wedge \neg A \text{ insod.?} \underset{2}{\rightarrow} (\gamma_1 \wedge \cdots \wedge \gamma_n \wedge \neg A)^{p^{s^c}} \text{ insod.?}$$

manca un'ultima trasformazione. Infatti, un enunciato in forma prenessa, Skolemizzata e confiuntiva è nella forma

$$F = \forall x_1 \cdots \forall x_n C_1 \wedge C_2 \wedge \cdots \wedge C_N$$

dove ogni $C_i = (l_1 \vee l_2 \vee \cdots \vee l_u)$ è chiamata clausola dove ogni

$$l_{ij} = \begin{cases} P(t_1, \cdots, t_n) \\ \neg P(t_1, \cdots, t_n) \end{cases}$$

è chiamato letterale, per un qualche $P(\cdot)$ formula atomica. Se

$$F = \forall x_1 \cdots \forall x_n (C_1 \wedge C_2 \wedge \cdots \wedge C_N)$$

con $F \in PNF \cap SNF \cap CNF$, operiamo i seguenti passaggi di notazione:

- Omettiamo i quantificatori. Possiamo farlo per due ragioni:
  1. Scriverli esplicitamente non aggiunge nessuna informazione: sappiamo che tutte le variabili che occorrono sono quantificate universalmente.
  2. $\mathscr{A} \models F$ se e solo se $\mathscr{A} \models \forall[F]$ per ogni $F$ formula ben formata.
- Notazione a sequenti. Si riscrive

  $$F = C_1 \wedge \cdots \wedge C_2$$

  come

  $$S_F = \{C_1, \cdots, C_n\}$$

  Questa riscrittura ha vari vantaggi:
  - Si può mettere in PNF, SNF e CNF ogni formula in $\Gamma \cup \{\neg A\}$ indipendentemente

– Se $\Gamma$ è infinito $S_\gamma$ sarà un insieme infinito di clausole finite

Inoltre, si riscrive anche ogni clausola: se ora è

$$C_i = l_{i1} \vee \cdots \vee l_{iu}$$

si può riscrivere

$$C_i = \{l_{ij} \text{ che occorrono negati}\} \cup \{l_{ij} \text{ che occorrono positivi}\}$$

e i due insiemi si definiscono, rispettivamente, $N_i$ e $P_i$. e si riscrive, in conclusione,

$$C_i \quad N_i \implies P_i$$

Quest'ultima notazione si chiama *notazione a sequenti* della clausola $C_i$.

**Esempio**

Ci chiediamo se sia vero che

$$\forall x \exists y (R(x,y) \to Q(f(x))) \models \forall x \exists y (R(x,y) \to Q(f(y)))$$

Il primo passo è riportare il tutto in una domanda di insoddisfacibilità:

$$\forall x \exists y (R(x,y) \to Q(f(x))), \neg(\forall x \exists y (R(x,y) \to Q(f(y)))) \text{ insod.?}$$

Si procede, ora, mettendo in PNF, SNF e CNF le due formule. La prima,

$$\forall x \exists y (R(x,y) \to Q(f(x)))$$

è già in PNF. Si procede sostituendo a $y$ un simbolo di funzione, creando la funzione di Skolem utilizzando un nuovo simbolo:

$$\forall x (R(x,s(x)) \to Q(f(x)))$$

e infine in CNF (o a sequenti):

$$\forall x (\neg R(x,s(x)) \vee Q(f(x)))$$

concludendo con

$$C_1 := \{\neg R(x,s(x)), Q(f(x))\}$$

e scritto in forma a sequenti

$$S_1 := R(x,s(x)) \implies Q(f(x))$$

La seconda formula

$$\neg(\forall x \exists y (R(x,y) \to Q(f(y)))) = \exists x \forall y \neg (R(x,y) \to Q(f(y)))$$

è già in PNF, e per portarla in SNF si sostituisce a $x$ un simbolo di funzione (costante), utilizzando un nuovo simbolo:

$$\forall y \neg (R(c,y) \to Q(f(y)))$$

e si porta in CNF:

$$\neg(R(c,y) \to Q(f(y))) \equiv R(c,y) \wedge \neg Q(f(y))$$

ottenendo due clausole:

$$C_2 = \{R(c,y)\}$$

$$C_3 = \{\neg Q(f(y))\}$$

e due sequenti:

$$S_2 : \Longrightarrow R(c,y)$$

$$S_3 : Q(f(y)) \Longrightarrow$$

E si riscrive, infine,

$$\{R(x,s(x)) \Longrightarrow Q(f(x)), \Longrightarrow R(c,y), Q(f(y)) \Longrightarrow\} \text{ insod.?}$$

### 8.2.2 Teoria di Herbrand

Sia $S$ un insieme di clausole nella Logica dei Predicati; quindi, si immagina $S$ ottenuto mettendo in PNF, SNF, CNF uno statement del tipo $\Gamma \models A$, di cui vogliamo stabilire la soddisfacibilità. Questo sarà eseguito riducendo $S$ insoddisfacibile a un problema di insoddisfacibilità nella Logica Proposizionale.

Il primo passo è la seguente definizione:

**Definizione** (Astrazione Proposizionale)**.** *L'**astrazione proposizionale** è una funzione che associa iniettivamente ad ogni formula atomica ground (quindi nella forma $P(t_1,\cdots,t_n)$ dove ogni $t_i$ è un termine ground, ossia non appaiono variabili) una lettera proposizionale, denotata con $p_{P(t_1,\cdots,t_n)}$.*

Per esempio, se una clausola $C$ espressa in sequenti è

$$C : A_1,\cdots,A_u \Longrightarrow B_1,\cdots,B_u$$

la sua astrazione proposizionale è la clausola ottenuta astraendo ogni lettera:

$$C : p_{A_1},\cdots,p_{A_u} \Longrightarrow p_{B_1},\cdots,p_{B_u}$$

La **Teoria di Herbrand** ha suo culmine in quello che è ormai noto come **Teorema di Herbrand**, anche se nella letteratura ci sono delle perplessità riguardo il suo status di *teorema*:

**Teorema** (di Herbrand)**.** *Un insieme di clausole S della Logica dei Predicati è insoddisfacibile se e solo se esiste un insieme finito di istanze ground di clausole di S la cui astrazione proposizionale è (proposizionalmente) insoddisfacibile.*

Per approfondire la Teoria di Herbrand, introduciamo delle nozioni.

**Definizione** (Istanziazione Ground)**.** *Sia S un insieme di clausole del Primo Ordine, e sia G un insieme di termini ground. Allora, con la notazione $S/G$ indichiamo l'istanziazione ground si S su G, vale a dire l'insieme di tutte le clausole (che risulteranno ground) ottenute da ciascuna clausola $C \in S$ sostituendo in G ogni sua variabile x con qualche $t \in G$, in modo simultaneo, in tutti i modi possibili.*

**Esempio** Sia

$$\begin{aligned} S &:= \{\{\implies P(x), D(x)\}, \{P(x), D(x) \implies\}\} \\ &= \{\{P(x), D(x)\}, \{\neg P(x), \neg D(x)\}\} \\ &= (P(x) \vee D(x)) \wedge (\neg P(x) \vee \neg D(x)) \end{aligned}$$

Sia $G$ ad esempio

$$G = \{a, b, c\}$$

Allora $S/G$ è

$$\begin{aligned} S/G :=& \{\{\implies P(a), D(a)\}, \{\implies P(b), D(b)\}, \{\implies P(c), D(c)\}, \\ & \{P(a), D(a) \implies\}, \{\implies P(b), D(b)\}, \{P(c), D(c) \implies\}\} \end{aligned}$$

Sia ora

$$G' = \{a, f(b,c)\}$$

allora $S/G'$ è

$$\begin{aligned} S/G :=& \{\{\implies P(a), D(a)\}, \{\implies P(f(b,c)), D(f(b,c))\}, \\ & \{P(a), D(a) \implies\}, \{P(f(b,c)), D(f(b,c)) \implies\}\} \end{aligned}$$

Siamo interessati ad istanziare $S$ su un particolare insieme $G$, ossia quello che ci permette migliormente di studiare il nostro problema: questo insieme è chiamato *Universo di Herbrand*:

**Definizione** (Universo di Herbrand)**.** *L'Universo di Herbrand $H_s$ di un insieme di clausole S del Primo Ordine è definito come segue. Sia $F_S^0$ l'insieme di tutti i simboli di funzione (e quindi anche le costanti) che occorrono in S.*

- *Se $F_S^0$ non contiene costanti, allora si pone $F_S = F_S^0 \cup \{\mathfrak{z}\}$, dove $\mathfrak{z}$ è una costante nuova.*
- *Se $F_S^0$ contiente qualche costante, allora $F_S := F_S^0$.*

*$H_S$ è per definizione l'insieme di tutti i termini costruibili utilizzando i simboli di $F_S$, che sono necessariamente ground.*

**Esempi**

**1** Sia

$$S := \{\{U(x) \implies M(x)\}, \{\implies U(s)\}, \{M(s) \implies\}\}$$

Allora

$$F_S^0 = \{s\} \quad F_S = \{s\} \quad H_S = \{s\}$$

**2** Sia

$$S := \{\{\implies P(x), D(x)\}, \{P(x), D(x) \implies\}\}$$

Allora

$$F_S^0 = \emptyset \quad F_S = \{\mathfrak{z}\} \quad H_S = \{\mathfrak{z}\}$$

**3** Sia

$$S := \{\{ \Longrightarrow P(f(x)), D(x)\}\}$$

Allora

$$F_S^0 = \{f(\cdot)\} \quad F_S = \{f(\cdot), \mathfrak{z}\} \quad H_S = \{\mathfrak{z}, f(\mathfrak{z}), f(f(\mathfrak{z})), \cdots\}$$

Argomenteremo che $S$ è insoddisfacibile se e solo se $(S/H_S)^*$, ossia l'astrazione proposizionale dell'istanziazione di $S$ sul suo Universo di Herbrand $H_S$, è insoddisfacibile; quest'ultimo insieme è spesso infinito, in quanto - oltre ai simboli di funzione "nativi" del linguaggio su cui è costruito $S$ - è costruito a seguito della Skolemizzazione dell'insieme $S$.

I due teoremi "di Herbrand" enunciati seguono entrambi piuttosto facilmente dalla semantica di Herbrand, che è il vero cuore della questione.

### 8.2.3 Semantica di Herbrand

La semantica di Herbrand afferma che è necessario considerare solo le $L$-strutture di Herbrand.

**Definizione** ($L$-struttura di Herbrand)**.** *Una L-struttura di Herbrand è un tipo particolare, più vincolato, di L-struttura, le quali vengono chiamate, a questo punto, di Tarski.*

*Sia allora*

$$\mathscr{A} = (U, I)$$

*una L-struttura di Tarski. Deve quindi essere:*

- $U \neq \emptyset$*, oltre a questo U è arbitrariamente scelto.*
- $I(c) \in U$*, arbitrariamente scelto.*
- $I(f)$*, con* $\beta(f) = n$*, è definito* $I(f) : U^n \to U$*, arbitrariamente scelto.*
- $I(P)$*, con* $\alpha(P) = n$*, è* $I(P) \subseteq U^n$ *arbitrariamente scelto.*

*Sia ora*

$$\mathscr{H} = (H_S, H)$$

*una L-struttura di Herbrand. Deve quindi essere:*

- $H_S$ *è l'Universo di Herbrand, il quale è diverso dall'insieme vuoto (*$H_S \neq \emptyset$*), ma ha una costruzione ben precisa.*
- $H(c) \in H_S$*, ma ben definito:* $H(c) = c$*.*
- $H(f)$*, con* $\beta(f) = n$*, è definito* $H(f) : H_S^n \to H_S$*, con definizione fissata:*

$$(H(f))(t_1, \cdots, t_n) := f(t_1, \cdots, t_n)$$

  *con* $t_1, \cdots, t_n$ *termini ground.*

- $H(P)$*, con* $\alpha(P) = n$*, è definito* $H(P) \subseteq H_S^n$*, con interpretazione arbitrariamente scelta.*

**Definizione** (Teorema Fondamentale della Teoria di Herbrand)**.** *Sia S un insieme di clausole del Primo Ordine. Allora, S ha un modello (di Tarski), ossia S è soddisfacibile, ossia esiste una L-struttura tarskiana* $\mathscr{A} = (U, I)$ *tale che* $\mathscr{A} \models S$ *se e solo se S ha un modello di Herbrand, ossia se esiste una L-struttura di Herbrand* $\mathscr{H} = (H_s, H)$ *tale che* $\mathscr{H} \models S$*.*

Ora, ci dobbiamo chiedere quale sia la relazione tra

$$(S/H_S)^* \qquad \text{e} \qquad \mathscr{H}_S \models S$$

La relazione è semplice. Sia $\mathfrak{v}_{\mathscr{H}_S}$ : Lettere Proposizionali $\to \{0,1\}$, ossia è un assegnamento, tale che $\mathfrak{v}_{\mathscr{H}_S}(p_{P(t_1,\cdots,t_n)}) = 1$ se e solo se $\mathscr{H}_S \models P(t_1,\cdots,t_n)$ e 0 altrimenti. Viceversa, data $\mathfrak{v}$ : Lettere Proposizionali $\to \{0,1\}$ si definisce $\mathscr{H}_\mathfrak{v} = (H_s, H_\mathfrak{v})$ tale che

$$\begin{cases} (t_1,\cdots,t_n) \in H_\mathfrak{v}(P) \iff \mathfrak{v}(p_{P(t_1,\cdots,t_n)}) = 1 \\ (t_1,\cdots,t_n) \notin H_\mathfrak{v}(P) \iff \mathfrak{v}(p_{P(t_1,\cdots,t_n)}) = 0 \end{cases}$$

È facile verificare, dunque, che $\mathscr{H}_S \models S$ se e solo se $\mathfrak{v}_{\mathscr{H}_S} \models (S/H_S)^*$.

**Definizione** (Teorema Riassuntivo (Herbrand, compattezza, completezza, calcolo Refutazionale $R^*$, DPP e DPLL))**.** *Dato un insieme S di clausole della Logica del Primo Ordine, è equivalente affermare:*

1. *S è insoddisfacibile*
2. *S non ha modelli di Herbrand*
3. $(S/H_S)^*$ *è proposizionalmente insoddisfacibile*
4. *Esiste* $S' \subseteq_\omega (S/H_S)^*$ *tale che* $S'$ *è proposizionalmente insoddisfacibile*
5. *Esiste* $H' \subseteq_\omega H_S$ *tale che* $(S/H')^*$ *è proposizionalmente insoddisfacibile*
6. *Esiste* $H' \subseteq_\omega H_S$ *e* $h \geq 0$ *tale che* $\quad \square \in R^h((S/H)^*)$
7. *Esiste una refutazione* $DPP \vdash_R (S/H')^* h$ *o* $DPLL \vdash_R (S/H')^*$

*La catena* $1 \cdots 7$ *fornisce un algoritmo per semidecidere l'insoddisfacibilità di S.*

### 8.2.4 Metodi Refutazionali

Sia

$$S = \{R(x,s(x)) \implies Q(f(x)), \implies R(c,y), Q(f(y)) \implies\}$$

Si procede alla risoluzione del problema dell'insoddisfacibilità, seguendo i passi definiti precedentemente:

$$F_S = \{c,s,f\} \quad H_S = \{c,s,f,f(c),s(c),s(s(c)),f(f(c)),s(f(c)),f(s(c)),s(s(s(c))),\cdots\}$$

Pertanto, è impossibile costruire $(S/H_S)^*$ direttamente, in quanto è anch'esso infinito. Si sceglie un insieme finto di $H_S$, per esempio $H = \{c\}$:

$$(S/H)^* = \{R(c,s(c)) \implies Q(f(c)), \implies R(c,c), Q(f(c)) \implies\}$$

Possiamo provare ad effettuare la risoluzione

$$\frac{R(c,s(c)) \implies Q(f(c)) \qquad Q(f(c)) \implies}{R(c,s(c)) \implies}$$

e rimaniamo con

$$(S/H)^* = \{R(c,s(c)) \implies, \implies R(c,c)\}$$

coi quali non si può fare niente, in quanto le astrazioni proposizionali sono diverse. Si amplia, quindi, $H' = \{c, s(c)\}$:

$$\begin{aligned}(S/H')^* = &\{R(c,s(c)) \implies Q(f(c)), \implies R(c,c), Q(f(c)) \implies, \\ &R(s(c),s(s(c))) \implies Q(f(s(c))), \implies R(c,s(c)), Q(f(s(c))) \implies\}\end{aligned}$$

nuovamente

$$\frac{R(c,s(c)) \implies Q(f(c)) \qquad Q(f(c)) \implies}{R(c,s(c)) \implies}$$

e

$$(S/H')^* = \{R(c,s(c)) \implies, \implies R(c,c), R(s(c),s(s(c))) \implies Q(f(s(c))), \implies R(c,s(c)), Q(f(s(c))) \implies\}$$

e si può risolvere

$$\frac{R(c,s(c)) \implies \qquad \implies R(c,s(c))}{\implies}$$

pertanto $(S/H')^*$ è insoddisfacibile $\rightarrow S/H_S$ è insoddisfacibile $\rightarrow S$ è insoddisfacibile, quindi il problema iniziale era vero. Tuttavia, già da questo esempio si può intuire che c'è qualche difetto sul quale si potrebbe lavorare. Inizialmente, bisognerebbe trovare un modo per generare un sottoinsieme finito $H \subseteq H_S$ utile dal quale si generano solo le clausole che sembrano utili, senza generare clausole inutili. L'idea è sviluppare un metodo che vada in questa direzione.

**Esempio: "paradosso" dell'uomo col cappello**

$$\models \exists x(P(x) \rightarrow \forall y P(y))$$

si asserisce che sia una verità logica, ma è davvero così? Possiamo utilizzare le tecnologie a nostra disposizione:

$\neg(\exists x(P(x) \rightarrow \forall y P(y)))$ è insoddisfacibile

$$\begin{aligned}&\forall x \neg(P(x) \rightarrow \forall y P(y)) \\ &\equiv \forall x \neg(\neg P(x) \vee \forall y P(y)) \\ &\equiv \forall x(P(X) \wedge \neg(\forall y P(y))) \\ &\equiv \forall x \exists y(P(x) \wedge \neg P(y))\end{aligned}$$

che si può Skolemizzare in

$\forall x(P(x) \wedge \neg P(f(x)))$ insoddisfacibile?

Si traduce in

$$C_1 := \{\implies P(x)\}, C_2 := \{P(f(x)) \implies\}$$

quindi si definisce

$$S = \{\implies P(x), P(f(x)) \implies\}$$

e

$$F_S = \{f(\cdot), \mathfrak{z}\} \quad H_S = \{\mathfrak{z}, f(\mathfrak{z}), \cdots\}$$

Inizialmente si istanzia

$$H = \{\mathfrak{z}\}$$

e si ottiene

$$S = \{ \Longrightarrow P(z), P(f(z)) \Longrightarrow \}$$

ma non si può fare nulla. Si istanzia

$$H = \{\mathfrak{z}, f(\mathfrak{z})\}$$

e si ottiene

$$S = \{ \Longrightarrow P(z), P(f(z)) \Longrightarrow \} \cup \{ \Longrightarrow P(f(z)), P(f(f((z))) \Longrightarrow \}$$

A questo punto si può risolvere:

$$\frac{P(f(z)) \Longrightarrow \qquad \Longrightarrow P(f(z))}{\Longrightarrow}$$

e pertanto $S$ è insoddisfacibile, e di contro è vero che

$$\models \exists x(P(x) \to \forall y P(y))$$

**Difetti**

Sia

$$S = \{\{ \Longrightarrow P(x,y), Q(x)\}, \{P(x, f(x)) \Longrightarrow \}\}$$

si costruiscono

$$F_S = \{f(\cdot), \mathfrak{z}\} \qquad H_S = \{\mathfrak{z}, f(\mathfrak{z}), f(f(\mathfrak{z})), \cdots\}$$

la risoluzione che possiamo opearare è, ad esempio,

$$\frac{\Longrightarrow P(\mathfrak{z}, f(\mathfrak{z})), Q(\mathfrak{z}) \qquad P(\mathfrak{z}, f(\mathfrak{z})) \Longrightarrow}{\Longrightarrow Q(\mathfrak{z})}$$

istanziando $x = \mathfrak{z}, y = f(\mathfrak{z})$; ma questa è una considerazione generale e prendendo invece $x = f(\mathfrak{z}), y = f(\mathfrak{z})$ si ottiene

$$\frac{\Longrightarrow P(f(\mathfrak{z}), f(f(\mathfrak{z}))), Q(f(\mathfrak{z})) \qquad P(f(\mathfrak{z}), f(f(\mathfrak{z})) \Longrightarrow}{\Longrightarrow Q(f(\mathfrak{z}))}$$

e si possono ottenere infinite altre risoluzioni. Per il nostro apparato concettuale, tutte queste risoluzioni sono equivalenti, ma noi vediamo qualcosa in più: vi è infatti un *pattern* che ci piacerebbe catturare, e tutte le ripetizioni diventerebbero l'istanza di un'unica risoluzione; il pattern è chiaro:

$$y \mapsto f(x)$$

questa è una sostituzione ma non è una istanziazione, in quanto $y$ non "diventa" un termine ground, ma possiede una variabile. Tuttavia, se si riesce a disegnare un quadro in cui questo è lecito, si risolve direttamente

$$\frac{\Longrightarrow P(x, f(x)), Q(x) \qquad P(x, f(x)) \Longrightarrow \qquad : y \mapsto f(x)}{\Longrightarrow Q(x)}$$

E da $\Longrightarrow Q(x)$ si può ritrovare, con le istanziazioni, tutta la famiglia di risoluzioni che si possono generare una per volta.

### 8.2.5 Teoria delle Sostituzioni

Quali clausole del Primo Ordine generano per istanziazione la clausola vuota? Solo la clausola vuota stessa. L'approccio proposto poco fa, allora, è promettente, poiché:

- Ci sono delle sosstituzioni ottime che sono "facili" da calcolare e trovare
- ("Lifting") Facendo prima le risoluzioni a livello non ground si riesce a posticipare la fase d'istanziazione (ground) da non avere più bisogno di farla, poiché se l'insieme di sequenti è insoddisfacibile si troverà la clausola vuota

Abbiamo già discusso di sostituzioni riguardo la Logica Proposizionale, in cui si sostituivano delle lettere proposizionali con una Formula. Nella Logica dei Predicati, abbiamo discusso di sostituzione di variabili individuali con termini. Anche ora discuteremo di qualcosa di simile, ma ora la notazione cambia in quanto sarà proprio la sostituzione il punto focale del discorso.

**Definizione** (Sostituzione)**.** *Una* ***sostituzione*** $\sigma$ *è una mappa*

$$\sigma : Var \to \tau_L$$

*con* $\tau_L$ *che indica l'insieme dei termini costruibili sul linguaggio L, anche non ground;* $\sigma$ *ha una proprietà importante che rende la computazione delle sostituzioni molto semplice: l'insieme delle variabili x tali che sono "mosse" dalla sostituzione, ossia* $x \neq \sigma(x)$*, è finito. L'insieme delle variabili "mosse" viene chiamato dominio di* $\sigma$*.*

Se il dominio di $\sigma$ è

$$dom(\sigma) = \{x_1, \cdots, x_k\}$$

allora si può visualizzare $\sigma$ nel seguente modo:

$$\sigma : x_1 \mapsto t_1, \cdots, x_k \mapsto t_k$$

Questa notazione sostituisce la notazione $([t_i/x_i])$ utilizzata fino ad ora.

**Definizione** (Estensione Canonica)**.** *Una sostituzione* $\sigma$ *si estende canonicamente ad una mappa*

$$\hat{\sigma} : \tau_L \mapsto \tau_L$$

*con la solita definizione di estensione canonica definita induttivamente:*

- ***base***
$$\hat{\sigma}(x) := x \qquad per\ ogni x \in Var$$
- ***passo induttivo***
$$\hat{\sigma}(f(t_1, \cdots, t_n)) := f(\hat{\sigma}(t_1), \cdots, \hat{\sigma}(t_n))$$

*Si può, inoltre, estendere* $\sigma$ *non solo ai termini ma a tutte le espressioni che ci servono, in particolar modo alle espressioni atomiche: Sia* $P(t_1, \cdots, t_n)$ *un'atomica del nostro linguaggio. Allora si definisce*

$$\hat{\sigma}(P(t_1, \cdots, t_n)) := P(\hat{\sigma}(t_1), \cdots, \hat{\sigma}(t_n))$$

*Sia E una qualsiasi espressione d'interesse: allora*

$$\hat{\sigma}(E)$$

*è l'espressione sostituendo tramite* $\sigma$ *simultaneamente tutti i termini che appaiono in E.*

Scriveremo spesso $E\sigma$ per indicare $\hat{\sigma}$ applicato all'espressione $E$ $\hat{\sigma}(E)$

**Proprietà delle Sostituzioni**

- **Composizione** Se $\gamma, \tau : Var \to \tau_L$ la sostituzione composta $\sigma\tau$ è data da

  $$\sigma\tau(x) = \hat{\tau}(\sigma(x))$$

  Ovviamente $\hat{\sigma}\tau$ applicata su una certa espressione $E$, denotato quindi $E\sigma\tau$ si ottiene applicando $\hat{\tau}(\sigma(x_i))$ ad ogni variabile occorrente in $E$.

- **Sostituzione Identica** la sostituzione $id$ è definita

  $$id(x_i) = x_i \quad \text{per ogni } x_i \in Var$$

  e ha dominio vuoto ($dom(id) = \emptyset$); gode della proprietà di composizione seguente:

  $$\sigma id = id\sigma = \sigma \text{ per ogni } \sigma : Var \to \tau_L$$

  quindi svolge il ruolo di elemento neutro.

- **Rinomine** una rinomina è una sostituzione $\rho : Var \to \tau_L$ che è una permutazione del suo dominio, ossia

  $$\rho(x_i) = x_j$$

  e $\rho$ è biiettiva sul suo dominio, quindi ogni rinomina è invertibile; esiste pertanto una certa sostituzione $\rho'$ tale che $\rho'(\rho(x_i)) = x_i$, in altre parole $\rho\rho' = \rho'\rho = id$.

- **Sostituzione Generale** $\sigma$ è detta *più generale* di $\tau$ se esiste una sostituzione $\delta$ tale che $\sigma\delta = \tau$. Per esempio, se

  $$\delta : x \mapsto f(z), y \mapsto z$$

  e

  $$\tau : x \mapsto f(z), y \mapsto c, z \mapsto c$$

  allora $\sigma$ è più generale di $\tau$ ($\sigma \geq \tau$) in quanto se definiamo

  $$\delta : z \mapsto c$$

  si ha che $\tau = \sigma\delta$.

- **Preordine** La relazione "essere più generale" $\sigma \geq \tau$ è riflessiva e transitiva, ma in generale non è antisimmetrica; esistono sostituzioni distinte (come le rinomine) $\sigma \neq \tau$ tali che $\sigma$ è più generale di $\tau$ e al contempo $\tau$ è più generale di $\sigma$. Quanto la relazione è riflessiva e transitiva ma non antisimmetrica, la relazione è definita *preordine* e non *ordine*.

  Se $\sigma \neq \tau$ e $\sigma \geq \tau$, $\tau \geq \sigma$, allora esistono due rinomine $\rho_1$, $\rho_2$ tali che

  $$\sigma\rho_1 = \tau \quad \tau\rho_2 = \sigma$$

- **Equivalenza Sostituzionale** Se $\sigma \geq \tau$ e $\tau \geq \sigma$, dichiariamo che $\sigma \equiv \tau$.

- **Ordine Parziale** L'insieme delle sostituzioni con la relazione "essere più generale" quozientato rispetto a $\equiv$ è un ordine parziale, ossia

  $$\{[\sigma]_{\equiv} : \sigma : Var \to \tau_L\}$$

  è un ordine parziale rispetto a $\geq$.

L'ultimo concetto che ci serve per arrivare alla definizione della Risoluzione Sollevata o Liftata è il seguente:

**Definizione** (Unificazione)**.** *Un problema di Unificazione è una lista finita di coppie di termini*

$$t_1 \stackrel{?}{=} u_1, \cdots, t_n \stackrel{?}{=} u_n$$

La soluzione ad un problema di unificazione è una sostituzione che verifica tutte le uguaglianze nella lista di coppie di termini, ossia una sostituzione

$$\sigma : Var \to \tau_L$$

tale che

$$\hat{\sigma}(t_1) = \hat{\sigma}(u_1), \cdots, \hat{\sigma}(t_n) = \hat{\sigma}(u_n)$$

**Definizione** (Unificatore)**.** *Una soluzione $\mu : Var \to \tau_L$ è detta **unificatore** di massima generalità (most general unifier) se e solo se $\mu$ è più generale di ogni altra soluzione al problema di unificazione.*

**Teorema** (di Unificazione)**.** *Ogni problema di unificazione $U$ o ammette una soluzione che è un MGU (ossia esiste $\mu$ che è un most general unifier) oppure non ammette alcuna soluzione. Si noti che se $mgu(U)$ esiste non è necessariamente unico, in quanto è unica la classe di equivalenza di $mgu(U) = [\mu]_{\equiv}$.*

*Esistono algoritmi che su input $U$ trovano velocemente $\mu = mgu(U)$ se esiste, e si fermano se $mgu(U)$ non esiste.*

Ci interesserà risolvere problemi del tipo

$$P(t_1, \cdots, t_m) \stackrel{?}{=} Q(s_1, \cdots, s_n)$$

per $P$ simbolo di predicato $m$-ario e $Q$ simbolo di predicato $n$-ario. In altre parole ci interessa unificare due atomiche. Si noti che il problema appena esposto è unificabile se $P = Q$ e dunque $m = n$. Il problema si riduce a

$$P(t_1, \cdots, t_n) \stackrel{?}{=} P(s_1, \cdots, s_n)$$

che a sua volta si riduce a

$$U : t_1 \stackrel{?}{=} s_1, \cdots, t_n \stackrel{?}{=} s_n$$

### 8.2.6 Calcolo R della risoluzione liftata

Si supponga di avere le seguenti regole scritte in forma a sequenti:

$$\{\{\Gamma \implies \Delta, A\}, \{B, \Gamma' \implies \Delta'\}\}$$

dove le variabili occorrenti in $\Gamma \implies \Delta, A$ sono disgiunte da quelle in $B, \Gamma' \implies \Delta'$; inoltre si può sempre rispettare questo vincolo attraverso le rinomine.

$A$ e $B$ sono proposizioni atomiche, mentre $\Gamma, \Delta, \Gamma', \Delta'$ sono insiemi di proposizioni atomiche.

Se si riesce a calcolare un $\mu = mgu(A, B)$, allora si può risolvere su $R(\mu(A), \mu(B))$:

$$\frac{\Gamma \implies \Delta, A \qquad B, \Gamma' \implies \Delta'}{\Gamma\mu, \Gamma'\mu, \implies \Delta\mu, \Delta'\mu}$$

Questa regola incarna il calcolo R. Da sola non basta a garantire la completezza refutazionale, ma è necessario aggiungere almeno una delle seguenti due regole di fattorizzazione:

- Fattorizzazione Destra se si ha un sequente $\Gamma \implies \Delta, A, B$ e $\mu = mgu(A,B)$, allora si può dedurre
$$\Gamma_\mu \implies \Delta_\mu, A_\mu$$
sostanzialmente unificando $A$ e $B$ e tenendo un'unica copia dei due.

- Fattorizzazione Sinistra se si ha un sequente $\Gamma, A, B \implies \Delta$ e $\mu = mgu(A,B)$, allora si può dedurre
$$\Gamma_\mu, A_\mu \implies \Delta_\mu$$
sostanzialmente unificando $A$ e $B$ e tenendo un'unica copia dei due.

Una refutazione di un insieme di clausole $S$ nel calcolo R è una successione di clausole

$$C_1, C_2, \cdots, C_u$$

tale che:

1. $C_u = \implies$
2. per ogni $C_i$ o $C_i$ è un membro di $S$ eventualmente rinominato, oppure è ottenuto da eventuali rinomine di $C_j, C_k$ con $(j,k < i)$ per risoluzione o fattorizzazione (destra o sinistra).

Il calcolo R è refutazionalmente completo per ogni insieme di clausole $S$ e quindi per la logica del Primo Ordine. Allora, se si ha un probelma del tipo

$$\Gamma \overset{?}{\models} A$$

si trasforma in un insieme di clausole del Primo Ordine $S$ e si tenta di refutarlo col calcolo $R$. Se questo accade, ossia $S \vdash_R \implies$ , allora $S$ è insoddisfacibile e $\Gamma \models A$. Se $\Gamma \models A$, allora $S$ è insoddisfacibile e pertanto deve essere $S \vdash_R \implies$ .

**Considerazioni Finali**

La completezza refutazionale di $R$ mostra che la logica del Primo Ordine è (almeno) semidecidibile. Almeno, in quanto il Teorema di Church afferma che non si può fare di meglio, ossia la Logica del Primo Ordine è semidecidibile ma non decidibile.

Sia $\Gamma \models A$ e $S$ l'insieme di clausole derivanti. $S$ potrebbe anche essere infinito. Sia, ora, $H_S$ l'universo di Herbrand relativo. Sia

$$S_1 \subseteq_\omega S_2 \subseteq_\omega \cdots \subseteq_\omega S$$

un insieme di sottoinsiemi di $S$ tali che $\bigcup_{i \in \omega} S_i = S$ e per ogni $i \in \omega$ sia $H^i$ l'universo di Herbrand di $S_i$. Chiaramente, si ha anche che

$$H^1 \subseteq H^2 \subseteq \cdots \subseteq H_s$$

e $\bigcup_{i \in \omega} H^i = H_s$. Per ogni $i$ si definisce

$$H^i_1 \subseteq_\omega H^i_2 \subseteq_\omega \cdots \subseteq_\omega H^i$$

Per ogni $k \in \omega$ e per ogni $i, j, i+j=k$ si applica DPP o DPLL a $S_i/H^i_j$. Se si trova la clausola vuota, allora $S$ è insoddisfacibile. Se non si trova, allora si passa a $k = k+1$. In questo modo si riescono a perlustrare tutti i sottoinsiemi finiti dello spazio di ricerca, e se $S$ è insoddisfacibile prima o poi si trova.

La morale sottesa a questa considerazione è che se $S$ è insoddisfacibile prima o poi si mostra che lo è. Se $S$ è soddisfacibile non ci sono criteri generali per fermare l'algoritmo.

**Definizione** (Completezza Formale)**.** *Una teoria $\Gamma$ si dice Formalmente Completa se e solo se per ogni enunciato $A$ o $\Gamma \vdash A$ oppure $\Gamma \vdash \neg A$. Per Completezza e Correttezza Semantica, questo equivale a dire che $\Gamma \models A$ oppure $\Gamma \models \neg A$.*

**Teorema** (Teorema di Incompletezza di Gödel)**.** *Se $T$ è una teoria assiomatizzabile, consistente e che esprime l'Aritmetica di Peano, allora esiste $\phi$ tale che*

$$T \nvdash \phi \qquad T \nvdash \neg\phi$$

*e, inoltre, $\mathcal{N} = (\mathbb{N}, +, *, 0, s)$ è tale che $\mathcal{N} \models \phi$.*

**Definizione** (Aritmetica di Presburger)**.** *Sostanzialmente, è l'Aritmetica di Peano senza moltiplicazione. PrA è formalmente completa e decidibile in tempo doppiamente esponenziale.*

**Definizione** (Aritmetica di Robinson)**.** *Sostanzialmente, è l'Aritmetica di Peano senza schema di induzione, più un assioma che sostituisce alcune particolarità dello schema di induzione:*

$$\forall x(\neg(x=0) \rightarrow \exists y(s(y)=x))$$

*Si può dimostrare che, pur senza induzione, è altrettanto formalmente incompleta, come tutte le sue estensioni consistenti e assiomatizzabili, come PA stessa. RA è semidecidibile ma non decidibile. Tuttavia, RA è finitamente assiomatizzabile.*

**Teorema** (Teorema di Church)**.** *Sia*

$$RA = \{A_1, A_2, \cdots, A_7\}$$

*e sia*

$$C = A_1 \wedge \cdots \wedge A_7$$

*$C$ è un enunciato nell'Aritmetica di Robinson. Sia $A$ un enunciato arbitrariamente scelto. Si suppone, per assurdo, che la Logica del Primo Ordine sia decidibile. Allora*

$$C \rightarrow A$$

*è un enunciato del Primo Ordine, dunque è decidibile se $\models C \rightarrow A$ oppure $\nvDash C \rightarrow A$. Se fosse $\models C \rightarrow A$, allora avremmo*

$$\frac{C \qquad C \rightarrow A}{A}$$

*per modus ponens, e che quindi $RA \models A$. Se invece fosse $\nvDash C \rightarrow A$, allora per il teorema di deduzione $C \nvDash A$, dunque $RA \nvDash A$. Dunque, abbiamo deciso se $RA \models A$ oppure $RA \nvDash A$, cioè, RA è decidibile. Assurdo. Pertanto, la Logica del Primo Ordine non è decidibile.*

Mentre l'insieme degli enunciati insoddisfacibili è enumerabile, l'insieme degli enunciati soddisfacibili non è enumerabile, poiché se così fosse ci sarebbe un modo per rendere decidibile una qualunque aritmetica, che è assurdo. enumerabile