

3 Calcoli logici

3.1 Alberi di refutazione

La tecnica degli alberi di refutazione si può estendere agli enunciati dei linguaggi predicativi aggiungendo le seguenti regole:

- Se A è $\exists xB$, si introduce una nuova costante c e alla fine di ogni ramo non chiuso passante per A si appende alla foglia il successore $B[x/c]$, come nello schema

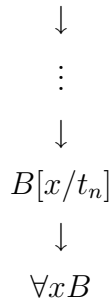
$$\begin{array}{c} [\exists xB] \\ \vdots \\ \downarrow \\ B[x/c] \end{array}$$

- Se A è $\neg\forall xB$, si introduce una nuova costante c e alla fine di ogni ramo non chiuso passante per A si appende alla foglia il successore $\neg B[x/c]$, come nello schema

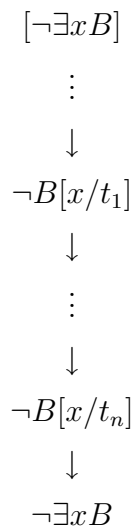
$$\begin{array}{c} [\neg\forall xB] \\ \vdots \\ \downarrow \\ \neg B[x/c] \end{array}$$

- Se A Se A è $\forall xB$, allora alla fine di ogni ramo non chiuso passante per A , per tutti i termini chiusi t_1, \dots, t_n che occorrono in qualche enunciato del ramo, e tali che $B[x/t_i]$ non occorre già nel ramo, si appendono alla foglia $n + 1$ nodi in serie, prima $B[x/t_1], \dots, B[x/t_n]$ e poi ancora $\forall xB$, come nello schema

$$\begin{array}{c} [\forall xB] \\ \vdots \\ \downarrow \\ B[x/t_1] \end{array}$$



- Se A Se A è $\neg\exists x B$, allora alla fine di ogni ramo non chiuso passante per A , per tutti i termini chiusi t_1, \dots, t_n che occorrono in qualche enunciato del ramo, e tali che $B[x/t_i]$ non occorre già nel ramo, si appendono alla foglia $n + 1$ nodi in serie $\neg B[x/t_1], \dots, \neg B[x/t_n]$, e poi ancora $\neg\exists x B$, come nello schema



Se l'albero è inizializzato con un enunciato, tutti i nodi dell'albero sono etichettati con enunciati, di un linguaggio possibilmente arricchito con nuove costanti. Il ruolo dei letterali è ora svolto dagli enunciati atomici e dalle negazioni degli enunciati atomici.

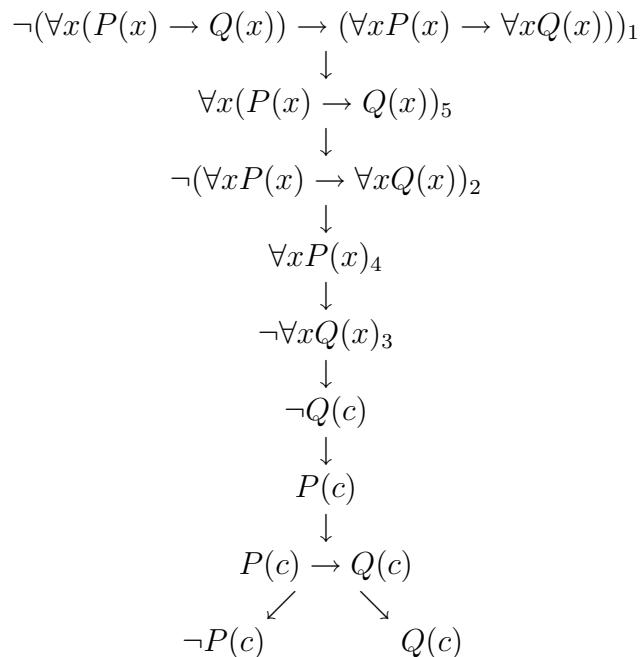
Il senso delle due ultime regole è il seguente; per la regola relativa al quantificatore universale (per la negazione dell'esistenziale valgono le stesse considerazioni) si vorrebbero sostituire a x in B tutti i termini chiusi; ma questi sono in generale infiniti, e neppure ben determinati, per il fatto che

successive applicazioni delle altre regole ad altri nodi possono introdurre nuove costanti; allora si incominciano a sostituire i termini esplicitamente esistenti, ma si riscrive l'enunciato $\forall xB$ in modo che quando eventualmente (se il ramo non si è nel frattempo chiuso) si torna a considerare l'enunciato, se nel frattempo si sono creati nuovi termini chiusi anche i nuovi vengano sostituiti.

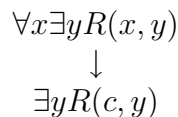
Se in una prima applicazione delle ultime due regole non esistono termini chiusi negli enunciati dell'albero, si introduce una nuova costante c e si sostituisce quella.

Praticamente, non c'è bisogno di riscrivere $\forall xB$, basta non marcarlo come già considerato e ricordarsi di tornare periodicamente a visitarlo. E quando tutti gli altri enunciati siano stati considerati e non ci siano altri termini da sostituire, lo si marca definitivamente per terminare.

Esempio di albero chiuso:



Non vale più la proprietà di terminazione, come mostra l'albero per l'enunciato $\forall x\exists yR(x, y)$



$$\begin{array}{c}
\downarrow \\
R(c, c_1) \\
\downarrow \\
\exists y R(c_1, y) \\
\downarrow \\
R(c_1, c_2) \\
\downarrow \\
\exists y R(c_2, y) \\
\downarrow \\
\vdots
\end{array}$$

e quello per $\forall x R(x, F(x))$

$$\begin{array}{c}
\forall x R(x, F(x)) \\
\downarrow \\
R(c, F(c)) \\
\downarrow \\
\forall x R(x, F(x)) \\
\downarrow \\
R(F(c), F(F(c))) \\
\downarrow \\
\forall x R(x, F(x)) \\
\downarrow \\
R(F(F(c)), F(F(F(c)))) \\
\downarrow \\
\vdots
\end{array}$$

Valgono però le proprietà fondamentali di correttezza e completezza.

Teorema 3.1.1 (Correttezza). *Se l'albero di refutazione con radice A si chiude, allora A è insoddisfacibile.*

Dimostrazione. La dimostrazione si svolge come nel caso proposizionale, dimostrando che se esiste un modello \mathcal{M} di A , allora a ogni stadio esiste almeno un ramo tale che tutti i suoi enunciati sono veri \mathcal{M} . L'unica differenza è che \mathcal{M} non è proprio sempre la stessa, ma può dover essere arricchita, in corrispondenza all'arricchimento del linguaggio con nuove costanti nel corso del processo, con la specifica delle denotazioni delle nuove costanti.

I casi dei connettivi si trattano nel modo già visto; sia ora l'enunciato non ancora considerato sul ramo dato allo stadio n è della forma $\exists xB$, vero nell'interpretazione; data una α e una x -variante α' che soddisfa B , per la nuova costante c introdotta con la regola si arricchisce l'interpretazione con $c^M = \alpha'(x)$ e si ha che anche $B[x/c]$ è vero nell'interpretazione.

Lo stesso per un enunciato della forma $\neg\forall xB$. Per un enunciato della forma $\forall xB$, poiché esso è vero in \mathcal{M} anche ogni $B[x/t_i]$ aggiunto è vero in \mathcal{M} ; lo stesso per enunciati $\neg\exists xB$. \square

Il precedente esempio di albero chiuso verifica che $\forall x(P(x) \rightarrow Q(x)) \models \forall xP(x) \rightarrow \forall xQ(x)$.

Teorema 3.1.2 (Completezza). *Se A è insoddisfacibile, l'albero con radice A si chiude.*

La dimostrazione segue dal

Lemma 3.1.1. *Se l'albero di refutazione con radice A non si chiude, allora per ogni ramo non chiuso, finito e terminato, o infinito, esiste un modello di A .*

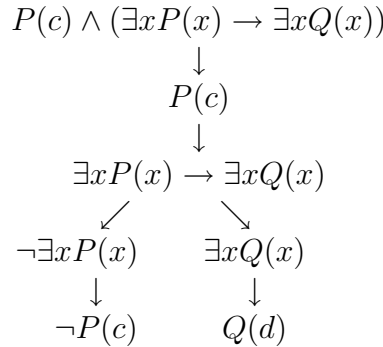
Dimostrazione. In ogni caso si definisce un'interpretazione \mathcal{M} nel seguente modo. L'universo M è l'insieme dei termini chiusi che occorrono in qualche enunciato del ramo (anche sottotermini di termini). L'interpretazione è definita solo per i simboli che effettivamente compaiono in qualche enunciato del ramo. Per ogni costante c , dell'alfabeto originario o introdotta nel corso del processo, si pone $c^M = c$; per ogni simbolo funzionale F a n argomenti e ogni $t_1, \dots, t_n \in M$, si pone $F^M(t_1, \dots, t_n) = F(t_1, \dots, t_n)$ se $F(t_1, \dots, t_n)$ appartiene a M , altrimenti un termine qualunque. Si ha allora che $t^M = t$ per ogni $t \in M$.

Infine per ogni simbolo predicativo P a n argomenti e ogni $t_1, \dots, t_n \in M$ si pone per definizione

$$\langle t_1, \dots, t_n \rangle \in P^M \text{ se e solo se } P(t_1, \dots, t_n) \text{ è un nodo del ramo.}$$

Con questa definizione, si verifica facilmente, per induzione sull'altezza degli enunciati, che per ogni enunciato A che occorre nel ramo si ha $\mathcal{M} \models A$. \square

Esempio L'albero



mostra che l'enunciato $P(c) \wedge (\exists x P(x) \rightarrow \exists x Q(x))$ è soddisfacibile con un modello (dato dal ramo di destra, quello di sinistra è chiuso) costituito da $M = \{c, d\}$, $c^M = c$, $d^M = d$ e $P^M = \{c\}$ e $Q^M = \{d\}$.

Gli alberi di refutazione si possono usare per stabilire la validità logica di formule aperte considerando la loro chiusura universale. Se, data A con la sola variabile libera x , per esempio, si considera $\forall x A$ e per verificarne la verità logica si inizializza un albero con $\neg \forall x A$, la prima regola introduce una nuova costante c e l'enunciato $\neg A[x/c]$. Se invece che da $\forall x A$ si parte da $A[x/c]$, la chiusura dell'albero stabilisce la verità logica di $A[x/c]$, dove però c , essendo nuova e quindi non ristretta da A o da sue sottoformule a denotare uno specifico elemento, svolge il ruolo di elemento arbitrario, come le variabili. Ma se si parte da $A[x/c]$ e l'albero non si chiude, ciò non significa che A è soddisfacibile, perché $A[x/c]$ ha un modello con un particolare valore per c ; significa che $\exists x A$ lo è. Di fatto si potrebbe definire il metodo in modo da permettere anche formule aperte, risulta solo un po' più faticosa la dimostrazione della correttezza e completezza.

3.2 Esercizi

1. Verificare con gli alberi di refutazione tutte le leggi logiche finora incontrate.
2. Utilizzare gli alberi di refutazione per spiegare che se una formula ha la struttura proposizionale di una tautologia allora è logicamente valida.
3. Trovare un controesempio a $\exists x P(x) \wedge \exists x Q(x) \rightarrow \exists x (P(x) \wedge Q(x))$.
4. Trovare un controesempio a $\forall x (P(x) \vee Q(x)) \rightarrow \forall x P(x) \vee \forall x Q(x)$.

5. Trovare un controesempio a $\forall xP(x) \rightarrow \forall xQ(x) \models \forall x(P(x) \rightarrow Q(x))$.
6. Verificare se i seguenti enunciati sono soddisfacibili, e in caso positivo descrivere i loro modelli, e quali sono infiniti:

$$\forall x\exists yR(x, y) \rightarrow \exists y\forall xR(x, y)$$

$$\forall xR(c, x) \rightarrow \forall x\exists yR(x, y)$$

$$\neg\forall xP(x) \wedge \forall x(Q(x) \rightarrow P(x)) \wedge \forall xQ(x)$$

$$\forall xR(c, x) \wedge \forall x\neg R(F(x), c) \wedge \forall xR(x, F(x))$$

7. Dimostrare che se l'enunciato della radice contiene solo simboli relazionali a un posto e nessun simbolo funzionale (linguaggio monadico) l'albero di refutazione è sempre finito.

Suggerimento. Dimostrare prima con le leggi del paragrafo 2.2 che ogni enunciato del genere è logicamente equivalente ad uno in cui non ci sono quantificatori nidificati, cioè nessun quantificatore cade dentro al raggio d'azione di un altro.

3.3 Forme prenesse e forme normali di Skolem

Per mezzo delle equivalenze relative al rapporto tra quantificatori e connettivi, elencate tra le leggi logiche di 2.2, è possibile trasformare una formula in una logicamente equivalente spostando all'esterno i quantificatori, finché si perviene a una formula in forma *prenessa*. Con questo si intende che la formula è costituita da un *prefisso* iniziale, formato da una lista di quantificatori, seguito da una formula in cui non occorrono più quantificatori ed è detta *matrice*.

Lemma 3.3.1. *Ogni formula è logicamente equivalente a una formula in forma prenessa con le stesse variabili libere.*

Dalle leggi logiche sopra elencate si vede che per la trasformazione è necessario in genere ricorrere alla rinomina delle variabili vincolate, e che la forma prenessa di una formula non è unica.

Esempio La seguente successione di formule costituisce una trasformazione in forma prenessa:

$$\begin{aligned}
& \forall xP(x) \rightarrow \forall x\exists yR(x, y) \vee \neg\exists xQ(x) \\
& \forall xP(x) \rightarrow \forall x\exists yR(x, y) \vee \forall x\neg Q(x) \\
& \forall xP(x) \rightarrow \forall x\exists yR(x, y) \vee \forall z\neg Q(z) \\
& \forall xP(x) \rightarrow \forall x\forall z(\exists yR(x, y) \vee \neg Q(z)) \\
& \forall xP(x) \rightarrow \forall x\forall z\exists y(R(x, y) \vee \neg Q(z)) \\
& \forall uP(u) \rightarrow \forall x\forall z\exists y(R(x, y) \vee \neg Q(z)) \\
& \exists u\forall x\forall z\exists y(P(u) \rightarrow R(x, y) \vee \neg Q(z)).
\end{aligned}$$

Una formula si dice *universale* se è in forma prenessa e il suo prefisso è costituito solo da quantificatori universali; un enunciato si dice *esistenziale* se è in forma prenessa e il suo prefisso è costituito solo da quantificatori esistenziali. Una formula senza quantificatori si può considerare in forma prenessa e sia universale sia esistenziale.

Una formula si dice in *forma normale di Skolem* se la formula è universale e la sua matrice è in forma normale congiuntiva (dove ora con “letterale” si intende una formula atomica o la negazione di una formula atomica).

Due formule A e B si dicono *equisoddisfacibili* se A è soddisfacibile se e solo se B è soddisfacibile (non necessariamente con lo stesso modello).

Lemma 3.3.2. *Ogni formula A è equisoddisfacibile ad una formula in forma normale di Skolem con le stesse variabili libere.*

Dimostrazione. Data una formula A , la si mette in forma prenessa equivalente. Per eliminare dal prefisso i quantificatori esistenziali si procede in questo modo: se la formula è della forma $\forall x_1 \dots \forall x_n \exists y B$, si introduce un nuovo simbolo funzionale F a n argomenti, e la formula è equisoddisfacibile con $\forall x_1 \dots \forall x_n B[y/F(x_1, \dots, x_n)]$. Se $\forall x_1 \dots \forall x_n \exists y B$ è valida in \mathcal{M} , si definisce F^M ponendo, per ogni $a_1, \dots, a_n \in M$, $F^M(a_1, \dots, a_n) = \alpha'(y)$ dove, se α con $\alpha(x_i) = a_i$ è tale che $\mathcal{M}, \alpha \models \exists y B$, α' è una y -variante che soddisfa B . Allora poiché α' soddisfa B e $(F(x_1, \dots, x_n))^\alpha = \alpha'(y)$, per il Lemma 2.2.2 α soddisfa $B[y/F(x_1, \dots, x_n)]$ e $B[y/F(x_1, \dots, x_n)]$ è valida nella struttura $\mathcal{M}' = \langle M, \dots, F^M \rangle$ che è \mathcal{M} arricchita con l'interpretazione F^M del simbolo F .

Il viceversa segue dalla legge logica $B[y/t] \rightarrow \exists y B$.

Se la forma prenessa è della forma $\exists y B$, si introduce una nuova costante c e si considera $B[y/c]$.

Eliminato un quantificatore esistenziale, si itera. \square

Esempio L'enunciato in forma normale di Skolem che è equisoddisfacibile a $\forall x \exists y R(x, y)$ di pag. 78 è lo stesso $\forall x R(x, F(x))$, pag. 79.

Quando si è trovata la forma universale equisoddisfacibile, la matrice si può sempre trasformare in forma normale congiuntiva. Ma dal punto di vista pratico non sempre conviene passare attraverso la forma prenessa; se la formula è una congiunzione $A_1 \wedge \dots \wedge A_n$, conviene trasformare prima ogni A_i in forma normale di Skolem, e dopo farne la congiunzione. Che il risultato sia equisoddisfacibile segue dal fatto che se $A_1 \wedge \dots \wedge A_n$ ha un modello \mathcal{M} , ciascuna delle forme di Skolem delle A_i è valida in un arricchimento, relativo a simboli funzionali diversi, della stessa \mathcal{M} .

Esempio All'enunciato $\exists x \forall y P(x, y) \wedge \neg \exists z \forall x P(x, z)$ si può associare la forma $\forall y P(c, y) \wedge \forall z \neg P(F(z), z)$, equivalente a $\forall y \forall z (P(c, y) \wedge \neg P(F(z), z))$ invece di passare alla forma prenessa $\exists x \forall y \forall z \exists u (P(x, y) \wedge \neg P(u, z))$ e quindi a $\forall y \forall z (P(c, y) \wedge \neg P(G(y, z), z))$ dove il simbolo funzionale G è a due argomenti.

3.4 Esercizi

1. Trasformare in forma prenessa e in forma normale di Skolem le seguenti formule, anche in più di un modo:

$$\begin{aligned} & \exists x P(x) \vee \neg \exists x P(x) \\ & \forall x P(x) \wedge \exists x \neg P(x) \\ & \exists x P(x) \rightarrow \exists x Q(x) \\ & \forall x \exists y (\forall z R(z, y) \vee \neg P(x, y)) \wedge \forall x \exists y \forall z \exists u (P(x, z) \rightarrow R(y, u)) \\ & \forall x \exists y R(x, y) \rightarrow (\exists y \forall x R(x, y) \vee \forall y \exists x \neg R(x, y)) \\ & \neg (\forall x \exists y R(x, y) \rightarrow (\exists y \forall x R(x, y) \vee \forall y \exists x \neg R(x, y))) \end{aligned}$$

2. Dare un esempio di una formula B per cui $\exists y B$ e $B[y/c]$ non sono logicamente equivalenti (trovare un modello per $\exists y B$ e $\neg B[y/c]$).

3.5 Interpretazioni di Skolem-Herbrand

Le interpretazioni definite nella dimostrazione del lemma 3.1.1, per ogni ramo non chiuso, sono particolari: l'universo è un insieme di termini chiusi; le funzioni sono definite in modo che risulti che ogni termine denota se stesso, $t^M = t$ e le relazioni sono definite scegliendo un insieme di enunciati

atomici come veri. Interpretazioni con queste caratteristiche si chiamano interpretazioni *di Skolem-Herbrand*.

Avvertenza Prima di pensare a chissà quali diavolerie, si consideri che le interpretazioni in cui i termini chiusi denotano se stessi sono le più naturali. Nelle prime esperienze con i numeri, le cifre $1, 2, \dots$ non denotano numeri, ma *sono* i numeri, ovvero questi si identificano con le cifre. Solo a livelli sofisticati si introduce una notazione diversa per i numeri denotati dalle cifre o dai numerali; ad esempio l'insieme vuoto \emptyset è il numero zero secondo la definizione insiemistica, e 0 la cifra del linguaggio aritmetico che lo denota, oppure in un'interpretazione \mathcal{M} dell'aritmetica l'elemento $0^{\mathcal{M}}$ di M è lo zero della struttura, e 0 sempre la costante del linguaggio che lo denota.

La differenza è che nella trattazione intuitiva tendiamo a dire che ad esempio $1 + 1$ denota 2 , mentre in un'interpretazione di Skolem-Herbrand $1 + 1$ denota se stesso, e *si riduce* alla forma normale 2 , ovvero, $1 + 1$ e 2 stanno nella stessa classe di equivalenza rispetto a $=$.

Se il linguaggio ha l'uguaglianza, le interpretazioni che si ottengono dagli alberi di refutazione, se si inseriscono gli assiomi dell'uguaglianza, sono interpretazioni quasi-normali.

Dal lemma 3.1.1 si può perciò dedurre

Corollario 3.5.1. *Se un enunciato A è soddisfacibile, A ha un modello di Skolem-Herbrand*

osservando che se A è soddisfacibile, l'albero con radice A non si chiude, e A ha modelli costruiti come nella dimostrazione del lemma.

Un'interpretazione di Skolem-Herbrand si può vedere come interpretazione proposizionale degli enunciati privi di quantificatori: la scelta degli enunciati atomici veri che individua l'interpretazione di Skolem-Herbrand, e che si chiama anche *base* di Herbrand (che nel caso dell'albero di refutazione sono quelli nei nodi di un ramo non chiuso, ma in altri contesti potrebbero essere scelti in modo diverso) si può pensare come l'attribuzione del valore 1 ad alcuni enunciati atomici (agli altri il valore 0); gli enunciati privi di quantificatori sono ottenuti da quelli atomici con i connettivi e il loro essere veri o falsi nell'interpretazione di Skolem-Herbrand coincide con il loro avere 1 o 0 come valore calcolato con le funzioni di verità.

Un insieme S di enunciati privi di quantificatori si dice *soddisfacibile in senso proposizionale* se esiste un'assegnazione v di valori $0, 1$ agli enunciati

atomici occorrenti come sottoenunciati degli enunciati di S tale che $v^*(A) = 1$ per ogni $A \in S$.

Per un enunciato universale $\forall x_1 \dots \forall x_n A$, sia S_A l'insieme degli enunciati privi di quantificatori che si ottengono sostituendo i termini chiusi in tutti i modi possibili nella matrice A .

Teorema 3.5.1 (Teorema di Skolem-Herbrand). *Un enunciato universale $\forall x_1 \dots \forall x_n A$ è insoddisfacibile se e solo se esiste un sottinsieme finito di S_A che è insoddisfacibile in senso proposizionale.*

Dimostrazione. Se $\forall x_1 \dots \forall x_n A$ è insoddisfacibile, l'albero che ha la radice $\forall x_1 \dots \forall x_n A$ si chiude. Nello sviluppo dell'albero si applicherà prima che si chiuda un numero finito di volte la regola per il quantificatore universale, che introduce nell'albero enunciati $A[x_1/t_1^i, \dots, x_n/t_n^i]$ a cui poi si applicano solo regole relative a connettivi.

Se si imposta un albero con un ramo iniziale in cui tutti questi enunciati $A[x_1/t_1^i, \dots, x_n/t_n^i]$ sono introdotti in serie, nello sviluppo successivo di quest'albero si applicano le stesse mosse che si sono applicate nel precedente, e che portano alla sua chiusura.

Viceversa se c'è un insieme finito di enunciati $A[x_1/t_1^i, \dots, x_n/t_n^i] \in S_A$ tale che l'albero che ha come radice la loro congiunzione si chiude, perché l'insieme è insoddisfacibile, allora per l'albero con radice $\forall x_1 \dots \forall x_n A$ nel suo sviluppo si può iterare la regola del quantificatore universale, prima di quelle relative ai connettivi (i nuovi termini vengono solo dalla sostituzione di termini in altri termini, perché non si applica né la regola del quantificatore esistenziale né quella della negazione dell'universale), finché prima o poi si introdurranno gli $A[x_1/t_1^i, \dots, x_n/t_n^i]$ che con il loro sviluppo seguente portano alla chiusura dell'albero. \square

Gli elementi di S_A si chiamano anche *esemplificazioni di base* (ingl. *ground instances*) di A , o della chiusura universale $\forall A$ di A .

Esempio Se $\forall x A$ è $\forall x (R(c, x) \wedge \neg R(x, F(x)))$, allora il sottinsieme di S_A insoddisfacibile in senso proposizionale è

$$\{R(c, c) \wedge \neg R(c, F(c)), R(c, F(c)) \wedge \neg R(F(c), F(F(c)))\}$$

che si ottiene con due sostituzioni ed ha la struttura proposizionale $\{P \wedge \neg Q, Q \wedge R\}$.

Il lemma 3.3.2 e il teorema di Skolem-Herbrand permettono una riduzione (del problema della validità) della logica predicativa alla logica proposizionale, modulo una ricerca nell'insieme infinito dei termini chiusi.

3.6 Esercizi

1. Applicando la forma normale di Skolem e il teorema di Skolem-Herbrand, mostrare che la congiunzione dei due enunciati

$$\forall x \forall y \forall z (R(x, y) \wedge R(y, z) \rightarrow R(x, z)) \text{ e } \exists y (R(c, y) \wedge R(y, c) \wedge \neg R(y, y))$$

è una contraddizione.

2. Scrivere in forma normale di Skolem l'enunciato

$$\exists y \forall x (R(y, x) \wedge \exists z (R(x, z) \rightarrow \neg R(x, x)))$$

e verificare con gli alberi di refutazione che è insoddisfacibile trovando le sostituzioni di termini chiusi che forniscono l'insieme proposizionalmente insoddisfacibile di esemplificazioni di base.

3.7 Unificazione

Per ottenere un insieme finito di esemplificazioni di base che sia proposizionalmente insoddisfacibile occorre talvolta fare diverse sostituzioni, e solo l'effetto combinato di più di una fornisce l'insieme cercato.

Esempio Da $\forall x \exists y \forall z (P(c, x) \wedge \neg P(z, y))$ si ottiene la forma normale di Skolem $\forall x \forall z (P(c, x) \wedge \neg P(z, F(x)))$. La prima sostituzione di c a x e c a z fornisce

$$P(c, c) \wedge \neg P(c, F(c))$$

una seconda di c a x e $F(c)$ a z fornisce

$$P(c, c) \wedge \neg P(F(c), F(c))$$

e una terza di $F(c)$ a x e c a z

$$P(c, F(c)) \wedge \neg P(c, F(F(c)))$$

da cui si vede la contraddizione tra $\neg P(c, F(c))$ della prima esemplificazione e $P(c, F(c))$ della terza.

Se invece eseguiamo le seguenti trasformazioni equivalenti della forma normale di Skolem:

$$\begin{aligned} & \forall x \forall z (P(c, x) \wedge \neg P(z, F(x))) \\ & \forall x \forall z P(c, x) \wedge \forall x \forall z \neg P(z, F(x)) \\ & \forall x P(c, x) \wedge \forall y \forall z \neg P(z, F(y)) \\ & \forall x \forall y \forall z (P(c, x) \wedge \neg P(z, F(y))) \end{aligned}$$

arriviamo alla matrice

$$P(c, x) \wedge \neg P(z, F(y))$$

dove è sufficiente l'unica sostituzione di $F(c)$ a x , c a y e c a z .

Con i passaggi eseguiti nell'esempio, è sempre possibile avere una matrice, che d'ora in poi chiameremo insieme di clausole, che sia *a variabili disgiunte*, cioè tale che nessuna variabile occorra in due clausole.

Per non dover applicare sistematicamente tutte le sostituzioni possibili, occorre individuare quelle utili, che, nella prospettiva di applicare il metodo di risoluzione per stabilire l'insoddisfacibilità, sono quelle che fanno comparire letterali complementari in clausole diverse. Si tratta di prevedere quali

sostituzioni, se ne esistono, renderanno uguali due formule atomiche. Questa previsione e l'individuazione della sostituzione è possibile con un algoritmo, che costruisce la sostituzione per approssimazioni successive, componendo successive sostituzioni, e quindi lavora con termini che contengono variabili.

Esempio Per la forma normale congiuntiva

$$(P(x) \vee R(x)) \wedge \neg P(F(y)) \wedge \neg R(F(G(c)))$$

la sostituzione che produce un enunciato insoddisfacibile è quella che a x sostituisce $F(G(c))$ e a y sostituisce $G(c)$. Ma questa la si può riconoscere considerando prima la sostituzione di $F(y)$ a x e quindi la sostituzione di $G(c)$ a y .

Una sostituzione simultanea del termine t_1 alla variabile x_1, \dots , e di t_n a x_n si indicherà con la notazione $\{x_1/t_1, \dots, x_n/t_n\}$ e sostituzioni del genere con lettere greche. L'applicazione di una sostituzione σ a un'espressione W si indicherà con $W\sigma$; la composizione di due sostituzioni σ e θ con $\sigma \circ \theta$, che è la sostituzione tale che $W(\sigma \circ \theta) = (W\sigma)\theta$.

Un insieme finito $W = \{E_1, \dots, E_n\}$ di espressioni (che per noi saranno sempre formule atomiche) si dice *unificabile* se esiste una sostituzione σ tale che $W\sigma = \{E_1\sigma, \dots, E_n\sigma\}$ ha un solo elemento, cioè σ rende uguali le espressioni di W a cui si applica. σ si chiama *unificatore* di W .

Algoritmo di unificazione. Dato un insieme W di espressioni, si supponga di avere generato allo stadio k una sostituzione σ_k e un insieme di espressioni W_k tali che $W_k = W\sigma_k$. Allo stadio iniziale si può supporre la sostituzione identica.

Se W_k ha un solo elemento, W è unificabile e σ_k un unificatore; altrimenti, si esaminino le espressioni di W_k da sinistra a destra finché si trova il primo punto di disaccordo, dove non tutte le liste sono uguali, e si definisca l'*insieme di disaccordo* D_k come insieme dei termini, uno per ciascuna espressione di W_k , che iniziano nel punto di disaccordo. Se D_k contiene due termini, uno dei quali è una variabile v_k e l'altro è un termine t_k che non contiene v_k , si ponga $\sigma_{k+1} = \sigma_k \circ \{v_k/t_k\}$ e $W_{k+1} = W_k\{v_k/t_k\}$ (da cui segue $W_{k+1} = W\sigma_{k+1}$).

Se in D_k non esiste una tale coppia di termini, si esce dicendo che W non è unificabile. Se in D_k ci sono diverse variabili v, \dots, u che non occorre in un termine t_k si può anche porre $\sigma_{k+1} = \{v/t_k, \dots, u/t_k\}$.

Quindi si itera.

L'algoritmo termina sempre perché a ogni stadio in cui si itera una variabile scompare, essendo sostituita ovunque da termini che non la contengono, e le variabili iniziali sono finite.

Esempio Se $W = \{P(x, F(x), v), P(y, z, F(y)), P(u, F(G(w)), F(G(w)))\}$,
o per maggior chiarezza

$$\begin{aligned} &P(x, F(x), v) \\ &P(y, z, F(y)) \\ &P(u, F(G(w)), F(G(w))), \end{aligned}$$

posto $W_0 = W$ si ha che $D_0 = \{x, y, u\}$ e si può porre $\sigma_1 = \{x/u, y/u\}$ ottenendo come W_1

$$\begin{aligned} &P(u, F(u), v) \\ &P(u, z, F(u)) \\ &P(u, F(G(w)), F(G(w))). \end{aligned}$$

D_1 è ora $\{F(u), z, F(G(w))\}$ e si può porre $\sigma_2 = \sigma_1 \circ \{z/F(u)\}$ ottenendo per W_2

$$\begin{aligned} &P(u, F(u), v) \\ &P(u, F(u), F(u)) \\ &P(u, F(G(w)), F(G(w))). \end{aligned}$$

$D_2 = \{u, G(w)\}$ e $\sigma_3 = \sigma_2 \circ \{u/G(w)\}$, e W_3 è

$$\begin{aligned} &P(G(w), F(G(w)), v) \\ &P(G(w), F(G(w)), F(G(w))) \\ &P(G(w), F(G(w)), F(G(w))). \end{aligned}$$

$D_3 = \{v, F(G(w))\}$, $\sigma_4 = \sigma_3 \circ \{v/F(G(w))\}$ e si ha infine

$$W_4 = \{P(G(w), F(G(w)), F(G(w)))\}.$$

Il risultato σ_4 delle successive composizioni si scrive

$$\sigma_4 = \{x/G(w), y/G(w), z/F(G(w)), u/G(w), v/F(G(w))\}.$$

Ovviamente se l'algoritmo fornisce un unificatore, W è unificabile. Viceversa, se W è unificabile, l'algoritmo fornisce un unificatore, anzi uno con proprietà aggiuntiva.

Nell'esempio, dato l'unificatore fornito dall'algoritmo, è subito visto che anche $\{x/G(s), y/G(s), z/F(G(s)), u/G(s), v/F(G(s))\}$, dove s è una nuova variabile, o $\{x/G(c), y/G(c), z/F(G(c)), u/G(c), v/F(G(c))\}$, o più in generale

$$\{x/G(t), y/G(t), z/F(G(t)), u/G(t), v/F(G(t))\}$$

per un termine qualunque t , cioè $\sigma_4 \circ \{w/t\}$, sono unificatori. Ma vale anche il viceversa.

Un unificatore σ di un insieme W si dice *generale* se per ogni altro unificatore θ di W esiste una sostituzione λ tale che $\theta = \sigma \circ \lambda$.

Un unificatore generale non è mai unico. In particolare si può eseguire una rinomina, sostituzione biunivoca delle variabili che compaiono nei termini dell'unificatore con nuove variabili, e si ottiene ancora un unificatore generale. Due unificatori generali differiscono solo per una rinomina.

L'unificatore fornito dall'algoritmo di unificazione è un unificatore generale.

Esempio $W = \{P(x, F(x), v), P(c, z, F(y)), P(u, F(G(w)), F(G(w)))\}$ non è unificabile. I primi passi analoghi al precedente esempio portano a

$$\begin{aligned} &P(c, F(c), v) \\ &P(c, F(c), F(c)) \\ &P(c, F(G(w)), F(G(w))) \end{aligned}$$

con le sostituzioni $\{x/c, u/c\} \circ \{z/F(c)\}$, ma ora $D_2 = \{c, G(w)\}$ e non sono soddisfatte le condizioni per proseguire.

Esempio $W = \{P(x, z, G(x)), P(y, F(x), y)\}$ non è unificabile perché al terzo passo l'insieme di disaccordo è $\{G(x), x\}$ oppure $\{G(y), y\}$; l'esito negativo non dipende dal fatto che le due espressioni non sono a variabili disgiunte; lo stesso succede se la seconda è $P(y, F(u), y)$.

3.8 Esercizi

Applicare l'algoritmo di unificazione per decidere se i seguenti insiemi di espressioni sono unificabili (e trovare un unificatore generale) oppure no:

$$\begin{aligned} &\{P(x, F(y), z), P(G(c), F(w), u), P(v, F(d), e)\} \\ &\{P(F(x, y), w), P(F(G(v), c), H(v)), P(F(G(v), c), H(d))\} \\ &\{P(x, F(x)), P(y, y)\} \\ &\{P(H(y), c, z), P(H(F(w)), c, w), P(H(F(c)), c, d)\} \\ &\{P(H(y), c, z), P(H(F(w)), c, w), P(H(F(c)), c, u)\} \\ &\{P(F(x), y), P(y, y), P(y, G(z))\} \\ &\{P(y, z, F(y)), P(x, F(x), v), P(u, F(G(w)), F(G(w)))\} \\ &\{P(y, z, F(y)), P(x, F(x), v), P(u, F(G(w)), F(G(w)))\} \\ &\{P(c, x, F(G(y))), P(z, F(y), F(y)), P(u, F(y), v)\}. \end{aligned}$$

3.9 Risoluzione con variabili

Dato un insieme di clausole con variabili, che si può sempre pensare sia la matrice di una forma normale di Skolem, si dirà che l'insieme di clausole è insoddisfacibile se è insoddisfacibile l'enunciato che ha quella forma normale di Skolem. L'insoddisfacibilità si può stabilire derivando la clausola vuota con una generalizzazione della regola di risoluzione.

Due letterali si chiamano *simili* se iniziano con lo stesso simbolo predicativo o iniziano entrambi con la negazione dello stesso simbolo predicativo; si chiamano *quasi-complementari* se uno inizia con un simbolo predicativo e l'altro con la negazione dello stesso simbolo, e si indicano con $L(t_1, \dots, t_n)$ e $L^c(s_1, \dots, s_n)$.

Definizione 3.9.1. *Date due clausole a variabili disgiunte che contengono l'una letterali quasi-complementari di letterali dell'altra,*

$$C_1 \cup \{L(t_{11}, \dots, t_{1n}), \dots, L(t_{h1}, \dots, t_{hn})\}$$

e

$$C_2 \cup \{L^c(s_{11}, \dots, s_{1n}), \dots, L^c(s_{k1}, \dots, s_{kn})\}$$

dove $\{L(t_{11}, \dots, t_{1n}), \dots, L(t_{h1}, \dots, t_{hn})\}$ è un insieme di letterali simili in una delle due clausole e $\{L^c(s_{11}, \dots, s_{1n}), \dots, L^c(s_{k1}, \dots, s_{kn})\}$ è un insieme di letterali quasi-complementari dei primi nella seconda, allora se σ è un unificatore generale di

$$\{L(t_{11}, \dots, t_{1n}), \dots, L(t_{h1}, \dots, t_{hn}), L(s_{11}, \dots, s_{1n}), \dots, L(s_{k1}, \dots, s_{kn})\}$$

la risolvente delle due clausole è $(C_1\sigma \cup C_2\sigma)\lambda$, dove λ è una rinomina tale che la clausola risolvente abbia variabili disgiunte dalle genitrici (e, quando inserita in una derivazione da un insieme S , da tutte le clausole di S e da quelle già ottenute per risoluzione).

Esempio La necessità, e non solo l'opportunità, di avere variabili disgiunte è illustrata dall'insieme di clausole $\{P(x), \neg P(F(x))\}$ da cui non si potrebbe derivare la clausola vuota con la risoluzione perché $P(x)$ e $P(F(x))$ non sono unificabili; invece tale insieme corrisponde all'enunciato $\forall x \exists y (P(x) \wedge \neg P(y))$ che è insoddisfacibile.

D'altra parte l'insieme $\{P(x), \neg P(F(y))\}$ che si può sostituire a quello dato è la matrice di $\forall x \forall y (P(x) \wedge \neg P(F(y)))$ che si può pensare proviene da $\forall x P(x) \wedge \forall y \exists z \neg P(z) \equiv \forall x P(x) \wedge \exists z \neg P(z) \equiv \forall x P(x) \wedge \neg \forall z P(z)$.

Esempio La necessità che anche le nuove clausole risolventi siano a variabili disgiunte da quelle già disponibili è illustrata dall'esempio dell'insieme di clausole $\{P(x) \vee P(F(x)), \neg P(y) \vee Q(y), \neg Q(z)\}$, associate all'enunciato insoddisfacibile $\forall x \exists y (P(x) \vee P(y)) \wedge \forall x (P(x) \rightarrow Q(x)) \wedge \neg \exists z Q(z)$.

Supponiamo di non eseguire le rinomine nelle clausole risolventi. Dalla prima e seconda, con l'unificatore $\{x/y\}$, si ottiene la risolvente $P(F(y)) \vee Q(y)$. Se ora si risolvesse questa con la terza rispetto al letterale $Q(y)$ con l'unificatore $\{z/y\}$ si avrebbe $P(F(y))$ che non potrebbe essere risolto con la seconda per la non unificabilità di $P(y)$ e $P(F(y))$. Se si usasse invece l'unificatore $\{y/z\}$, si otterrebbe $P(F(z))$ che risolto con la seconda, con l'unificatore $\{y/F(z)\}$ darebbe $Q(F(z))$ e si sarebbe di nuovo bloccati.

Se si partisse risolvendo la seconda e la terza, con l'unificatore $\{y/z\}$ si otterrebbe $\neg P(z)$ che con la prima e l'unificatore $\{x/z\}$ darebbe $P(F(z))$; questa con la seconda e l'unificatore $\{y/F(z)\}$ darebbe $Q(F(z))$ non unificabile con la terza clausola.

Se invece in questa derivazione, da $\neg P(z)$ si risolvesse con la prima con l'unificatore $\{z/x\}$ si otterrebbe $P(F(x))$, e questa con la seconda e l'unificatore $\{y/F(x)\}$ darebbe $Q(F(x))$ risolvibile infine con la terza. Oppure $P(F(x))$ si risolve con la clausola precedentemente ottenuta $\neg P(z)$.

Ma quando si voglia meccanizzare il procedimento, l'algoritmo di unificazione è fissato, e in esso un ordine in cui considerare le variabili e scegliere ad esempio tra $\{x/z\}$ e $\{z/x\}$, non si può modificarlo secondo astuzia o convenienza. Alcune euristiche si possono inserire nel programma, ad esempio tra due unificatori $\{x/z\}$ e $\{z/x\}$ preferire quello in cui la variabile da sostituire ha meno occorrenze, ma una volta fissate, queste vengono applicate. Programmare l'applicazione di tutti i possibili unificatori peraltro non sembra la soluzione più ragionevole.

Se ora torniamo alla prima risoluzione ed eseguiamo la rinomina $\{y/v\}$, possiamo risolvere la clausola ottenuta con la terza (usando indifferentemente sia $\{v/z\}$ sia $\{z/v\}$) e procedere fino alla clausola vuota.

Schematicamente, se numeriamo le tre clausole

- 1 $P(x) \vee P(F(x))$
- 2 $\neg P(y) \vee Q(y)$
- 3 $\neg Q(z)$

possiamo rappresentare la derivazione con

$$\begin{array}{r}
\begin{array}{r}
1 \quad 2 \\
\downarrow \swarrow \{x/y\} \\
P(F(y)) \vee Q(y) \\
P(F(v)) \vee Q(v) \quad 3 \\
\downarrow \swarrow \{z/v\} \\
P(F(v)) \\
P(F(u)) \quad 2 \\
\downarrow \swarrow \{y/F(u)\} \\
Q(F(u)) \\
Q(F(w)) \quad 3 \\
\downarrow \swarrow \{z/F(w)\} \\
\Box
\end{array}
\end{array}$$

La condizione di avere sempre tutte le clausole a variabili disgiunte d'altra parte provoca altre complicazioni, come quella di dover unificare più letterali contemporaneamente. Se si considera $\{P(x) \vee P(y), \neg P(z) \vee \neg P(u)\}$, che con una sola risoluzione dà \Box , non è possibile refutarlo risolvendo un letterale alla volta; infatti risolvendo ad esempio rispetto a $P(x)$ e $\neg P(z)$ la risolvente non sarebbe $P(y) \vee \neg P(u)$, che risolta con la prima, con $\{x/u\}$, darebbe $P(y)$ che poi sarebbe eliminato con due risoluzioni con la seconda, ma sarebbe $P(x_1) \vee \neg P(x_2)$, ancora con due letterali; solo unificando i due letterali positivi con uno negativo, o con entrambi i negativi, si arriva alla clausola vuota; da qui la formulazione della regola.

Tuttavia nell'esecuzione manuale si può talvolta evitare di applicare alla risolvente la rinomina, quando sia certo che i letterali che vi compaiono non dovranno o potranno più essere risolti con letterali di clausole già presenti a variabili non disgiunte.

Si potrebbe evitare la rinomina supponendo di avere a disposizione tante copie diverse delle clausole iniziali, cosa che è sempre possibile perché $\forall xC$ è equivalente a $\forall xC \wedge \forall yC[x/y]$, copie che si potrebbero produrre al momento del bisogno; non si risolverebbe tuttavia il problema dell'eventuale risoluzione con clausole precedentemente ottenute.

Valgono le proprietà di correttezza e completezza per la risoluzione con variabili: un insieme di clausole è insoddisfacibile se e solo se da esso è derivabile la clausola vuota.

Per la dimostrazione della completezza, un ruolo essenziale gioca il seguente Lemma[subsection]

Lemma 3.9.1 (Lifting). *Se $C_1 \cup C_2$ è la risolvente proposizionale di due clausole chiuse $C_1 \cup \{L\}$ e $C_2 \cup \{L^c\}$ provenienti, con due sostituzioni θ_1 e θ_2 , dalle clausole con variabili disgiunte*

$$C'_1 \cup \{L(t_{11}, \dots, t_{1n}), \dots, L(t_{h1}, \dots, t_{hn})\}$$

e

$$C'_2 \cup \{L^c(s_{11}, \dots, s_{1n}), \dots, L^c(s_{k1}, \dots, s_{kn})\}$$

dove

$$\{L(t_{11}, \dots, t_{1n}), \dots, L(t_{h1}, \dots, t_{hn})\}\theta_1 = \{L\}$$

e

$$\{L^c(s_{11}, \dots, s_{1n}), \dots, L^c(s_{k1}, \dots, s_{kn})\}\theta_2 = \{L^c\},$$

allora se $C'_1\sigma \cup C'_2\sigma$ è la risolvente di queste due clausole, esiste una sostituzione λ tale che $(C'_1\sigma \cup C'_2\sigma)\lambda = C_1 \cup C_2$.

Dimostrazione. Se D'_1 e D'_2 sono le due clausole con variabili a cui sono applicate θ_1 e θ_2 , l'enunciato del *lifting* è riassunto dal seguente diagramma:

$$\begin{array}{ccc} D'_1, D'_2 & \xrightarrow{\sigma} & C'_1\sigma \cup C'_2\sigma \\ \downarrow \theta_1 \theta_2 & & \downarrow \lambda \\ C_1 \cup \{L\}, C_2 \cup \{L^c\} & \longrightarrow & C_1 \cup C_2 \end{array}$$

La dimostrazione si basa sulla semplice osservazione che, se σ è un unificatore *generale* di

$$\{L(t_{11}, \dots, t_{1n}), \dots, L(t_{h1}, \dots, t_{hn}), L(s_{11}, \dots, s_{1n}), \dots, L(s_{k1}, \dots, s_{kn})\}$$

allora dopo aver considerato che θ , l'unione di θ_1 e θ_2 che agiscono su variabili disgiunte, è anch'essa un unificatore dello stesso insieme, deve esistere λ tale che $\theta = \sigma \circ \lambda$. \square

Grazie a questo lemma, una refutazione svolta sulle esemplificazioni di base di un insieme di clausole con variabili (di un enunciato insoddisfacibile),

che esiste per il teorema di Skolem-Herbrand e la completezza della risoluzione proposizionale, si trasforma in una refutazione di queste per mezzo della regola di risoluzione con variabili, secondo lo schema

$$\begin{array}{ccccc}
 D_1, D_2 & \dashrightarrow & D_3, D_4 & \dashrightarrow & D_5 \cdots \cdots \rightarrow \\
 \downarrow & & \lambda_1 \downarrow & & \lambda_2 \downarrow \\
 C_1, C_2 & \longrightarrow & C_3, C_4 & \longrightarrow & C_5 \cdots \cdots \rightarrow
 \end{array}$$

Quando si arriva a $C\lambda = \square$ anche C deve essere la clausola vuota.

Valgono i raffinamenti analoghi a quelli della risoluzione proposizionale, ad esempio la risoluzione lineare ordinata e, per le clausole di Horn, la risoluzione a input.

In pratica, per trovare una refutazione di un insieme S di clausole, conviene cercare prima una derivazione proposizionale della clausola vuota dalle clausole ottenute in questo modo: dalle formule atomiche occorrenti nelle clausole si cancellano i termini, lasciando solo il simbolo predicativo che è come una lettera proposizionale. Trovata una derivazione proposizionale si vede se la stessa struttura può essere mantenuta ripristinando le formule originarie e accompagnando le risoluzioni con le necessarie unificazioni.

Esempio L'insieme $S = \{R(c, x) \vee \neg P(y), \neg R(x, F(x)), P(F(x)) \vee R(x, y)\}$ è la matrice dell'enunciato $\forall x \exists z \forall y ((P(y) \rightarrow R(c, x)) \wedge (\neg P(z) \rightarrow R(x, y)) \wedge \neg R(x, z))$.

Si consideri l'insieme associato di clausole proposizionali $\{R \vee \neg P, \neg R, P \vee R\}$; esso ammette la refutazione

$$\begin{array}{ccc}
 R \vee \neg P & & \neg R \\
 \swarrow \quad \searrow & & \\
 \neg P & & P \vee R \\
 \swarrow \quad \searrow & & \\
 R & & \neg R \\
 \swarrow \quad \searrow & & \\
 & & \square
 \end{array}$$

che è una guida per la seguente refutazione di S , dopo che si sono riscritte le clausole come $\{R(c, x) \vee \neg P(y), \neg R(z, F(z)), P(F(v)) \vee R(v, w)\}$, a variabili disgiunte:

$$\begin{array}{ccc}
R(c, x) \vee \neg P(y) & & \neg R(z, F(z)) \\
\searrow \swarrow \{z/c, x/F(c)\} & & \\
\neg P(y) & & P(F(v)) \vee R(v, w) \\
\searrow \swarrow \{y/F(v)\} & & \\
R(v, w) & & \neg R(z, F(z)) \\
\searrow \swarrow \{y/z, w/F(z)\} & & \\
& & \square
\end{array}$$

Si noti che se esiste una derivazione per risoluzione con variabili, come l'ultima, allora cancellando tutti i termini e le sostituzioni resta la derivazione proposizionale precedente. L'esistenza di questa è dunque condizione necessaria per l'esistenza di quella con variabili.

Non è vero il viceversa, la condizione non è sufficiente; se si considera il nuovo insieme di clausole $\{R(c, x) \vee \neg P(y), \neg R(F(z), F(z)), P(F(v)) \vee R(v, w)\}$, ad esso è associato lo stesso insieme di clausole proposizionali, con le stesse refutazioni. Ma non è possibile eseguire neanche la prima risoluzione, perché $R(c, x)$ e $R(F(z), F(z))$ non sono unificabili.

Anche alle altre possibili refutazioni dell'insieme di clausole proposizionali non è possibile far corrispondere una refutazione delle clausole con variabili, perché prima o poi si arriva a clausole non unificabili. Ad esempio a

$$\begin{array}{ccc}
P \vee R & & \neg R \\
\searrow \swarrow & & \\
P & & R \vee \neg P \\
\searrow \swarrow & & \\
R & & \neg R \\
\searrow \swarrow & & \\
& & \square
\end{array}$$

corrisponde

$$\begin{array}{ccc}
P(F(v)) \vee R(v, w) & & \neg R(F(z), F(z)) \\
\searrow \swarrow \{v/F(z), w/F(z)\} & & \\
P(F(F(z))) & & R(c, x) \vee \neg P(y) \\
\searrow \swarrow \{y/F(F(z))\} & & \\
R(c, x) & &
\end{array}$$

che si interrompe qui perché $R(c, x)$ non è unificabile con $R(F(z), F(z))$.

Alla stessa conclusione si arriva esplorando le altre possibilità (esercizio). In effetti il nuovo insieme di clausole è la matrice dell'enunciato soddisfacibile $\forall x \exists z \forall y ((P(y) \rightarrow R(c, x)) \wedge (\neg P(z) \rightarrow R(x, y)) \wedge \neg R(z, z))$.

3.10 Esercizi

1. Verificare con la risoluzione se i seguenti insiemi di clausole sono insoddisfacibili:

$$\{P(x), \neg P(x) \vee Q(F(x)), \neg Q(F(c))\}$$

$$\{P(x), Q(x, F(x)) \vee \neg P(x), \neg Q(G(y), z)\}$$

$$\{P(x) \vee Q(F(x)), \neg P(c) \vee R(x, y), \neg R(c, x), \neg Q(F(c))\}$$

$$\{\neg P(x), \neg R(F(x), y), P(F(x)) \vee R(x, c) \vee Q(x), \neg Q(x) \vee P(y)\}$$

$$\{\neg P(x) \vee Q(x), \neg Q(c) \vee \neg P(x), P(x) \vee R(F(x)), \neg R(x)\}.$$

2. Dimostrare con la risoluzione che

$$\{\neg P(x), \neg R(F(x), y), P(F(x)) \vee R(x, c) \vee Q(x), \neg Q(x) \vee P(y)\}$$

è insoddisfacibile.

3. Dimostrare che

$$\{\neg P(x) \vee R(x, F(x)), \neg R(c, x) \vee Q(x), \neg Q(G(x)) \vee P(x), P(c) \vee Q(x), \neg P(x) \vee \neg Q(F(x))\}$$

è insoddisfacibile, mentre

$$\{\neg P(x) \vee R(x, F(x)), \neg R(c, x) \vee Q(x), \neg Q(G(x)) \vee P(G(x)), P(c) \vee Q(x), \neg P(x) \vee \neg Q(F(x))\}$$

non lo è.

4. Verificare, riconducendosi al calcolo della risoluzione, se

$$\forall x \exists y (P(x, y) \vee Q(x, y)), \forall x \forall y (P(x, y) \rightarrow R(x)),$$

$$\forall x \forall y (Q(x, y) \rightarrow R(x)) \models \exists x R(x)$$

$$\forall x P(x), \exists x P(x) \rightarrow \forall x \exists y R(x, y) \models \forall x \exists y R(x, y)$$

$$\exists x \neg P(x) \rightarrow \exists y \neg R(y), \forall x P(x) \rightarrow \exists x Q(x),$$

$$\exists x Q(x) \rightarrow \neg \forall x R(x) \models \exists x \neg R(x)$$

$$\exists x P(x) \rightarrow \forall x R(c, x), \forall x (\forall z \neg P(z) \rightarrow \forall v R(x, v)) \models$$

$$\models \exists x \forall y R(x, y).$$

3.11 Linguaggi con uguaglianza

Il fatto che $P(x)$ e $\neg P(c)$ siano un insieme refutabile per risoluzione, con l'unificatore $\{x/c\}$, potrebbe far pensare che il suo significato sia che $P(x)$ e $\neg P(c)$ sono incompatibili con $x = c$, ma non è così: significa che $\forall x P(x)$ e $\neg \forall x P(x)$ sono incompatibili tra loro.

$P(x)$ e $\neg P(y)$ sono incompatibili con $x = y$ nei modelli normali, nel senso che in essi $P(x) \wedge \neg P(y) \rightarrow x \neq y$ è valida (ma non lo è la sua contrapposta, che è un assioma dell'uguaglianza).

Esempio Se si considera $x = y \rightarrow (R(x, z) \rightarrow R(y, z))$, la sua negazione non fornisce le tre clausole $x = y$, $R(x, z)$ e $\neg R(y, z)$, ma le clausole $c = d$, $R(c, e)$ e $\neg R(d, e)$, dove c , d ed e sono costanti, e non è possibile derivare la clausola vuota. La negazione trasforma le variabili universali in costanti.

La formula è valida nei modelli normali; infatti se si aggiunge la sua negazione agli assiomi dell'uguaglianza, allora la clausola vuota è derivabile utilizzando anche il particolare assioma dell'uguaglianza $x = y \rightarrow (R(x, e) \rightarrow R(y, e))$ che è la clausola $x \neq y \vee \neg R(x, e) \vee R(y, e)$ (esercizio).

Gli assiomi dell'uguaglianza sono clausole di Horn.

Nel contesto della risoluzione, invece di eseguire risoluzioni con gli assiomi dell'uguaglianza, è molto più efficiente utilizzare una regola aggiuntiva, accanto a quella della risoluzione (ordinata):

Regola di paramodulazione: Da due clausole $C \vee L(\dots t \dots)$ e $D_1 \vee r = s \vee D_2$, se r e t sono unificabili con l'unificatore generale σ , dedurre $C\sigma \vee L(\dots s \dots)\sigma \vee D_1\sigma \vee D_2\sigma$.

Nell'esempio di sopra, da $c = d$ e $R(c, e)$ si può dedurre immediatamente $R(d, e)$, e risolverla con $\neg R(d, e)$.

In presenza della regola di paramodulazione si può fare a meno degli assiomi dell'uguaglianza, salvo i riflessivi.

3.12 Programmazione logica

Esempio Sia data la seguente storia: Giovanni ama Maria? Se uno si emoziona quando guarda una ragazza, allora si può dire che l'ama. Giovanni si emoziona se sta vicino a Maria e Maria sorride. Uno sorride quando è contento. Se a uno fanno un regalo che gli piace, questi è contento. A Maria piacciono i dischi di Baglioni. Giovanni regala a Maria un disco di Baglioni e sta sempre vicino a Maria e non le toglie gli occhi di dosso.

Con una opportuna semplificazione e formalizzazione, la storia può essere resa dalla congiunzione dei seguenti enunciati:

$$\begin{aligned} & \forall x \forall y (G(x, y) \wedge E(x) \rightarrow (R(y) \rightarrow A(x, y))) \\ & R(m) \\ & V(g, m) \wedge S(m) \rightarrow E(g) \\ & \forall x (C(x) \rightarrow S(x)) \\ & \forall x \forall z (\exists y (R(x, y, z) \wedge P(y, z)) \rightarrow C(z)) \\ & \forall x (D(x) \rightarrow P(x, m)) \\ & \exists x (D(x) \wedge R(g, x, m)) \\ & V(g, m) \\ & V(g, m) \rightarrow G(g, m), \end{aligned}$$

ovvero in forma di clausole:

$$\begin{aligned} 1 & \quad A(x, y) \leftarrow R(y), G(x, y), E(y) \\ 2 & \quad R(m) \leftarrow \\ 3 & \quad E(g) \leftarrow V(g, m), S(m) \\ 4 & \quad S(z) \leftarrow C(z) \\ 5 & \quad C(w) \leftarrow R(u, v, w), P(v, w) \\ 6 & \quad P(j, m) \leftarrow D(j) \\ 7 & \quad D(d) \leftarrow \\ 8 & \quad R(g, d, m) \leftarrow \\ 9 & \quad V(g, m) \leftarrow \\ 10 & \quad G(g, m) \leftarrow V(g, m), \end{aligned}$$

con il *goal*

$$11 \quad \leftarrow A(g, m),$$

a cui si risponde con la seguente derivazione:

$$\begin{aligned} 12 & \quad \leftarrow R(m), G(g, m), E(g) \quad \text{da 1 con } \{x/g, y/m\} \\ 13 & \quad \leftarrow G(g, m), E(g) \quad \text{da 2} \\ 14 & \quad \leftarrow V(g, m), E(g) \quad \text{da 10} \\ 15 & \quad \leftarrow E(g) \quad \text{da 9} \\ 16 & \quad \leftarrow V(g, m), S(m) \quad \text{da 3} \\ 17 & \quad \leftarrow S(m) \quad \text{da 9} \\ 18 & \quad \leftarrow C(m) \quad \text{da 4 con } \{z/m\} \\ 19 & \quad \leftarrow R(u, v, m), P(v, m) \quad \text{da 5 con } \{w/m\} \\ 20 & \quad \leftarrow P(d, g) \quad \text{da 8 con } \{u/g, v/d\} \\ 21 & \quad \leftarrow D(d) \quad \text{da 6 con } \{j/d\} \\ 22 & \quad \leftarrow \quad \text{da 7} \end{aligned}$$

e il *goal* è soddisfatto.

Esempio Sia dato il programma

$$\begin{array}{l} 1 \quad P(x) \leftarrow \\ 2 \quad R(c, y) \leftarrow P(y) \end{array}$$

con il *goal*

$$3 \quad \leftarrow R(z, F(z)),$$

Ad esso si risponde unificando il *goal* con la testa della 2, ottenendo come nuovo *goal*

$$4 \quad \leftarrow P(F(c))$$

e quindi

$$5 \quad \leftarrow$$

con la sostituzione $\{x/F(c)\}$.

Si noti che in una reale applicazione della programmazione logica la risposta che sarebbe data è

$$z := c$$

in quanto la richiesta non è tanto se il *goal* è soddisfatto, ma da chi. In notazione logica, l'interrogazione dell'esempio deriva dal desiderio di sapere se

$$\forall x P(x), \forall x (P(x) \rightarrow R(c, x)) \models \exists x R(x, F(x)).$$

Le variabili del *goal* derivano da quantificatori esistenziali, e richiedono per la risposta positiva un'esemplificazione specifica; se il *goal* deriva da una putativa conseguenza universale del programma, allora non ha variabili ma costanti, come effetto della trasformazione mediante la negazione.