GRAU EN ENGINYERIA INFORMÀTICA

LÒGICA COMPUTACIONAL

Notes de Lògica

Felip Manyà i Serres
IIIA-CSIC

Índex

Capítol 1

Lògica Proposicional

1.1 Introducció

La Lògica s'ocupa de l'estudi dels mecanismes que permeten fer raonaments vàlids. Un raonament diem que és vàlid si sempre que les premisses són certes la conclusió és necessàriament certa. La lògica no estudia la veritat o falsetat de les premisses i la conclusió aïlladament, sinó la relació entre la veritat o falsetat de les premisses i la veritat o falsetat de la conclusió. Dit d'una altra manera, li interessa com es propaga la veritat des de les premisses cap a la conclusió. Tot seguit veurem alguns exemples de raonaments vàlids:

	Si el cotxe no té gasolina llavors no arranca
	El cotxe no té gasolina
per tant	el cotxe no arranca

	Si el cotxe no té asseguranca llavors no arranca
	El cotxe no té assegurança
per tant	el cotxe no arranca

Observem en aquests raonaments que no es pot donar que les premisses siguin certes i la conclusió falsa. Aquest no és el cas del següent raonament:

	Si el cotxe no té gasolina llavors no arranca
	el cotxe no arranca
per tant	el cotxe no te gasolina

aquest és un raonament no vàlid, ja que poden haver moltes altres raons, apart que no hi hagi gasolina, per a que el cotxe no arranqui. La validesa d'un raonament no té res a veure amb la temàtica que tracten les premisses i les conclusions, ni amb la seva veritat o falsetat. La validesa, des d'un punt de vista lògic, depén exclusivament de l'estructura (forma) del raonament. Els raonaments vàlids que hem fet servir com exemples tenen la següent forma:

	Si A llavors B
	A
per tant	B

Així doncs, si substituim A i B per qualsevol enunciat obtindrem un raonament vàlid. Noteu que hi ha altres estructures de raonaments vàlids, com per exemple:

	Si A llavors B
	Si A llavors C
per tant	Si A llavors B i C

Els lògics construeixen sistemes que formalitzen la noció de raonament vàlid. El primer pas, per a construir aquests sistemes, és definir un llenguatge formal en el qual poguéssim representar les premisses i les conclusions com enunciats d'aquest llenguatge. Aquesta part s'anomena *sintaxi* de la lògica. Ara bé, com nosaltres estem interessats en com es propaga la veritat de les premisses cap a les conclusions, hem de definir amb precisió quan un enunciat del llenguatge formal és veritat o fals per a una interpretació particular. D'aquesta part se n'encarrega la *semàntica* o *teoria de models*. Apart de la sintaxi i la semàntica, una lògica té una *teoria de la prova*. La teoria de la prova estudia els procediments automatitzables (i les seves propietats) que permeten trobar si un raonament és vàlid mitjançant la manipulació simbòlica dels enunciats del llenguatge, sense tenir en compte la seva semàntica. La lògica que estudiarem en aquest capítol és el Càlcul Proposicional (CP_0), també anomenada lògica d'enunciats, lògica proposicional i Càlcul de predicats d'ordre 0, d'on vé CP_0.

1.2 El Llenguatge del CP_0

El llenguatge del CP_0 l'utilitzarem per a representar enunciats del llenguatge natural (en el nostre cas del català) dels que podem dir que o bé són veritat o bé són falsos. Aquest tipus d'enunciats s'anomenen *enunciats declaratius* i els emprem per fer raonaments. Els enunciats declaratius simples els representarem en CP_0 per les lletres majúscules P,Q i R, posant subindexos a la P si és necessari, i les anomenarem *àtoms*. Els enunciats declaratius compostos els representarem com una combinació d'àtoms i connectives. Les *connectives* són una formalització de les partícules no, i, o, la construcció condicional si... llavors..., i la bicondicional si i només si:

- $\neg$ nota la partícula *no* i s'anomena negació.
- $\wedge$ nota la partícula *i* i s'anomena conjunció.
- $\vee$ nota la partícula *o* i s'anomena disjunció.
- $\rightarrow$ nota la construcció *si... llavors...* i s'anomena condicional o implicació material.
- $\leftrightarrow$ nota la construcció *si i només si* i s'anomena bicondicional.

Exemple 1.1 L'enunciat declaratiu "si en Joan canta òpera i na Margarida toca el piano llavors els veïns no poden fer migdiada " es pot descomposar amb els següents enunciats declaratius simples:

P = Joan canta òpera.
Q = Na Margarida toca el piano.
R = Els veïns poden fer migdiada.

Les lletres d'àtoms P, Q, R denoten aquests enunciats simples i junt amb les connectives de conjunció, condicional i negació ens permeten representar l'enunciat declaratiu com un enunciat del CP_0 de la següent manera:

$$((P \wedge Q) \rightarrow (\neg R))$$

El llenguatge del CP_0 és un llenguatge formal i el definirem donant l'alfabet dels seus símbols i les regles gramaticals que hem de seguir per a la construcció de les seves fòrmules o enunciats.

Definició 1.1 (alfabet). *L'alfabet $\sum$ del llenguatge del CP_0 té els següents components:*

- *Un conjunt de símbols d'àtoms* $\{P, Q, R, P_1, P_2, P_3, \ldots\}$
- *Un conjunt de símbols de connectives* $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$
- *Un conjunt de símbols de puntuació* $\{(,)\}$

$$\sum = \{\neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,), P, Q, R, P_1, P_2, P_3, \ldots\}$$

Definició 1.2 (enunciat). *Donat un alfabet $\sum$ direm que*

- *Tot àtom de $\sum$ és un enunciat.*
- *Si A és un enunciat, aleshores $(\neg A)$ també ho és.*
- *Si A i B són enunciats, aleshores $(A \wedge B), (A \vee B), (A \rightarrow B)$ i $(A \leftrightarrow B)$ també ho són.*
- *No hi ha cap altre enunciat.*

Notació: Les lletres $A, B, C, A_1, A_2, A_3, \ldots$ ...noten enunciats i les lletres gregues $\Gamma, \Delta, \Theta, \Gamma_1, \Gamma_2, \Gamma_3 \ldots$ noten conjunts d'enunciats. El conjunt de tots els enunciats que podem construir sobre l'alfabet $\sum$ usant les regles anteriors forma el *llenguatge del CP_0*. Noteu que de fet no definim un llenguatge únic sinó una familia de llenguatges ja que el conjunt de símbols d'àtom no està prefixat. Notarem $\mathcal{L}(\mathcal{P})$ aquell llenguatge proposicional que conté a $\mathcal{P}$ com a conjunt de símbols d'àtom. $\mathcal{P}$ s'acostuma a dir que és el conjunt de *variables proposicionals*. Aquestes regles ens permeten programar un procediment recursiu per a decidir si un enunciat pertany o no pertany al llenguatge.

Exemple 1.2 Les expressions $P, (P \vee Q), ((\neg P) \rightarrow Q), ((\neg(P_1 \wedge P_3)) \rightarrow (P_2 \vee P_4)), ((P \vee Q) \wedge (\neg R))$ són enunciats, mentre que les expressions $((\neg(P_1 P_2)) \rightarrow P_3 \vee P_4)), (P \vee Q)(\neg R)$ no ho són.

Podem relaxar la notació, per a facilitar la lectura, amb els següents convenis:

1. Eliminar els parèntesis exteriors. Per exemple $P_1 \wedge P_2$ es llegeix $(P_1 \wedge P_2)$.
2. Associar una prioritat a les connectives, decreixentment, així:
$$\leftrightarrow, \rightarrow, \vee, \wedge, \neg$$
de manera que les connectives amb prioritat més alta tenen més àmbit. Per exemple, $\neg P_1 \wedge P_2$ es llegeix $((\neg P_1) \wedge P_2)$ i $Q \rightarrow R \leftrightarrow P$ es llegeix $((Q \rightarrow R) \leftrightarrow P)$.
3. Quan hi ha varies ocurrències d'una mateixa connectiva seguim la regla d'associativitat per l'esquerra. Per exemple, $P_1 \rightarrow P_2 \rightarrow P_3$ es llegeix $((P_1 \rightarrow P_2) \rightarrow P_3)$.

1.2.1 Llenguatge i metallenguatge

Quan treballem en Lògica cal distingir entre el llenguatge propi de la lògica, anometat *llenguatge objecte*, i el llenguatge que emprem per a parlar del llenguatge de la lògica, anomenat *metallenguatge* o *llenguatge de l'observador*. Per aclarir aquests conceptes penseu en quan estudiaveu llatí. El llenguatge objecte era el llatí i el metallenguatge era el català. Es de suposar que no feieu les classes en llatí!. El mateix passa quan estudieu un llenguatge de programació. En el cas del CP_0 el llenguatge objecte és el que hem definit a l'apartat anterior. Si us hi fixeu, les lletres que noten els atoms, anomenades variables proposicionals, pertanyen al llenguatge del CP_0, mentre que les lletres que utilitzem per a notar enunciats no hi pertanyen. En aquest cas, no parlem de variables sinó de *metavariables*.

1.3 Semàntica

La semàntica estudia la relació entre un llenguatge formal i les seves interpretacions, utilitzant el concepte de veritat d'un enunciat com a pont. Des d'un punt de vista sintàctic, un llenguatge és un conjunt d'enunciats, on els enunciats estan formats a partir de connectives i símbols d'àtoms. O sigui, són strings de caràcters construïts seguint unes determinades regles gramaticals i no tenen cap significat. Atribuir significat a un símbol d'àtom consisteix en assignar-li un dels dos valors de veritat: verdader (V) o fals (F). Assignem el valor verdader quan l'enunciat declaratiu simple que denota l'àtom creiem que és cert en el context sobre el qual raonem; altrament li assignem el valor fals. El significat d'un enunciat format a partir de dos enunciats més simples i una connectiva es defineix en funció del valor de veritat dels enunciats simples. Com els enunciats els construïm combinant àtoms i connectives, si sabem el valor de veritat dels àtoms que intervenen en un enunciat podem saber el valor de veritat de l'enunciat, ja que el significat de les connectives no depén del contexte sobre el que raonem.

Definició 1.3 (interpretació). *Sigui $\mathcal{L}(\mathcal{P})$ un llenguatge del CP_0, on $\mathcal{P}$ és el conjunt de variables proposicionals. Una interpretació de $\mathcal{L}(\mathcal{P})$ és una funció $\mathcal{I}$ amb domini el conjunt de variables proposicionals $\mathcal{P}$ i amb rang el conjunt de valors de veritat $\{V, F\}$.*

Una interpretació, per tant, consisteix en assignar un valor de veritat a cada variable proposicional. Si A és un enunciat d'un llenguatge $\mathcal{L}(\mathcal{P})$ pel qual hem definit una interpretació $\mathcal{I}$ i $P_1, \ldots, P_n$ són els àtoms que ocorren a A, la funció $\mathcal{I}$ restringida a $P_1, \ldots, P_n$ diem que és una interpretació de A. La següent definició estableix el significat de les connectives i dóna les regles que permeten conèixer el valor de veritat d'un enunciat per a una interpretació.

Definició 1.4 (semàntica dels enunciats). *Siguin A i B enunciats. El significat d'un enunciat per una interpretació $\mathcal{I}$ és defineix com:*

- *Si A és un àtom llavors A és veritat per a $\mathcal{I}$ si $\mathcal{I}$ assigna el valor V a A ($\mathcal{I}(A) = V$). Altrament A és fals.*
- *$\neg A$ és veritat per a $\mathcal{I}$ quan A és fals, i fals quan A és veritat.*

- $(A \wedge B)$ *és veritat per a* $\mathcal{I}$ *quan tant* A *com* B *són veritat. Altrament és fals.*
- $(A \vee B)$ *és veritat per a* $\mathcal{I}$ *quan almenys un dels enuncitas (* A *o* B*) és veritat. Altrament és fals.*
- $(A \rightarrow B)$ *és fals per a* $\mathcal{I}$ *quan* A *és veritat i* B *és fals. Altrament és veritat.*
- $(A \leftrightarrow B)$ *és veritat per a* $\mathcal{I}$ *quan* A *i* B *tenen el mateix valor de veritat. Altrament és fals.*

Noteu que aquestes regles defineixen amb precisió el significat de les connectives, i possiblement no correspón a la manera que vosaltres les empreu en català. Si, en canvi, el seu significat correspón a com les empreu quan construïu expresions booleanes a l'hora de programar. Per exemple, la connectiva $\vee$ fa referència al o inclusiu. La construcció condicional $\rightarrow$ estableix que no es pot donar el cas que l'enunciat a la seva esquerra, anomenat *antecedent*, sigui veritat i el de la seva dreta, anomenat *conseqüent*, sigui fals. Una altra manera d'expressar les anteriors regles és mitjançant les següents taules, anomenades taules de veritat.

A	B	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$
V	V	V	V	V	V
V	F	F	V	F	F
F	V	F	V	V	F
F	F	F	F	V	V

Si la connectiva és la negació, llavors sols involucra un enunciat.

A	$\neg A$
V	F
F	V

Donats un llenguatge $\mathcal{L}(\mathcal{P})$ i una interpretació $\mathcal{I}$ per aquest llenguatge, un mètode per assignar un valor de veritat als enunciats del llenguatge és aplicar les regles que defineixen el seu significat. Un altre mètode és utilitzar les taules de veritat com es mostra al següent exemple:

Exemple 1.3 Sigui $\mathcal{I}$ una interpretació que assigna a P el valor V i a Q el valor F, per saber quin valor de veritat assigna $\mathcal{I}$ a l'enunciat $P \wedge Q$ busquem la fila de la taula on A és V i B és F. El valor de $P \wedge Q$ és el valor que pren $A \wedge B$ en aquesta fila. En el nostre cas és F. Noteu que hem associat els àtoms P i Q amb els enunciats A i B de la taula de veritat.

Donat un enunciat qualsevol podem construir la seva taula de veritat. La taula de veritat d'un enunciat ens dóna el valor de veritat de l'enunciat per a totes les possibles interpretacions dels seus àtoms. Aquesta taula de veritat es construeix incrementalment com es mostra al següent exemple.

Exemple 1.4 La taula de veritat de l'enunciat $P \rightarrow Q \wedge \neg P$ és:

P	Q	$\neg P$	$Q \wedge \neg P$	$P \rightarrow Q \wedge \neg P$
V	V	F	F	F
V	F	F	F	F
F	V	V	V	V
F	F	V	F	V

Noteu que si un enunciat té n àtoms distints, aleshores hi ha 2^n possibles interpretacions per aquest enunciat.

Definició 1.5 (model d'un enunciat). *Sigui $\mathcal{I}$ una interpretació i A un enunciat. Direm que $\mathcal{I}$ és un model de A o que $\mathcal{I}$ satisfà A, notat $\models_{\mathcal{I}} A$, ssi $\mathcal{I}$ assigna el valor de veritat verdader a A . Altrament, direm que $\mathcal{I}$ falsifica A i ho notarem $\not\models_{\mathcal{I}} A$.*

Una interpretació $\mathcal{I}$ és un model d'un conjunt d'enunciats Γ ssi és un model de cadascún dels enunciats de Γ. Noteu que si $\Gamma = \{A_1, \ldots, A_n\}$, llavors $\mathcal{I}$ és un model de Γ ssi $\mathcal{I}$ és un models de $A_1 \wedge \ldots \wedge A_n$.

1.4 Tipus d'enunciats

Tot seguit definirem els conceptes d'enunciat vàlid i enunciat insatisfactible. Aquests enunciats sempre prenen el mateix valor de veritat per a totes les possibles interpretacions.

Definició 1.6 (enunciat vàlid o tautologia). *Un enunciat A direm que és vàlid ssi totes les possibles interpretacions satisfan A. També es diu que és una tautologia i es nota per $\models A$. En cas contrari es diu que A és invàlid i es nota per $\not\models A$.*

Una manera de saber si un enunciat és una tautologia és construir la taula de veritat i mirar si s'avalua a V per a totes les interpretacions.

Exemple 1.5 Demostrar que $\neg A \vee A$ és una tautologia ($\models \neg A \vee A$).

A	$\neg A$	$\neg A \vee A$
V	F	V
F	V	V

Teorema 1.1 *Si A i $A \to B$ són tautologies llavors B és una tautologia.*

Demostració: Suposem que B no és una tautologia. Aleshores existeix una interpretació $\mathcal{I}$ on B pren el valor F. Com A és una tautologia pren el valor V per totes les interpretacions. Per tant, existeix una interpretació $\mathcal{I}$ on A pren el valor V i B el valor F. Per aquesta interpretació $\mathcal{I}$, $A \to B$ pren el valor F, contra la hipòtesi, ja que $A \to B$ és una tautologia. Per tant, B és una tautologia. ■

Teorema 1.2 *Sigui A un enunciat on apareixen les variables proposicionals $P_1, \ldots, P_n$ i siguin $A_1, \ldots, A_n$ enunciats. Si A és una tautologia llavors l'enunciat B, obtingut de A reemplaçant cada ocurrència de P_i per A_i $(1 \leq i \geq n)$, és una tautologia.*

Demostració: Sigui $\mathcal{I}$ una interpretació arbitrària. Suposem que $A_1, \ldots, A_n$ prenen el valors de veritat $x_1, \ldots, x_n$ a $\mathcal{I}$. Si assignem els valors de veritat $x_1, \ldots, x_n$ a $P_1, \ldots, P_n$, el valor de veriat de A és el mateix que el valor de veritat de B. Com A és una tautologia el seu valor de veritat per qualsevol interpretació és V. Per tant, B sempre pren el valor V. ■

La inversa no és necessàriament certa. Per exemple, si notem A la tautologia $P \vee \neg P$ ($\models A$) i reemplacem A per l'àtom P quedaria $\models P$, i això mai és cert.

Definició 1.7 (enunciat insatisfactible). *Un enunciat A direm que és insatisfactible ssi totes les possibles interpretacions falsifiquen A. També es diu que és inconsistent i es nota per $\models \neg A$. En cas contrari es diu que A és satisfactible i es nota per $\not\models \neg A$.*

Una manera de saber si un enunciat és insatisfactible és construir la taula de veritat i mirar si s'avalua a F per a totes les interpretacions.

Exemple 1.6 Demostrar que $\neg A \wedge A$ és insatisfactible ($\models \neg(\neg A \wedge A)$).

A	$\neg A$	$\neg A \wedge A$
V	F	F
F	V	F

Teorema 1.3 *Un enunciat A és vàlid ssi $\neg A$ és insatisfactible.*

Demostració: ($\Rightarrow$) Suposem que A és vàlid. Per a qualsevol interpretació $\mathcal{I}$ es té $\models_{\mathcal{I}} A$ i $\not\models_{\mathcal{I}} \neg A$, ja que $\neg A$ és fals per a una interpretació I quan A és veritat, segons la definició semàntica de $\neg$. Per tant $\neg A$ és insatisfactible. ($\Leftarrow$) Suposem que $\neg A$ és insatisfactible. Per a qualsevol interpretació $\mathcal{I}$ es té $\not\models_{\mathcal{I}} \neg A$ i $\models_{\mathcal{I}} A$, ja que A és veritat per a una interpretació $\mathcal{I}$ quan $\neg A$ és fals, segons la definició semàntica de $\neg$. Per tant A és vàlid. ■

Aquest teorema estableix que hi ha dues maneres de provar si un enunciat és vàlid: o bé ho podem fer directament, construint la taula de veritat de A i comprovant que A és veritat per totes les interpretacions, o bé ho podem fer indirectament demostrant que $\neg A$ és insatisfactible. Aquest darrer mètode s'anomena *reducció a l'absurd* o *refutació*. Un conjunt d'enunciats $\Gamma = \{A_1, \ldots, A_n\}$ és satisfactible ssi té almenys un model, és vàlid ssi totes les interpretacions són models de Γ i és insatisfactible ssi no té cap model. Noteu que demostrar la satisfactibilitat, validesa o insatisfactibilitat de Γ és equivalent a demostrar-ho per l'enunciat $A_1 \wedge \ldots \wedge A_n$. Noteu que no tots els enunciats són vàlids o insatisfactibles. Els enunciats que són invàlids i satisfactibles s'anomenen enunciats contingents.

Definició 1.8 (enunciat contingent). *Un enunciat A direm que és contingent ssi existeixen interpretacions que satisfan A i interpretacions quefalsifiquen A.*

Els diferents tipus d'enunciats queden reflexats a la figura 1.1.

1.5 Conseqüència lògica

En un raonament distinguim entre les premisses i la conclusió. Direm que el raonament és vàlid si sempre que les premisses són certes la conclusió és necessàriament certa. El concepte de conseqüència lògica formalitza la idea de raonament vàlid.

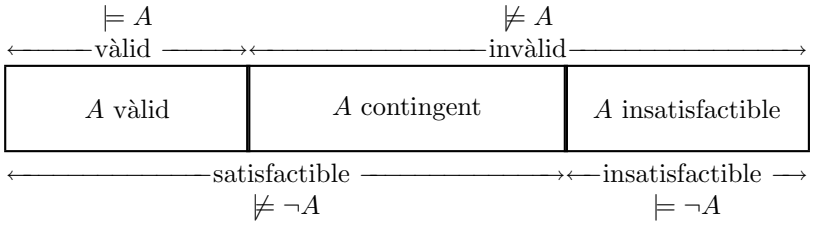


Figura 1.1: tipus d'enunciats

Definició 1.9 (conseqüència lògica). *Siguin $\{A_1, A_2, \ldots, A_n\}$ un conjunt finit d'enunciats i B un enunciat. Direm que B és una conseqüència lògica de $\{A_1, A_2, \ldots, A_n\}$ ssi per a qualsevol interpretació $\mathcal{I}$ que satisfà $A_1 \wedge A_2 \wedge ... \wedge A_n$, $\mathcal{I}$ també satisfà B. Es nota per $A_1, A_2, \ldots, A_n \models B$.*

Si $\Gamma = \{A_1, A_2, \ldots, A_n\}$ és un conjunt finit d'enunciats, la relació $A_1, A_2, \ldots, A_n \models B$ s'escriu $\Gamma \models B$. Els següents teoremes redueixen el problema de demostrar que un enunciat és una conseqüència lògica d'un conjunt d'enunciats a demostrar la validesa o insatisfactibilitat d'aquests enunciats.

Teorema 1.4 *Siguin $A_1, A_2, ..., A_n$ i B enunciats. B és una conseqüència lògica de $A_1, A_2, \ldots, A_n$ ssi l'enunciat $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \rightarrow B)$ és vàlid.*

Demostració: ($\Rightarrow$) Sigui $\mathcal{I}$ una interpretació qualsevol. Donat l'enunciat $A_1 \wedge A_2 \wedge \ldots \wedge A_n$ hi ha dues possibilitats:

- $\mathcal{I}$ satisfà $A_1 \wedge A_2 \wedge \ldots \wedge A_n$. Per definició de conseqüència lògica $\mathcal{I}$ també satisfà B. Si $\mathcal{I}$ satisfà l'antecedent i el conseqüent, també satisfa el condicional. Per tant $\mathcal{I}$ satisfà $((A_1 \wedge A_2 \wedge ... \wedge A_n) \rightarrow B)$.
- $\mathcal{I}$ no satisfà $A_1 \wedge A_2 \wedge \ldots \wedge A_n$. Si $\mathcal{I}$ no satisfà l'antecedent, $\mathcal{I}$ sempre satisfà el condicional. Per tant $\mathcal{I}$ satisfà $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \rightarrow B)$.

($\Leftarrow$) Suposem que $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \rightarrow B)$ és vàlid. Per tant, per a qualsevol interpretació $\mathcal{I}$, si $\mathcal{I}$ satisfà $(A_1 \wedge A_2 \wedge \ldots \wedge A_n)$ també satisfà B. Per tant, $A_1, A_2, \ldots, A_n \models B$. ■

Teorema 1.5 *Siguin $A_1, A_2, \ldots, A_n$ i B enunciats. B és una conseqüència lògica de $A_1, A_2, ..., A_n$ ssi l'enunciat $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$ és insatisfactible.*

Demostració: ($\Rightarrow$) Suposem que $A_1, A_2, \ldots, A_n \models B$. Donada una interpretació $\mathcal{I}$ hi ha dues possibilitats:

- $\mathcal{I}$ és model de $A_1, A_2, \ldots, A_n$. Com $A_1, A_2, \ldots, A_n \models B$ es té que $\mathcal{I}$ satisfà B. Si $\mathcal{I}$ satisfà B llavors $\mathcal{I}$ no satisfà $\neg B$. Per tant, $\mathcal{I}$ no és model de $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$.
- $\mathcal{I}$ no és model de $A_1, A_2, \ldots, A_n$. Per tant, $\mathcal{I}$ no és model de $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$.

$((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$ és insatisfactible, ja que com acabem de demostrar no és satisfà per cap interpretació. ($\Leftarrow$) Suposem que $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$

és insatisfactible. Sigui $\mathcal{I}$ una interpretació que satisfà $A_1 \wedge A_2 \wedge \ldots \wedge A_n$. Com $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$ és insatisfactible, $\mathcal{I}$ no pot ser model de $\neg B$. Per tant, $A_1, A_2, \ldots, A_n \models B$, ja que tots els models de $A_1 \wedge A_2 \wedge \ldots \wedge A_n$ també ho són de B. ■

Exemple 1.7 Una manera de verificar que Q és una conseqüència lògica de $P \to Q, P$ és demostrar que $((P \to Q) \wedge P) \to Q$ és vàlid:

P	Q	$P \to Q$	$(P \to Q) \wedge P$	$((P \to Q) \wedge P) \to Q$
V	V	V	V	V
V	F	F	F	V
F	V	V	F	V
F	F	V	F	V

Una altra manera de comprovar que

$$P \to Q, P \models Q$$

és verificar que $\{P \to Q, P, \neg Q\}$ és un conjunt d'enunciats insatisfactible.

1.6 Equivalències lògiques

En aquesta secció estudiem quan dos enunciats són semànticament equivalents. Dit d'una altra manera, quan són lògicament equivalents.

Definició 1.10 (enunciats lògicament equivalents). *Siguin A i B dos enunciats. Direm que A i B són lògicament equivalents si i només si els models de A i B coincideixen. Es nota $A \equiv B$. L'expressió $A \equiv B$ s'anomena equivalència.*

Noteu que el símbol d'equivalència $\equiv$ és un metasímbol. Tot seguit donem alguns exemples d'equivalències:

$$
\begin{array}{rcll}
A \wedge A & \equiv & A & \\
A \vee A & \equiv & A & \text{(Idempotència)} \\
 & & & \\
A \wedge B & \equiv & B \wedge A & \\
A \vee B & \equiv & B \vee A & \text{(Commutativitat)} \\
 & & & \\
(A \wedge B) \wedge C & \equiv & A \wedge (B \wedge C) & \\
(A \vee B) \vee C & \equiv & A \vee (B \vee C) & \text{(Associativitat)} \\
 & & & \\
(A \wedge (A \vee B)) & \equiv & A & \\
(A \vee (A \wedge B)) & \equiv & A & \text{(Absorció)} \\
 & & & \\
(A \wedge (B \vee C)) & \equiv & ((A \wedge B) \vee (A \wedge C)) & \\
(A \vee (B \wedge C)) & \equiv & ((A \vee B) \wedge (A \vee C)) & \text{(Distributivitat)} \\
 & & & \\
\neg(\neg A) & \equiv & A & \text{(Doble negació)} \\
 & & & \\
\neg(A \wedge B) & \equiv & \neg A \vee \neg B & \\
\neg(A \vee B) & \equiv & \neg A \wedge \neg B & \text{(Lleis de De Morgan)} \\
 & & & \\
(A \wedge B) & \equiv & B, \text{ si } A \text{ és una tautologia} & \\
(A \vee B) & \equiv & A, \text{ si } A \text{ és una tautologia} & \text{(Lleis de la tautologia)} \\
 & & & \\
(A \wedge B) & \equiv & A, \text{ si } A \text{ és insatisfactible} & \\
(A \vee B) & \equiv & B, \text{ si } A \text{ és insatisfactible} & \text{(Lleis d'insatisfactibilitat)}
\end{array}
$$

La demostració d'aquestes equivalències, que deixem pel lector, es pot fer usant la definició de la semàntica dels enunciats o usant les taules de veritat, on caldrà comprovar que l'enunciat a l'esquerra del símbol d'equivalència té els mateixos models que l'enunciat a la dreta. Nosaltres hem definit el llenguatge del CP_0 considerant $\{\neg, \wedge, \vee, \rightarrow, \leftrightarrow\}$ com a conjunt de connectives. També haguessim pogut definir-lo prenent el conjunt $\{\neg, \wedge, \vee\}$, ja que qualsevol enunciat de la forma $A \rightarrow B$ o $A \leftrightarrow B$ és lògicament equivalent a un enunciat de la forma $\neg A \vee B$ o $(\neg A \vee B) \wedge (\neg B \vee A)$, respectivament. Els conjunts de connectives amb aquesta propietat s'anomenen *adequats* o *funcionalment complets*. Altres exemples de conjunts adequats són $\{\neg, \vee\}$, $\{\neg, \wedge\}$ i $\{\neg, \rightarrow\}$. Si bé ens és més còmode treballar amb totes les connectives, ja que les expressions resulten més curtes i són més intuitives, a l'hora de fer demostracions per inducció sobre l'estructura dels enunciats escollirem el conjunt $\{\neg, \wedge, \vee\}$.

Teorema 1.6 (teorema de substitució). *Siguin A_1 i B_1 enunciats lògicament equivalents. Sigui A un enunciat que conté A_1 com a subenunciat. Si B és l'enunciat obtingut a partir de A substituint una ocurrència de A_1 per B_1 llavors A i B són lògicament equivalents.*

Demostració: Ho provarem per inducció sobre el nombre (n) de connectives que ocorren a l'enunciat A. *Pas base:* $n = 0$ (A no té connectives). En aquest cas A és un àtom. Per tant $A = A_1$ i $B = B_1$. Com A_1 i B_1 són lògicament equivalents es té $A \equiv B$. *Pas inducció:* Suposarem que A té $n > 0$ connectives i que tot enunciat amb menys de n connectives té la propietat requerida. A més a més, suposarem que $A \neq A_1$ ja que si $A = A_1$ es té que $B = B_1$, i com $A_1 \equiv B_1$

llavors $A \equiv B$. Com $\{\neg, \vee, \wedge\}$ és un conjunt adequat de connectives hem de considerar 3 casos: *Cas 1:* A és de la forma $\neg C$. A_1 és un subenunciat de C. Per hipòtesi d'inducció $C \equiv C'$, on C' l'hem obtingut de C substituint A_1 per B_1. Per tant $B = \neg C'$, i per la definició semàntica de $\neg$ es té que $A \equiv B$. *Cas 2:* A és de la forma $(C_1 \vee C_2)$. A_1 és un subenunciat de C_1 o C_2. Suposem que ho és de C_1 (el mateix raonament val per C_2). Per hipòtesi d'inducció $C_1 \equiv C'$, on C' l'hem obtingut de C_1 substituint A_1 per B_1. Per tant $B = C' \vee C_2$, i per la definició semàntica de $\vee$ es té que $A \equiv B$. *Cas 3:* A és de la forma $(C_1 \wedge C_2)$. A_1 és un subenunciat de C_1 o C_2. Suposem que ho és de C_1 (el mateix raonament val per C_2). Per hipòtesi d'inducció $C_1 \equiv C'$, on C' l'hem obtingut de C_1 substituint A_1 per B_1. Per tant $B = C' \wedge C_2$, i per la definició semàntica de $\wedge$ es té que $A \equiv B$. ■

Aquest teorema dóna un mètode per a demostrar que dos enunciats són lògicament equivalents a partir d'equivalències conegudes.

Exemple 1.8 Si volem demostrar que $P \wedge Q \equiv \neg(\neg P \vee \neg Q)$, partim de $P \wedge Q$ i apliquem la llei de doble negació sobre tot l'enunciat, obtenint $P \wedge Q \equiv \neg\neg(P \wedge Q)$. Després apliquem la llei de Morgan al subenunciat $\neg(P \wedge Q)$ i obtenim $\neg\neg(P \wedge Q) \equiv \neg(\neg P \vee \neg Q)$.

1.7 Forma normal conjuntiva

Donat un enunciat del CP_O volem trobar un enunciat lògicament equivalent que tingui una estructura sintàctica més regular de manera que ens sigui més fàcil poder realitzar demostracions automàticament. En aquest apartat estudiarem la forma normal conjuntiva (FNC) i donarem un algorisme per a transformar un enunciat en la seva FNC.

Definició 1.11 (literal). *Un literal és un àtom o un àtom precedit per un simbol de negació.*

Definició 1.12 (forma normal conjuntiva). *Un enunciat A direm que està en forma normal conjuntiva ssi és de la forma $A_1 \wedge A_2 \wedge \ldots \wedge A_n, n \geq 1$ on cada $A_1, \ldots, A_n$ és una disjunció de literals.*

Qualsevol enunciat el podem expressar en FNC emprant el següent algorisme: *Pas 1.* Eliminar les connectives $\rightarrow$ i $\leftrightarrow$ usant les equivalències:

$$A \leftrightarrow B \equiv (A \rightarrow B) \wedge (B \rightarrow A) \quad (1.1)$$
$$A \rightarrow B \equiv \neg A \vee B \quad (1.2)$$

Pas 2. Reduir al màxim l'àmbit dels símbols de negació usant les següents equivalències tants cops com faci falta:

$$\neg(\neg A) \equiv A \quad (1.3)$$
$$\neg(A \vee B) \equiv \neg A \wedge \neg B \quad (1.4)$$
$$\neg(A \wedge B) \equiv \neg A \vee \neg B \quad (1.5)$$

Pas 3. Obtenir la forma normal conjuntiva usant les següents equivalències tants cops com faci falta:

$$A \vee (B \wedge C) \equiv (A \vee B) \wedge (A \vee C) \quad (1.6)$$
$$A \vee B \equiv B \vee A \quad (1.7)$$
$$A \wedge B \equiv B \wedge A \quad (1.8)$$
$$(A \vee B) \vee C \equiv A \vee (B \vee C) \quad (1.9)$$
$$(A \wedge B) \wedge C \equiv A \wedge (B \wedge C) \quad (1.10)$$

Exemple 1.9 La forma normal conjuntiva de l'enunciat:

$$(\neg P \vee Q) \rightarrow R$$

és

$(\neg P \vee Q) \rightarrow R$	
$\neg(\neg P \vee Q) \vee R$	aplicant (1.2)
$(\neg\neg P \wedge \neg Q) \vee R$	aplicant (1.4)
$(P \wedge \neg Q) \vee R$	aplicant (1.3)
$R \vee (P \wedge \neg Q)$	aplicant (1.7)
$(R \vee P) \wedge (R \vee \neg Q)$	aplicant (1.6)

Noteu que l'algorisme de transformació a FNC preserva l'equivalència lògica entre l'enunciat generat en el pas anterior i el generat en el propi pas, ja que de fet no fem més que aplicar el teorema de substitució. Per tant, un altre mètode de verificar si dos enunciats són lògicament equivalents és comprovar si tenen la mateixa forma normal.

1.8 Decidibilitat

El problema de decidir si un conjunt finit d'enunciats és o no és insatisfactible s'anomena el *problema de la insatisfactibilitat*, i el problema de decidir si un enunciat és o no és conseqüència lògica d'un conjunt finit d'enunciats s'anomena el *problema de la validesa*. De fet, com ja hem vist en aquest capítol, el problema de la validesa es pot reduir a demostrar la insatisfactibilitat d'un conjunt d'enunciats. Aquest tipus de problemes, on les úniques possibles respostes són sí o no, s'anomenen *problemes de decisió*. En el cas que existeixi un procediment efectiu que, en un temps finit, doni una resposta positiva (sí) o negativa (no) a un problema de decisió diem que el problema és *decidible*. Altrament, diem que el problema és *indecidible*. Tant el problema de la insatisfactibilitat com el de la validesa són decidibles per enunciats de la lògica proposicional, ja que per tot conjunt finit d'enunciats podem construir la seva taula de veritat i decidir si és o no insatisfactible. Fent abús de llenguatge, diem que la lògica proposicional és decidible.

Exercicis

Exercici 1 Dissenyar un procediment per decidir si una expressió és un enunciat de la lògica proposicional.

Exercici 2 Eliminar tants parèntesis com sigui possible dels següents enunciats:

1. $(P_1 \to (P_2 \vee (P_3 \to P_4)))$
2. $((P_1 \to P_2) \vee (P_3 \to P_4))$
3. $(P_1 \to ((P_2 \vee P_3) \to P_4))$
4. $((P \to (\neg Q)) \wedge R)$
5. $((P \vee (\neg Q)) \vee (P \vee R))$

Exercici 3 Recuperar els parèntesis dels següents enunciats:

1. $P_1 \vee \neg P_2 \wedge P_3$
2. $P_1 \to \neg(P_1 \vee P_2) \wedge P_3 \leftrightarrow P_4$
3. $P \vee Q \to R \to P$
4. $P \to \neg Q \to R$

Exercici 4 Representar els següents enunciats declaratius com enunciats del CP_0.

1. Si dius mentides et cauran les dents.
2. Aquest dentrific evita la càries i elimina el sarro si et raspalles les dents després de cada menjada.
3. Si vols arrancar-te un queixal amb anestèsia vés al dentista, sinó vés al barber.
4. Només ens recordem de Santa Bàrbara quan trona.
5. En Joan i Na Margarida s'estimen. Si en Joan no estimés a na Margarida, ni ell cantaria òpera ni ella tocaria el piano.
6. En Joan i na Margarida aniran de vacances a Pollensa o a Cadaqués. Si van a Pollensa ens duran una ensaimada.
7. El preu de les hipoteques es manté encara que el tipus d'interés interbancari baixa.
8. En Carles comprarà un pis sol si baixen les hipoteques.
9. No es necessari tenir avaladors per aconseguir les noves hipoteques.
10. L'habitació del primer pis té molta llum, però la del segón no té finestra.

Exercici 5 Construir la taula de veritat pels següents enunciats:

1. $((P_1 \to P_2) \to P_2)$
2. $\neg((P_1 \to P_2) \to (\neg(P_2 \to P_1)))$
3. $((P_1 \wedge P_2) \vee ((P_3 \wedge P_4) \to P_1))$
4. $(((\neg P_1) \wedge P_2) \to ((\neg P_2) \wedge P_3))$

Exercici 6 Trobar un model per a l'enunciat $(P \to (Q \vee R))$.

Exercici 7 Demostrar que els següents enunciats són tautologies:

1. $\neg(P \wedge \neg P)$
2. $\neg\neg P \rightarrow P$
3. $\neg P \rightarrow (P \rightarrow Q)$
4. $((P \rightarrow Q) \rightarrow P) \rightarrow P$
5. $((P \rightarrow Q) \wedge P) \rightarrow P \wedge Q$
6. $P \rightarrow (Q \rightarrow (R \rightarrow Q))$

Exercici 8 Un conjunt d'enunciats és minimalment insatisfactible ssi és insatisfactible i tots els seus subcojunts propis són satisfactibles. Determinar quins dels següents conjunts són minimalment insatisfactibles:

1. $\{P \rightarrow \neg Q, P \rightarrow \neg R, P \wedge Q, R\}$
2. $\{P \rightarrow \neg Q, P \rightarrow \neg R, Q \vee R, P\}$
3. $\{P \rightarrow Q, P \rightarrow R, Q \wedge R, \neg P\}$

Exercici 9 De quin tipus d'enunciats creus que n'hi ha més, de vàlids o d'insatisfactibles?

Exercici 10 Demostrar les següents conseqüències lògiques:

1. $P, P \rightarrow Q \models P \wedge Q$
2. $Q, P \rightarrow (Q \rightarrow R) \models P \rightarrow R$
3. $(P \vee Q) \wedge R \models (P \wedge R) \vee (Q \wedge R)$
4. $P \rightarrow (Q \rightarrow \neg P) \models P \rightarrow \neg Q$
5. $(P \rightarrow Q) \rightarrow (P \rightarrow R) \models P \rightarrow (Q \rightarrow R)$

Exercici 11 El carnisser no portava fang a la sabata. Si el carnisser hagués estat l'assassí hauria saltat per la finestra, i si hagués saltat per la finestra, s'hauria embrutat les sabates de fang. És el carnisser l'assassí? ...

Exercici 12 L'Anna o la Montse aniran a la festa que organitzen els estudiants d'Informàtica. Si hi va l'Anna, el Carles o el David hi aniran, i si hi va el Carles l'Eva i la Carme també hi aniran. Però si el David no hi va la Carme no hi anira si hi va la Eva. Hi anira la Montse si no hi va el David ? ...

Exercici 13 El curs passat varen succeir diferents robatoris a l'aula d'informàtica: 2 Mb de memòria, un ratolí, una disquetera, ... Donada la gravetat i freqüència d'aquesta mena de fets, el director va decidir contractar els serveis d'un detectiu. De moment s'han trobat quatre sospitosos, als quals ens referirem per A, B, C i D. Les conclusions de la investigació són les següents:

1. Si A és culpable llavors B és còmplice.
2. Si B és culpable, C és còmplice o A és innocent.

3. Si D és innocent, A és culpable i C és innocent
4. Si D és culpable, A també ho és.

D'aquests quatre sospitosos, dir quins són innocents i quins són culpables o còmplices.

Exercici 14 Digueu si els següents fets són certs o falsos, raonant la resposta:

1. Si A és una tautologia, llavors per cada conjunt d'enuncitas Γ es té que $\Gamma \models A$.
2. Si Γ és insatisfactible llavors per qualsevol enunciat A es té que $\Gamma \models A$.
3. Si $\Gamma \models A$, llavors existeix almenys una interpretació que és model de Γ i de A.
4. Per a qualsevol enunciat A i conjunt d'enunciats Γ, $\Gamma \not\models A$ si i només si $\Gamma \models \neg A$.

Exercici 15 La Sra. Antonieta té tres netes: Anna, Eva i Mireia. Dissabte pasat, mentre les guardava, se li van cruspir les pastes del te. Sota l'amenaça de retirar-los-hi l'assignació econòmica setmanal, varen fer les següents declaracions:

- Anna: Ha estat l'Eva, la Mireia no en té cap culpa.
- Eva: La Mireia és culpable sols si l'Anna ho és.
- Mireia: Jo no me les he menjat, ha estat alguna d'elles.

1. Si suposem que cap neta menteix, quines són innocents i quines són culpables?.
2. Si totes són innocents, quines han enganyat l'avia?.
3. Si la que és innocent diu la veritat i la que és culpable menteix, quines són innocents i quines són culpables?.

Exercici 16 Demostrar les següents equivalències lògiques:

1. $P \vee Q \equiv \neg(\neg P \wedge \neg Q)$
2. $(P \to Q) \to Q \equiv P \vee Q$
3. $P \to (Q \to R) \equiv (P \wedge Q) \to R$
4. $(P \to Q) \wedge (P \to R) \equiv (P \to Q \wedge R)$

Exercici 17 Definir les connectives lògiques $\vee$, $\wedge$ i $\leftrightarrow$ en funció de les connectives $\neg$ i $\to$.

Exercici 18 Expressar en forma normal conjuntiva els següents enunciats:

1. $(P \leftrightarrow \neg Q)$
2. $\neg(P \vee Q) \to R$

3. $\neg(P \to Q) \vee (P \vee Q)$
4. $(((P \to Q) \to R) \to S)$
5. $(P \wedge Q \wedge R) \vee (\neg P \wedge \neg Q \wedge R)$

Exercici 19 Demostrar les següents equivalències lògiques transformant l'enunciat a la dreta i l'enunciat a l'esquerra del símbol d'equivalència en la mateixa formal normal.

1. $P \vee Q \equiv \neg(\neg P \wedge \neg Q)$
2. $(P \to Q) \to Q \equiv P \vee Q$
3. $P \to (Q \to R) \equiv (P \wedge Q) \to R$
4. $(P \to Q) \wedge (P \to R) \equiv (P \to Q \wedge R)$

Exercici 20 Demostrar que per a cada enunciat A existeix un enunciat lògicament equivalent en forma normal conjuntiva (fer-ho per inducció).

Capítol 2

Procediments de prova proposicionals

2.1 Introducció

Hem vist al capítol 1 que les taules de veritat són un procediment de decisió per resoldre el problema de la validesa i el problema de la insatisfactibilitat de la lògica proposicional. No obstant, construir taules de veritat és computacionalment costós, ja que si el nombre d'àtoms diferents que apareixen a un enunciat és n, hem de construir una taula amb 2^n files. Per altra banda, les taules no són molt intuitives, en el sentit que no reflexen la forma de raonar dels humans. En aquest capítol estudiarem altres procediments de decisió, que anomenarem *procediments de prova*, que permeten solucionar els problemes de la validesa i de la insatisfactibilitat emprant el concepte de *prova* en lloc del d'interpretació. Fins ara per a comprovar si $\Gamma \models A$, miràvem si totes les interpretacions que satisfan Γ també satisfan A. A partir d'ara per a comprovar si $\Gamma \models A$ mirarem si existeix una prova de A a partir de Γ. Per entendre què és una prova hem de parlar primer del concepte *càlcul*. Un càlcul per un llenguatge està format per un conjunt de regles d'inferència. Les regles d'inferència formalitzen esquemes de raonaments vàlids i permeten derivar enunciats a partir d'altres enunciats del llenguatge. Un exemple de regla d'inferència és el *modus ponens*:

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

Aquesta regla d'inferència estableix que donat un enunciat A i un enunciat de la forma $A \to B$ podem derivar l'enunciat B; per exemple, si tenim l'enunciat $P \vee Q$ i l'enunciat $P \vee Q \to R$ podem derivar l'enunciat R. Com les regles d'inferència formalitzen esquemes de raonaments vàlids es clar que $P \vee Q \to R, P \vee Q \models R$. Noteu que no hem emprat més que la regla sense necessitat de construir cap taula. Un altre exemple de regla d'inferència és la regla *d'introducció de la disjunció*:

$$\frac{A}{A \vee B}$$

Aquesta regla d'inferència estableix que donat un enunciat A podem derivar l'enunciat $A \vee B$, on B és qualsevol enunciat del llenguatge; per exemple, si

tenim l'enunciat P podem derivar l'enunciat $P \vee Q$. Com les regles formalitzen esquemes de raonaments vàlids es clar que $P \models P \vee Q$. Noteu que no hem emprat més que la regla sense necessitat de construir cap taula. Si partint d'un conjunt d'enunciats Γ, aplicant regles d'inferència, podem derivar l'enunciat A, direm que hem trobat una *prova* de A a partir de Γ i ho notarem $\Gamma \vdash A$. $\Gamma \vdash A$ es llegeix A es dedueix de Γ. La successió d'enunciats generats fins arribar a A diem que és una prova de $\Gamma \vdash A$. Sabem que comprovar que $\Gamma \models A$ és equivalent a comprovar $\Gamma \cup \{\neg A\}$ és insatisfactible. Quan partim d'un conjunt d'enunciats $\Gamma \cup \{\neg A\}$ i aplicant regles d'inferència derivem el conjunt buit, diem que hem trobat una prova per refutació de $\Gamma \cup \{\neg A\}$ i ho notem $\Gamma \vdash A$. Ara bé, la *relació de satisfactibilitat* ($\models$) l'hem definida en funció del concepte d'interpretació, mentre que la *relació de deductibilitat* ($\vdash$) l'hem definida en funció de la derivació d'enunciats aplicant regles d'inferència. Perquè un càlcul ens sigui útil per a comprovar si $\Gamma \models A$ ens interessarà que tingui les següents propietats:

1. **Solidesa** Que tot enunciat A deduible d'un conjunt d'enunciats Γ sigui una conseqüència lògica de Γ. O sigui, $\Gamma \vdash A \Rightarrow \Gamma \models A$. Aquesta propietat s'exigeix a tots els càlculs, ja que sinó podriem deduir enunciats que no fossin conseqüències lògiques.

2. **Completesa** Que per a tot enunciat A que és una conseqüència lògica d'un conjunt d'enunciats Γ existeix una prova de A a partir de Γ. O sigui, $\Gamma \models A \Rightarrow \Gamma \vdash A$.

En aquest capítol estudiarem dos càlculs: el dels tableaux semàntics i el de resolució. Aquests càlculs permeten definir uns procediments de prova fàcils de programar. Per cadascun, donarem un procediment per il.lustrar com es pot comprovar automàticament l'existència d'una prova. Tots dos són procediments de prova per refutació.

2.2 Tableaux semàntics

En aquesta secció estudiem el càlcul dels tableaux semàntics, demostrem la seva solidesa i completesa a l'hora de fer demostracions per refutació i donem un procediment de prova per il.lustrar com automatizar-lo. Per introduir els tableaux semàntics, comencem amb l'exemple següent:

Exemple 2.1 Sigui $\Gamma = \{P \wedge \neg R, \neg P \vee Q, R \vee \neg Q\}$ un conjunt d'enunciats del qual volem esbrinar si és satisfactible. Resoldrem aquest problema reduint-lo a demostrar la satisfactibilitat d'altres conjunts d'enunciats pels quals és més senzill decidir sobre la seva satisfactibilitat. Per la definició semàntica de la conjunció es té que una interpretació $\mathcal{I}$ és model de $P \wedge \neg R$ ssi $\mathcal{I}$ és model de P i de $\neg R$. Per tant, Γ és satisfactible ssi $\Gamma_1 = \Gamma \cup \{P, \neg R\} = \{P \wedge \neg R, \neg P \vee Q, R \vee \neg Q, P, \neg R\}$ és satisfactible. Per la definició semàntica de la disjunció es té que una interpretació $\mathcal{I}$ és model de $\neg P \vee Q$ ssi $\mathcal{I}$ és model de $\neg P$ o de Q. Per tant, Γ_1 és satisfactible ssi $\Gamma_{21} = \Gamma_1 \cup \{\neg P\} = \{P \wedge \neg R, \neg P \vee Q, R \vee \neg Q, P, \neg R, \neg P\}$ o $\Gamma_{22} = \Gamma_1 \cup \{Q\} = \{P \wedge \neg R, \neg P \vee Q, R \vee \neg Q, P, \neg R, Q\}$ és satisfactible. Γ_{21} és insatisfactible, ja que no existeix cap interpretació que pugui ser a la vegada model de P i $\neg P$. Per tant, Γ_1 és satisfactible ssi Γ_{22} és satisfactible. Aplicant el mateix raonament que ens ha permés obtenir Γ_{21} i Γ_{22} a partir de Γ_1, es té que

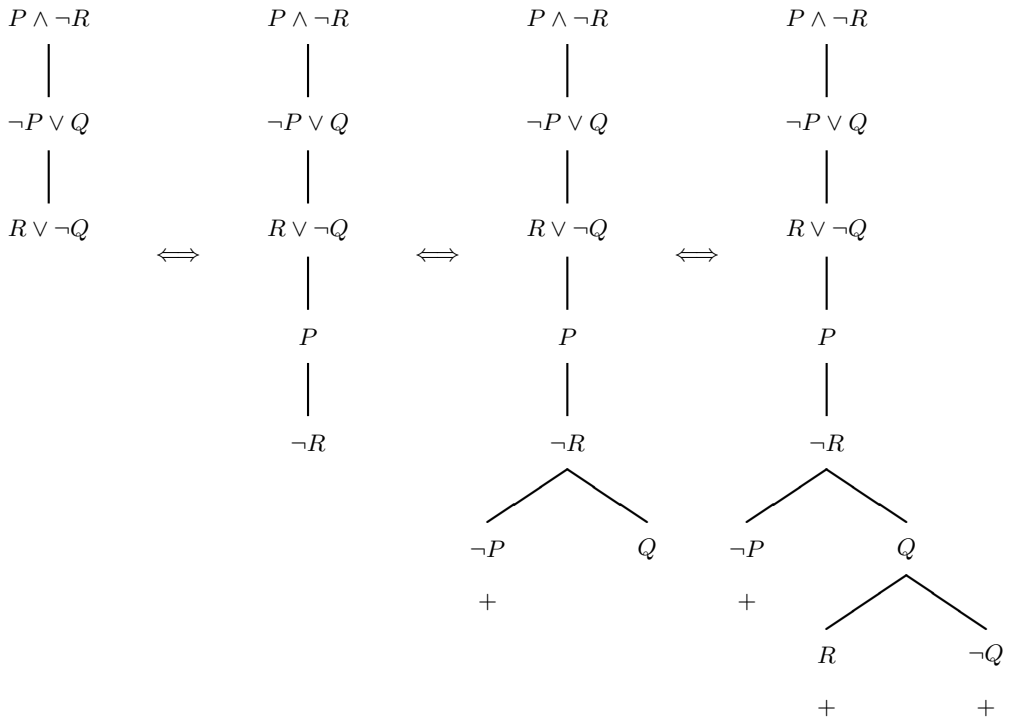


Figura 2.1: Prova per tableaux semàntics

Γ_{22} és satisfactible ssi $\Gamma_{31} = \Gamma_{22} \cup \{R\} = \{P \wedge \neg R, \neg P \vee Q, R \vee \neg Q, P, \neg R, Q, R\}$ o $\Gamma_{32} = \Gamma_{22} \cup \{\neg Q\} = \{P \wedge \neg R, \neg P \vee Q, R \vee \neg Q, P, \neg R, , Q, \neg Q\}$ és satisfactible. Γ_{31} i Γ_{32} són insatisfactibles per contenir un àtom i la seva negació. Per tant, tirant enrera en el raonament seguit, podem concloure que Γ és insatisfactible.

El raonament seguit a l'exemple 2.1 el podem representar gràficament com una seqüència d'arbres com es mostra a la figura 2.1. Abans de comentar com obtenir aquesta seqüència donem la definició següent.

Definició 2.1 (arbres). *Un arbre d'enunciats és un arbre binari ordenat on cada node té per etiqueta un enunciat. Una branca d'un arbre d'enunciats és un camí des de l'arrel a una fulla. Una branca b d'un arbre d'enunciats és tancada ssi a b apareix un enunciat A i la seva negació $\neg A$. Una branca oberta és una branca que no és tancada. Un arbre d'enunciats és tancat ssi totes les seves branques són tancades.*

Tot seguit veurem com obtenir la seqüència d'arbres binaris que constitueix una prova de la insatisfactibilitat del conjunt Γ de l'exercici 2.1. Partim d'un arbre format per una sola branca que conté els enunciats de Γ. La obtenció de Γ_1 a partir de Γ la representem afegint els nodes P i $\neg R$ a la branca anterior, obtenint el segon arbre de la figura 2.1. Noteu que aquest arbre conté els mateixos enunciats que Γ_1. La obtenció de Γ_{21} i Γ_{22} a partir de Γ_1, la representem

afegint com a fills del node terminal del segon arbre els nodes $\neg P$ i Q, obtenint el tercer arbre de la seqüència. Els enunciats de la branca de l'esquerra són els mateixos que els de Γ_{21} i els enunciats de la branca de la dreta són els mateixos que els de Γ_{22}. Com la branca de més a l'esquerra conté un enunciat i la seva negació ho marquem amb el signe + i diem que hem tancat la branca. D'aquesta forma representem el fet que Γ_{21} sigui insatisfactible. La obtenció de Γ_{31} i Γ_{32} a partir de Γ_{22} la representem afegint els nodes R i $\neg Q$ com a fills del node terminal de la branca de la dreta del tercer arbre, obtenint el darrer arbre de la seqüència. Com hem obtingut un arbre on totes les branques són tancades, diem que hem obtingut una prova per tableaux semàntics de la insatisfactibilitat de Γ.

A l'exemple 2.1 hem vist com podem demostrar la insatisfactibilitat d'un conjunt d'enunciats quan tenim conjuncions o disjuncions, però no hem considerat la resta de connectives. El mètode dels tableaux semàntics es basa en la idea de que tot enunciat que no sigui un literal sempre es pot expressar com la conjunció o disjunció de dos enunciats més simples (excepte el cas $\neg\neg A$, on A és un enunciat). L'enunciat inicial és satisfactible ssi ho és la seva expressió com una disjunció o conjunció. Per expressar un enunciat com una conjunció de dos enunciats utilitzarem les equivalències lògiques següents:

$$\begin{aligned} A \wedge B &\equiv A \wedge B \\ \neg(A \vee B) &\equiv \neg A \wedge \neg B \\ \neg(A \rightarrow B) &\equiv A \wedge \neg B \\ A \leftrightarrow B &\equiv (A \rightarrow B) \wedge (B \rightarrow A) \end{aligned}$$

Els enunciats de la forma $A \wedge B, \neg(A \vee B), \neg(A \rightarrow B), A \leftrightarrow B$ s'anomenen *enunciats de tipus* α. Si $\alpha \equiv \alpha_1 \wedge \alpha_2$, α_1 nota el conjuntant de l'esquerra i α_2 nota el conjuntant de la dreta. Per expressar un enunciat com una disjunció de dos enunciats utilitzarem les equivalències lògiques següents:

$$\begin{aligned} A \vee B &\equiv A \vee B \\ \neg(A \wedge B) &\equiv \neg A \vee \neg B \\ A \rightarrow B &\equiv \neg A \vee B \\ \neg(A \leftrightarrow B) &\equiv \neg(A \rightarrow B) \vee \neg(B \rightarrow A) \end{aligned}$$

Els enunciats de la forma $A \vee B, \neg(A \wedge B), A \rightarrow B, \neg(A \leftrightarrow B)$ s'anomenen *enunciats de tipus* β. Si $\beta \equiv \beta_1 \vee \beta_2$, β_1 nota el disjuntant de l'esquerra i β_2 nota el disjuntant de la dreta. Un enunciat doblement negat $\neg\neg A$ el podem expressar com A fent ús de l'equivalència lògica $\neg\neg A \equiv A$. Un enunciat de la forma $\neg\neg A$ direm que és un *enunciat de tipus* σ, i l'enunciat A que resulta d'eliminar la doble negació el notarem σ_1. L'obtenció d'una prova per tableaux semàntics es pot representar gràficament com una seqüència d'arbres binaris als que anomenarem tableaux. Si obtenim un tableau on totes les seves branques són tancades, tenim una prova de que Γ és insatisfactible. La satisfactibilitat de Γ sols la podrem garantir quan quedi alguna branca oberta on ja no podem descomposar els enunciats amb enunciats més simples. Tot seguit definim més formalment com construir un tableau.

Definició 2.2 (tableau semàntic). *Un tableau per un conjunt finit d'enunciats* $\Gamma = \{A_1, A_2, \ldots, A_n\}$ *és un arbre d'enunciats que es pot construir en un nombre finit de passos mitjançant les regles següents:*

- *Regla d'inicialització (R_{ini}): un arbre d'enunciats format per una sola branca amb n nodes etiquetats amb els enunciats de Γ és un tableau per a Γ, anomenat tableau inicial.*

$$\begin{array}{c} A_1 \\ | \\ A_2 \\ | \\ \vdots \\ | \\ A_n \end{array}$$

- *Regla α: Sigui T un tableau per Γ amb una branca oberta b que conté un node amb un enunciat de tipus α. Si la fulla de b és f, l'arbre d'enunciats T' que s'obté afegint α_1 com a successor de f i α_2 com a successor de α_1 és un tableau per Γ.*

- *Regla β: Sigui T un tableau per Γ amb una branca oberta b que conté un node amb una enunciat de tipus β. Si la fulla de b és f, l'arbre d'enunciats T' que s'obté afegint β_1 com a successor esquerra de f i β_2 com a successor dreta de f és un tableau per Γ.*

- *Regla σ: Sigui T un tableau per Γ amb una branca oberta b que conté un node amb un enunciat de tipus σ. Si la fulla de b és f, l'arbre d'enunciats T' que s'obté afegint σ_1 com a successor de f és un tableau per Γ.*

Un cop definit el concepte de tableau, definim els conceptes d'extensió directa d'un tableau i de prova.

Definició 2.3 (extensió directa d'un tableau). *Un tableau T' és una extensió directa d'un tableau T ssi T' s'ha obtingut a partir de T aplicant un cop la regla α, la regla β o la regla σ de construcció de tableaux.*

Definició 2.4 (prova). *Direm que una sequència finita de tableaux $T_0, \ldots, T_n$ és una prova per refutació de Γ, i ho notarem $\Gamma \vdash_{tb}$, ssi*

- *T_0 és un tableau inicial per Γ;*

- *T_i, $0 < i \leq n$, és una extensió directa de T_{i-1};*

- *T_n és un tableau tancat.*

Exemple 2.2 La figura 2.2 mostra una prova de la insatisfactibilitat de $\Gamma = \{P \to Q, \neg(S \to Q), P\}$.

De vegades, per simplificar les coses, en lloc de representar tota la seqüència, la colapsem en un mateix gràfic. En aquest cas, per una prova entendrem l'últim arbre de la seqüència, i si es possible construir un tableau tancat T pel conjunt d'enunciats Γ direm que T és una prova per refutació de Γ i ho notarem $\Gamma \vdash_{tb}$ tableau.

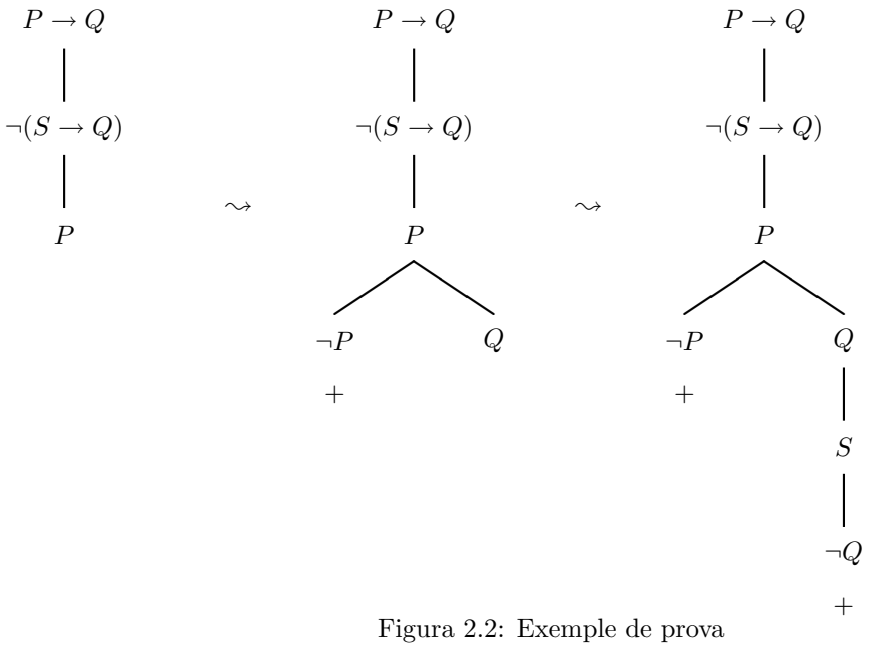


Figura 2.2: Exemple de prova

2.2.1 Solidesa i completesa

En aquesta secció demostrem les propietats de solidesa i completesa pel càlcul dels tableaux semàntics. Aquestes propietats són les següents:

- **Solidesa:** Si T és un tableau tancat per un conjunt d'enunciats Γ, llavors Γ és insatisfactible.
- **Completesa:** Si Γ és un conjunt d'enunciats insatisfactible, llavors existeix un tableau tancat per Γ.

En primer lloc demostrarem la solidesa, però abans definirem model d'una branca i model d'un tableau, i donarem un lema que emprarem a la demostració de solidesa.

Definició 2.5 (model d'una branca). *Sigui T un tableau i $\mathcal{I}$ una interpretació. Direm que $\mathcal{I}$ satisfà una branca b de T ($\mathcal{I} \models b$), o que $\mathcal{I}$ és un model de b, ssi $\mathcal{I}$ satisfà tots els enunciats associats als nodes de b.*

Definició 2.6 (model d'un tableau). *Sigui T un tableau i $\mathcal{I}$ una interpretació. Direm que $\mathcal{I}$ satisfà un tableau T ($\mathcal{I} \models T$), o que $\mathcal{I}$ és un model de T, ssi $\mathcal{I}$ satisfà almenys una branca de T.*

Lema 2.1 *Sigui T un tableau, $\mathcal{I}$ una interpretació tal que $\mathcal{I} \models T$ i T' un tableau que és una extensió directa de T. Si $\mathcal{I} \models T$ llavors $\mathcal{I} \models T'$.*

Demostració: Com $\mathcal{I} \models T$, existeix almenys una branca b de T tal que $\mathcal{I} \models b$. T' s'ha obtingut de T afegint successors o bifurcant alguna branca b_1 de T. Si $b_1 \neq b$, llavors b segueix essent una branca de T' i, per tant, $\mathcal{I} \models T'$. Si $b_1 = b$ distingim els casos següents:

1. T' s'ha obtingut de T aplicant la regla α a un enunciat A (de tipus α) de la branca b. O sigui, s'ha substituit la branca b per una branca b' resultant d'afegir a b els constituents α_1 i α_2 de l'enunciat A. Com $\mathcal{I} \models A$ i $A \equiv \alpha_1 \wedge \alpha_2$, es té que $\mathcal{I} \models \alpha_1$ i $\mathcal{I} \models \alpha_2$. Per tant, $\mathcal{I} \models b'$.

2. T' s'ha obtingut de T aplicant la regla β a un enunciat A (de tipus β) de la branca b. O sigui, s'ha substituit la branca b per dues branques b'_1 (resultant d'afegir a b el constituent β_1) i b'_2 (resultant d'afegir a b el constituent β_2). Com $\mathcal{I} \models A$ i $A \equiv \beta_1 \vee \beta_2$, es té que $\mathcal{I} \models \beta_1$ o $\mathcal{I} \models \beta_2$. Per tant, $\mathcal{I} \models b'_1$ o $\mathcal{I} \models b'_2$.

3. T' s'ha obtingut de T aplicant la regla σ a un enunciat A (de tipus σ) de la branca b. O sigui, s'ha substituit la branca b per una branca b' resultant d'afegir a b el constituent σ_1 de l'enunciat A. Com $\mathcal{I} \models A$ i $A \equiv \sigma_1$, es té que $\mathcal{I} \models \sigma_1$. Per tant, $\mathcal{I} \models b'$.

Com en els tres casos considerats $\mathcal{I}$ satisfà alguna branca de T', es té que $\mathcal{I} \models T'$. ■

Teorema 2.1 (solidesa). *Si T és un tableau tancat per un conjunt d'enunciats Γ llavors Γ és insatisfactible.*

Demostració: T s'ha construit per una seqüència de tableaux

$$T_0, T_1, \ldots, T_{i-1}, T_i, \ldots, T_n = T, n \geq 0$$

on T_0 és un tableau inicial per a Γ i cada T_i $(i > 0)$ és una extensió directa de T_{i-1}. Suposem que Γ sigui satisfactible. Sigui $\mathcal{I}$ un model de Γ. Demostrarem per inducció sobre i que $\mathcal{I} \models T_i$ per tot $0 \leq i \leq n$. *Pas base:* $\mathcal{I} \models T_0$, ja que Γ és satisfactible i T_0 sols conté els enunciats de Γ. *Pas inducció:* Aceptem com a hipòtesi d'inducció que $\mathcal{I} \models T_{i-1}$ i provarem que $\mathcal{I} \models T_i$. Com T_i és una extensió directa de T_{i-1}, pel lema anterior es té que $\mathcal{I} \models T_i$. Per tant, $\mathcal{I} \models T_i$ per tot $0 \leq i \leq n$. En particular, $\mathcal{I} \models T_n = T$; però això es una contradicció ja que T és un tableau tancat. Per tant Γ es insatisfactible. ■

Un cop demostrada la solidesa anem a demostrar la completesa. Com de costum, primer donarem alguns lemes i definicions.

Definició 2.7 (complexitat d'un enunciat). *Siguin A i B enunciats. La complexitat d'un enunciat $\|A\|$ es defineix per recursió sobre l'estructura de A de la següent manera:*

- $\|A\| = 0$ *si A és un àtom*
- $\|\neg A\| = \|A\| + 1$
- $\|A \wedge B\| = \|A\| + \|B\| + 2$
- $\|A \vee B\| = \|A\| + \|B\| + 2$
- $\|A \rightarrow B\| = \|A\| + \|B\| + 2$
- $\|A \leftrightarrow B\| = 2(\|A\| + \|B\|) + 6$

Lema 2.2 (reducció de complexitat).

1. *Per tot enunciat de tipus α es verifica que $\|\alpha\| > \|\alpha_1\| + \|\alpha_2\|$.*
2. *Per tot enunciat de tipus β es verifica que $\|\beta\| > \|\beta_1\| + \|\beta_2\|$.*
3. *Per tot enunciat de tipus σ es verifica que $\|\sigma\| > \|\sigma_1\|$.*

Definició 2.8 (branca completa). *Sigui α un enunciat de tipus α, on α_1, α_2 són els seus conjuntants; β un enunciat de tipus β, on β_1, β_2 són els seus disjuntants; i σ un enunciat de tipus σ, on σ_1 és la seva simplificació. Una branca b direm que és completa ssi compleix les condicions següents:*

1. *si $\alpha \in b$ llavors $\alpha_1, \alpha_2 \in b$;*
2. *si $\beta \in b$ llavors $\beta_1 \in b$ o $\beta_2 \in b$;*
3. *si $\sigma \in b$ llavors $\sigma_1 \in b$.*

Definició 2.9 (tableau acabat). *Un tableau T direm que és acabat ssi totes les branques són tancades o té alguna branca oberta i completa.*

Definició 2.10 (conjunt de Hintikka). *Un conjunt d'enunciats Γ direm que és un conjunt de Hintikka ssi compleix les condicions següents:*

1. Γ *no conté un àtom* p *i la seva negació* $\neg p$*;*
2. *si* Γ *conté un enunciat de tipus* α *llavors* $\alpha_1, \alpha_2 \in \Gamma$*;*
3. *si* Γ *conté un enunciat de tipus* β *llavors* $\beta_1 \in \Gamma$ *o* $\beta_2 \in \Gamma$*;*
4. *si* Γ *conté un enunciat de tipus* σ *llavors* $\sigma_1 \in \Gamma$*.*

Lema 2.3 (lema de Hintikka). *Si un conjunt d'enunciats* Γ *és un conjunt de Hintikka llavors* Γ *és satisfactible.*

Demostració: Demostrarem que la següent interpretació satisfà qualsevol enunciat $A \in \Gamma$.

$$\mathcal{I}(p) = \begin{cases} 1 & \text{si } p \in \Gamma \\ 0 & \text{si } \neg p \in \Gamma \\ 1 & \text{si } p \notin \Gamma \text{ i } \neg p \notin \Gamma \end{cases}$$

La demostració la farem per inducció completa sobre la complexitat de A. *Pas base:* $\|A\| \leq 1$ Si $A \in \Gamma$ i $\|A\| \leq 1$ es té que A és un àtom p o la seva negació $\neg p$, i com no poden apareixer tots dos a Γ per ser un conjunt de Hintikka es té que $\mathcal{I} \models A$. *Pas inducció:* $\|A\| \geq 2$ Aceptem com a hipòtesi d'inducció que si $B \in \Gamma$ i $\|B\| < \|A\|$ llavors $\mathcal{I} \models B$. Distingim tres casos:

1. A és de tipus α, per tant $\alpha_1, \alpha_2 \in \Gamma$. Per hipòtesi d'inducció $\mathcal{I} \models \alpha_1$ i $\mathcal{I} \models \alpha_2$. Per la definició semàntica de la conjunció es té que $\mathcal{I} \models \alpha_1 \wedge \alpha_2$. Sabem que $\alpha \equiv \alpha_1 \wedge \alpha_2$. Per tant, $\mathcal{I} \models \alpha$
2. A és de tipus β, per tant $\beta_1 \in \Gamma$ o $\beta_2 \in \Gamma$. Per hipòtesi d'inducció $\mathcal{I} \models \beta_1$ o $\mathcal{I} \models \beta_2$. Per la definició semàntica de la disjunció es té que $\mathcal{I} \models \beta_1 \vee \beta_2$. Sabem que $\beta \equiv \beta_1 \vee \beta_2$. Per tant, $\mathcal{I} \models \beta$
3. A és de tipus σ, per tant $\sigma_1 \in \Gamma$. Per hipòtesi d'inducció $\mathcal{I} \models \sigma_1$. Sabem que $\sigma \equiv \sigma_1$. Per tant, $\mathcal{I} \models \sigma$.

■

Teorema 2.2 *Si* b *és una branca oberta i completa d'un tableau* T *llavors* T *és satisfactible.*

Demostració: Una branca oberta i completa compleix les condicions de conjunt de Hintikka. Pel lema anterior sabem que existeix una interpretació $\mathcal{I}$ tal que $\mathcal{I} \models b$. Per tant, $\mathcal{I} \models T$. ■

Teorema 2.3 (completesa). *Si* Γ *és insatisfactible, llavors tot tableau* T *acabat per* Γ *és tancat.*

Demostració: Suposem que Γ és insatisfactible i T un tableau acabat i obert. Per tant, T tindrà una branca completa i oberta. Pel teorema anterior tindrem que Γ és satisfactible, contra la hipòtesi. ■

Noteu que hem demostrat una propietat més forta que la que hem enunciat al principi de la secció, ja que no sols hem demostrat que per tot conjunt d'enunciats Γ insatisfactible existeix un tableau tancat, sinó que a més a més hem demostrat que qualsevol tableau acabat per Γ és tancat.

2.2.2 Procediments de demostració de tableaux

En aquesta secció estudiem un procediment de demostració de tableaux, i comentem algunes millores que ens permetran construir procediments més eficients. L'objectiu del procediment és determinar de manera automàtica la satisfactibilitat d'un conjunt d'enunciats Γ

Procediment tableaux
Entrada: Γ (conjunt d'enunciats)
Sortida: "satisfactible" o "insatisfactible"

inici
$T \leftarrow$ tableau- inicial(Γ)
mentre no acabat(T) **fer**
una extensió directa de T
$T \leftarrow$ extensió-directa(T)
fmentre
si tancat(T)
llavors
tornar("Γ és insatisfactible")
sino
tornar(" Γ és satisfactible")
fsi
finici

En aquest programa trobem dues fonts d'indeterminisme:

- branca a expandir;
- enunciat a expandir de la branca seleccionada.

Procedimient tableaux
Entrada: llista de llistes d'enunciats
Sortida: "satisfactible" o "insatisfactible"
inici
$L1 \leftarrow$ cap(lista)
$L2 \leftarrow$ cua(lista)
si buida(llista) **llavors tornar**(" insatisfactible")
si tancada(L1) **llavors tornar**(tableaux(L2))
si existeix $A \in L1$ tal que **no** marcat(A)
llavors
marcar(A)
opció tipus(A)
literal: **tornar**(tableaux(llista))
σ: **tornar**(tableaux($[\mathrm{L1} * [\sigma_1]] * \mathrm{L2}$))
α: **tornar**(tableaux($[\mathrm{L1} * [\alpha_1, \alpha_2]] * \mathrm{L2}$))
β: **tornar**(tableaux($[\mathrm{L1} * [\beta_1], \mathrm{L1} * [\beta_2]] * \mathrm{L2}$))
fopció
sino
tornar("satisfactible")
fsi
finici

Cada enunciat es representa com una estructura amb dos camps, el primer conté l'enunciat i el segón és un camp booleà que inicialment pren el valor fals. El procediment marcar posa a vertader aquest camp. Les funcions buida,tancada i marcat són booleanes; buida retorna vertader si el seu argument és una llista buida, tancada retorna vertader si el seu argument és una llista on apareix un enunciat i la seva negació, i marcat retorna vertader si el seu argument és un enunciat que ha estat marcat pel procediment marcar. Les funcions cap i cua prenen com argument una llista; cap retorna el primer element de la llista i cua retorna la llista que resulta al treure el primer element. La funció $*$ és la concatenació de llistes, per exemple, $[A, B] * [C] = [A, B, C]$, $[A, B] * [[C, D], [E, F, G]] = [A, B, [C, D], [E, F, G]]$ i $[[A, B]] * [[C, D], [E, F, G]] = [[A, B], [C, D], [E, F, G]]$. Cal observar que, inicialment, l'entrada al procediment és una llista que té un únic element que és a la vegada una llista.

2.3 Resolució

En aquesta secció estudiem el càlcul de Resolució. Aquest càlcul es caracteritza per estar format per una sola regla d'inferència anomenada Principi de Resolució. Per poder aplicar el principi de resolució, cal transformar els enunciats a forma clausal.

2.3.1 Forma Clausal

La forma clausal la podem veure com una representació compacta de la forma normal conjuntiva (FNC). Recordem que donat un enunciat qualsevol del CP_0 el podem transformar en un enunciat lògicament equivalent en FNC. Tot seguit donarem unes quantes definicions i estudiarem la transformació de FNC a forma clausal.

Definició 2.11 (dual d'un literal). *Si L és un literal idèntic a un àtom P llavors el dual de L - notat L^d - és igual a $\neg P$. Altrament, si L és de la forma $\neg P$ llavors $L^d = P$.*

Definició 2.12 (clàusula). *Una clàusula és un conjunt finit de zero o més literals que representa la disjunció d'aquests literals.*

Exemple 2.3 La disjunció $P \vee Q \vee \neg R$ la podem representar amb la clàusula $\{P, Q, \neg R\}$.

Direm que una clàusula és unitària si conté un únic literal. Les clàusules $\{P\}$ i $\{\neg Q\}$ són unitàries. Una clàusula que no conté cap literal direm que és la clàusula buida i la notarem per $\Box$. Com la clàusula buida no té cap literal que pugui ser satisfet per una interpretació, és sempre insatisfactible i denota el valor de veritat F.

Definició 2.13 (forma clausal). *Sigui $A = (L_{1,1} \vee \ldots \vee L_{1,n_1}) \wedge \ldots \wedge (L_{k,1} \vee \ldots \vee L_{k,n_k})$ un enunciat en FNC on els $L_{i,j}$ són literals. La representació en forma clausal de A vé donada pel següent conjunt de clàusules*

$$\{\{L_{1,1}, \ldots, L_{1,n_1}\}, \ldots, \{L_{k,1}, \ldots, L_{k,n_k}\}\}.$$

Exemple 2.4 La representació en forma clausal de l'enunciat $A = (P \vee \neg Q \vee \neg R) \wedge (P \vee Q) \wedge \neg P$ és

$$\{\{P, \neg Q, \neg R\}, \{P, Q\}, \{\neg P\}\}.$$

Una clàusula representa una disjunció. Una coma que separa dos literals dins d'una clàusula representa un símbol de disjunció, mentre que una coma que separa dues clàusules representa un símbol de conjunció. La raó per la qual representem un enunciat en FNC com un conjunt de conjunts de literals és degut a que la notació de conjunt proporciona automàticament les simplificacions que sorgeixen de l'associativitat, commutativitat i idempotència de les connectives $\vee$ i $\wedge$, ja que els elements d'un conjunt no estan ordenats. Per tant, no existeix una correspondència bijectiva entre conjunts de clàusules i enunciats. Per exemple, la forma clausal $\{\{Q\}, \{P, \neg R\}\}$ representa, entre d'altres, els següents enunciats:

$$((P \vee \neg R) \wedge (Q \wedge Q))$$
$$((P \vee \neg R) \wedge (Q \vee Q))$$
$$(Q \wedge (\neg R \vee P))$$
$$(Q \wedge ((\neg R \vee \neg R) \vee P))$$

Els conceptes de satisfactibilitat, validesa, insatisfactibilitat i conseqüència lògica també s'apliquen a clàusules i conjunts de clàusules. Tot seguit donem la seva definició:

Definició 2.14

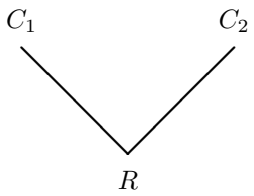
- *Una interpretació $\mathcal{I}$ satisfà una clàusula C, notat $\models_{\mathcal{I}} C$, ssi satisfà almenys un literal de C.*
- *Una interpretació $\mathcal{I}$ satisfà un conjunt de clàusules $\mathcal{C}$, notat $\models_{\mathcal{I}} \mathcal{C}$, ssi satisfà totes les clàusules de $\mathcal{C}$.*
- *Una clàsula és vàlida ssi es satisfà per a totes les interpretacions.*
- *Un conjunt de clàusules és vàlid ssi totes les seves clàusules són vàlides.*
- *Un conjunt de clàusules és insatisfactible ssi no es satisfà per a cap interpretació.*
- *Una clàusula C és conseqüència lògica d'un conjunt de clàusules $\mathcal{C}$ ssi totes les interpretacions que satisfàn $\mathcal{C}$ també satisfàn C.*

2.3.2 Principi de Resolució

Com hem dit a la introducció, el càlcul de resolució té una única regla d'inferència, anomenada principi de resolució. El preu que s'ha de pagar per tenir una sola regla és que hem d'expressar els enunciats en forma clausal.

Definició 2.15 (Principi de Resolució). *Siguin $C_1 = \{L_1, \ldots, L_n\}$ i $C_2 = \{L'_1, \ldots, L'_m\}$ dues clàusules. Direm que la clàusula $R = (C_1 - \{L_i\}) \cup (C_2 - \{L'_j\})$ és una resolvent de C_1 i C_2 si $L_i \in C_1$, $L'_j \in C_2$ i $L_i^d = L'_j$.*

Noteu que si $C_1 = \{L\}$ i $C_2 = \{L^d\}$ llavors obtenim la clàusula buida. La clàusula buida es nota pel símbol $\square$. L'aplicació del principi de resolució a dues clàusules es representa gràficament de la següent manera:



Exemple 2.5 Tot seguit donem alguns exemples de resolvents:

$$\begin{array}{lll} C_1 = \{P, \neg Q, R\} & C_2 = \{Q, R, \neg S\} & R = \{P, R, \neg S\} \\ C_1 = \{Q\} & C_2 = \{\neg P, \neg Q, R\} & R = \{\neg P, R\} \\ C_1 = \{\neg P\} & C_2 = \{P\} & R = \square \\ C_1 = \{P, \neg Q\} & C_2 = \{Q\} & R = \{P\} \end{array}$$

Definició 2.16 *Sigui $\mathcal{C}$ un conjunt de clàusules. Llavors $Res(\mathcal{C})$ es defineix com*

$$Res(\mathcal{C}) = \mathcal{C} \cup \{R | R \text{ es una resolvent de dues clausules de } \mathcal{C}\}$$

. A més a més definim:

$$\begin{array}{rcl} Res^0(\mathcal{C}) & = & \mathcal{C} \\ Res^{n+1}(\mathcal{C}) & = & Res(Res^n(\mathcal{C})) \ \forall n \geq 0 \\ Res^*(\mathcal{C}) & = & \bigcup_{n \geq 0} Res^n(\mathcal{C}) \end{array}$$

Exemple 2.6 Calcular $Res^*(\mathcal{C})$, on

$$\mathcal{C} = \{\{P, Q, R\}, \{\neg S\}, \{\neg Q, S\}, \{\neg P, S\}\}$$

$$\begin{array}{l} Res^0(\mathcal{C}) = \mathcal{C} \\ Res^1(\mathcal{C}) = Res^0(\mathcal{C}) \cup \{\{P, R, S\}, \{Q, R, S\}, \{\neg Q\}, \{\neg P\}\} \\ Res^2(\mathcal{C}) = Res^1(\mathcal{C}) \cup \{\{P, R\}, \{Q, R\}, \{R, S\}\} \\ Res^3(\mathcal{C}) = Res^2(\mathcal{C}) \cup \{R\} \end{array}$$

Com ja no podem derivar més resolvents es té que $Res^*(\mathcal{C}) = Res^3(\mathcal{C})$

Definició 2.17 (prova). *Siguin $\mathcal{C}$ un conjunt de clàusules i C' una clàusula. Una prova per resolució de C' a partir de $\mathcal{C}$ és una seqüència finita $C_1, C_2, \ldots, C_n$ de clàusules tals que cada C_i, $1 \leq i \leq n$, és una clàusula de $\mathcal{C}$ o és una resolvent de dues clàusules C_j, C_k, $1 \leq j, k < i$, i $C_n = C'$. Una prova de $\square$ a partir de $\mathcal{C}$ s'anomena una refutació de $\mathcal{C}$.*

Si existeix una prova per resolució de C' a partir de $\mathcal{C}$ ho notem $\mathcal{C} \vdash_{Res} C'$. Quan no hi ha perill de confusió de que el càlcul que emprem és el de resolució ho notem simplement $\mathcal{C} \vdash C'$. A la resta del capítol ho notarem de la darrera forma.

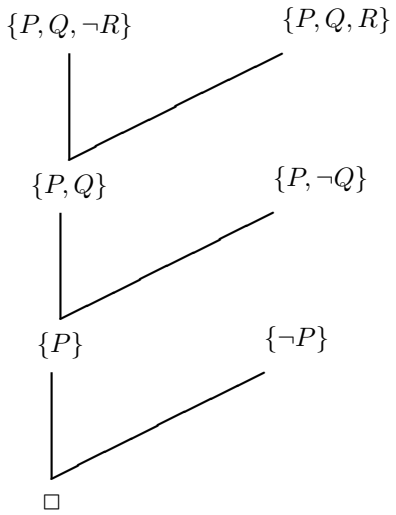
Exemple 2.7 Una refutació per resolució del conjunt de clàusules

$$\mathcal{C} = \{\{P, Q, \neg R\}, \{\neg P\}, \{P, Q, R\}, \{P, \neg Q\}\}$$

és:

$$\begin{array}{ll} C_1 = \{P, Q, \neg R\} & (\textit{clàusula de } \mathcal{C}) \\ C_2 = \{P, Q, R\} & (\textit{clàusula de } \mathcal{C}) \\ C_3 = \{P, Q\} & (\textit{resolvent de } C_1, C_2) \\ C_4 = \{P, \neg Q\} & (\textit{clàusula de } \mathcal{C}) \\ C_5 = \{P\} & (\textit{resolvent de } C_3, C_4) \\ C_6 = \{\neg P\} & (\textit{clàusula de } \mathcal{C}) \\ C_7 = \square & (\textit{resolvent de } C_5, C_6) \end{array}$$

$C_1, C_2, C_3, C_4, C_5, C_6, C_7$ és una refutació de $\mathcal{C}$.



2.3.3 Completesa i solidesa

En aquest apartat demostrarem el teorema de resolució pel CP_0 que estableix les propietats de solidesa i completesa del càlcul de resolució. Tot seguit veurem el lema de resolució i la regla de divisió, utilitzats a la demostració del teorema.

Teorema 2.4 (lema de resolució). *Siguin $\mathcal{C}$ un conjunt de clàusules, C_1 i C_2 dues clàusules de $\mathcal{C}$ i R una resolvent de C_1 i C_2. Llavors, $\mathcal{C}$ i $\mathcal{C} \cup \{R\}$ són lògicament equivalents, o sigui, tenen els mateixos models.*

Demostració: ($\Leftarrow$) Si I és un model de $\mathcal{C} \cup \{R\}$, també és un model de $\mathcal{C}$.

($\Rightarrow$) Sigui I un model de $\mathcal{C}$, i per tant, un model de C_1 i C_2. Com R és de la forma $R = (C_1 - \{L\}) \cup (C_2 - \{L^d\})$ tenim dues possibilitats:

- I és un model de L, llavors I és un model de $(C_2 - \{L^d\})$, i per tant un model de R.
- I és un model de L^d, llavors I és un model de $(C_1 - \{L\})$, i per tant un model de R.

Per tant, els models de $\mathcal{C}$ són models de $\mathcal{C} \cup \{R\}$. ■

Teorema 2.5 (regla de divisió de Davis i Putnam). *Sigui $\mathcal{C}$ un conjunt de clàusules de la forma $\mathcal{C} = \{A_1 \vee L, \ldots, A_p \vee L, B_1 \vee \neg L, \ldots, B_q \vee \neg L, C_1 \ldots C_r\}$*

on L és un literal que no apareix a $C_i, 1 \leq i \leq r$. Siguin $\mathcal{C}_1$ i $\mathcal{C}_2$ dos conjunts de clàusules obtingudes de $\mathcal{C}$ de la forma $\mathcal{C}_1 = \{A_1, \ldots, A_p, C_1, \ldots, C_r\}$ i $\mathcal{C}_2 = \{B_1, \ldots, B_q, C_1, \ldots, C_r\}$. $\mathcal{C}$ és insatisfactible ssi $\mathcal{C}_1$ i $\mathcal{C}_2$ ho són.

Demostració: ($\Rightarrow$) Suposem $\mathcal{C}$ insatisfactible i $\mathcal{C}_1$ (o $\mathcal{C}_2$) satisfactible. Aleshores existeix un model dels $A_i's$ (o dels $B_i's$) i dels $C_i, 1 \leq i \leq r$, però aquest model, juntament amb $L = F$ (V), també ho és de $\mathcal{C}$, contra la hipòtesi. ($\Leftarrow$) Suposem que $\mathcal{C}_1$ i $\mathcal{C}_2$ insatisfactibles i $\mathcal{C}$ satisfactible. Aleshores existeix un model que conté $L = F$ (V) i que fa veritat els $A_i's$ ($B_i's$) i els $C_i, 1 \leq i \leq r$. O sigui, també és un model de $\mathcal{C}_1$ ($\mathcal{C}_2$), contra la hipòtesi. ■

Teorema 2.6 (Teorema de Resolució).
Un conjunt de clàusules $\mathcal{C}$ és insatisfactible ssi $\square \in Res^*(C)$.

Demostració: ($\Leftarrow$, Solidesa) Suposem que $\square \in Res^*(\mathcal{C})$. Llavors $\exists n$ tal que $\square \in Res^n(\mathcal{C})$. La clàusula buida $\square$ sols es pot obtenir com a resolvent de dues clàusules de la forma $\{L\}, \{L^d\}$. Per tant, $\{L\}, \{L^d\} \in Res^n(\mathcal{C})$. Com no existeix cap interpretació que satisfaci $\{L\}$ i $\{L^d\}$, no hi ha cap interpretació que satisfaci $Res^n(\mathcal{C})$. Pel lema de resolució es té:

$$\mathcal{C} \equiv Res^1(\mathcal{C}) \equiv Res^2(\mathcal{C}) \equiv \ldots \equiv Res^n(\mathcal{C})$$

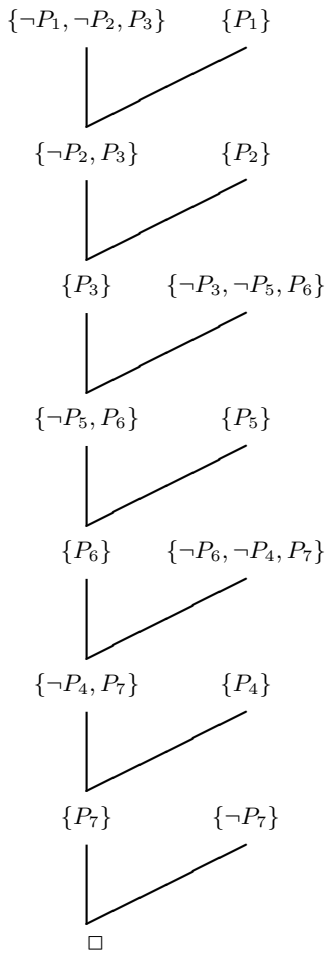
O sigui, $\mathcal{C}$ i $Res^n(\mathcal{C})$ són lògicament equivalents. Pert tant, si $Res^n(\mathcal{C})$ és insatisfactible, llavors $\mathcal{C}$ és insatisfactible.

($\Rightarrow$, Completesa) Apliquem el principi d'inducció sobre el nombre (n) d'àtoms diferents que ocorren en un conjunt de clàusules. Si $n = 0$ tenim la clàusula buida, per tant $\square \in Res^*(\mathcal{C})$. Sigui $\mathcal{C}$ un conjunt de clàusules amb $n+1$ àtoms diferents. $\mathcal{C}$ és de la forma $\mathcal{C} = \{A_1 \vee L, \ldots, A_p \vee L, B_1 \vee \neg L, \ldots, B_q \vee \neg L, C_1 \ldots C_r\}$ on L és un literal que no apareix a $C_i, 1 \leq i \leq r$. A partir de $\mathcal{C}$ definim dos nous conjunts de clàusules:

$$\mathcal{C}_1 = \{A_1, \ldots, A_p, C_1, \ldots, C_r\}$$
$$\mathcal{C}_2 = \{B_1, \ldots, B_q, C_1, \ldots, C_r\}$$

$\mathcal{C}_1$ i $\mathcal{C}_2$ tenen com a màxim n àtoms diferents. Per la regla de Davis-Putman es té que $\mathcal{C}_1$ i $\mathcal{C}_2$ són insatisfactibles, ja que hem suposat que $\mathcal{C}$ ho era. Com $\mathcal{C}_1$ i $\mathcal{C}_2$ tenen n o menys àtoms diferents, per la hipòtesi d'inducció es té que $\square \in Res^*(\mathcal{C}_1)$ i $\square \in Res^*(\mathcal{C}_2)$. O sigui, existeix una refutació $C_1^1, C_2^1, \ldots, C_m^1 = \square$ per a $\mathcal{C}_1$ i una refutació $C_1^2, C_2^2, \ldots, C_n^2 = \square$ per a $\mathcal{C}_2$. Si reconstruïm la refutació de $\mathcal{C}_1$ afegint el literal L en aquells llocs on estava abans d'aplicar la regla de divisió, obtindrem una nova seqüència de clàusules $C_1'^1, C_2'^1, \ldots, C_{m'}'^1$, on $C_m'^1$ és $\square$ o L. Aplicant el mateix raonament per $\mathcal{C}_2$ es té $C_1'^2, C_2'^2, \ldots, C_{n'}'^2$, on $C_{n'}'^2$ és $\square$ o $\neg L$. Si $C_{m'}'^1$ o $C_{n'}'^2$ és $\square$ tenim una refutació de $\mathcal{C}$, sinó com $C_{m'}'^1 = L$ i $C_{n'}'^2 = \neg L$ resolem aquestes dues clàusules i obtenim $\square$. ■

El teorema 2.6 garanteix la completesa i solidesa de la resolució per refutació, o sigui, $\Gamma \models A$ ssi $\Gamma, \neg A \vdash_{Res} \square$. Ara bé, no garanteix que $\Gamma \models A$ ssi $\Gamma \vdash_{Res} A$. Per la propietat de solidesa podem afirmar que si $\Gamma \vdash_{Res} A$ aleshores $\Gamma \models A$, però la inversa no és certa. Ho podem veure amb un contraexemple: $P \models P \vee Q$, en canvi $P \not\vdash_{Res} P \vee Q$. Degut a aquesta situació, s'acostuma a dir que la resolució és completa per a càlcul per refutació i que no és completa com a càlcul deductiu.



Exemple 2.8 Les següents reaccions químiques permeten produir quatre molècules d'àcid sulfúric:

$$\begin{aligned} 4SO + 4H_2 &\longrightarrow 2S_2 + 4H_2O \\ 2S_2 + 6O_2 &\longrightarrow 4SO_3 \\ 4SO_3 + 4H_2O &\longrightarrow 4H_2SO_4 \end{aligned}$$

Provar que si fem reaccionar molècules de SO, H_2, O_2 i H_2O, en quantitats adecuades, podrem produir molècules d'àcid sulfúric. Els compostos químics del nostre problema els notarem pels següents àtoms: $P_1 = 4SO$, $P_2 = 4H_2$, $P_3 = 2S_2$, $P_4 = 4H_2O$, $P_5 = 6O_2$, $P_6 = 4SO_3$ i $P_7 = 4H_2SO_4$. Les reaccions químiques les representem pels enunciats A_1, A_2 i A_3:

$$\begin{aligned} A_1 : P_1 \wedge P_2 &\rightarrow P_3 \wedge P_4 \\ A_2 : P_3 \wedge P_5 &\rightarrow P_6 \\ A_3 : P_6 \wedge P_4 &\rightarrow P_7 \end{aligned}$$

Com disposem dels compostos denotats per P_1, P_2, P_4 i P_5 hem de provar:

$$A_1, A_2, A_3, P_1, P_2, P_4, P_5 \models P_7$$

Si expresem els enunciats A_1, A_2 i A_3 en forma clausal obtenim les clàusules següents:

$$A_1 : \{\{\neg P_1, \neg P_2, P_3\}, \{\neg P_1, \neg P_2, P_4\}\}$$
$$A_2 : \{\neg P_3, \neg P_5, P_6\}$$
$$A_3 : \{\neg P_6, \neg P_4, P_7\}$$

Comprovar la conseqüència lògica és equivalent a trobar una refutació de:

$$\{\{\neg P_1, \neg P_2, P_3\}, \{\neg P_1, \neg P_2, P_4\}, \{\neg P_3, \neg P_5, P_6\},$$
$$\{\neg P_6, \neg P_4, P_7\}, \{P_1\}, \{P_2\}, \{P_4\}, \{P_5\}, \{\neg P_7\}\}$$

$$\begin{array}{lcll}
C_1 & = & \{\neg P_1, \neg P_2, P_3\} & (clàusula) \\
C_2 & = & \{P_1\} & (clàusula) \\
C_3 & = & \{\neg P_2, P_3\} & (resolvent\ C_1, C_2) \\
C_4 & = & \{P_2\} & (clàusula) \\
C_5 & = & \{P_3\} & (resolvent\ C_3, C_4) \\
C_6 & = & \{\neg P_3, \neg P_5, P_6\} & (clàusula) \\
C_7 & = & \{\neg P_5, P_6\} & (resolvent\ C_5, C_6) \\
C_8 & = & \{P_5\} & (clàusula) \\
C_9 & = & \{P_6\} & (resolvent\ C_7, C_8) \\
C_{10} & = & \{\neg P_6, \neg P_4, P_7\} & (clàusula) \\
C_{11} & = & \{\neg P_4, P_7\} & (resolvent\ C_9, C_{10}) \\
C_{12} & = & \{P_4\} & (clàusula) \\
C_{13} & = & \{P_7\} & (resolvent\ C_{11}, C_{12}) \\
C_{14} & = & \{\neg P_7\} & (clàusula) \\
C_{15} & = & \square & (resolvent\ C_{13}, C_{14})
\end{array}$$

Com hem trobat una refutació queda demostrat que podrem produir molècules d'àcid sulfúric. La figura 2.8 dóna una representació gràfica d'aquesta prova.

2.3.4 Procediments de Resolució

Recordem que un enunciat B és una conseqüència lògica d'un conjunt d'enunciats $A_1, A_2, \ldots, A_n$ ssi l'enunciat $((A_1 \wedge A_2 \wedge \ldots \wedge A_n) \wedge \neg B)$ és insatisfactible. Si notem per $A'_1, A'_2, \ldots, A'_n$, la forma clausal de $A_1, A_2, \ldots, A_n$ i per B' la forma clausal de $\neg B$, pel teorema de resolució, comprovar que $A_1, A_2, \ldots, A_n \models B$ és equivalent a comprovar que $\square \in Res^*(\mathcal{C})$, on $\mathcal{C}$ és el conjunt format per les clàusules de $A'_1, A'_2, \ldots, A'_n, B'$. Per a comprovar que $\square \in Res^*(C)$ n'hi ha prou amb trobar una refutació de $\mathcal{C}$. D'aquesta forma es com es demostren per resolució les conseqüències lògiques.

Procediment resolució
Entrada: $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$
begin
$X := \mathcal{C}$
repeat
$Y := X$
$X := Res(Y)$

```
  until (□ ∈ X) ∨ (X = Y)
  if (□ ∈ X) then return ("C insatisfactible")
  else return ("C satisfactible")
  endif
end;
```

Exercicis

Exercici 21 Quin inconvenient té un càlcul que és complet però no és sòlid.

Exercici 22 Demostrar les següents conseqüències lògiques emprant tableaux:

1. $P, P \to Q \models P \wedge Q$
2. $Q, P \to (Q \to R) \models P \to R$
3. $(P \vee Q) \wedge R \models (P \wedge R) \vee (Q \wedge R)$
4. $P \to (Q \to \neg P) \models P \to \neg Q$
5. $(P \to Q) \to (P \to R) \models P \to (Q \to R)$

Exercici 23 Resoldre l'exemple 2.8 emprant tableaux.

Exercici 24 Formalitza el raonament següent i demostra la seva validesa emprant tableaux: Tres famoses folclòriques participen en un debat televisiu sobre el frau a Hisenda. Podem estar segurs que si alguna de les tres defrauda també ho fa alguna de les altres dues. Per altra banda, és indubtable que alguna de les tres defrauda. Per tant, podem concloure que en el millor dels casos, sols n'hi ha una que no defrauda.

Exercici 25 Expressar en forma clausal els següents enunciats:

1. $(P \leftrightarrow \neg Q)$
2. $\neg(P \vee Q) \to R$
3. $\neg(P \to Q) \vee (P \vee Q)$
4. $(((P \to Q) \to R) \to S)$
5. $(P \wedge Q \wedge R) \vee (\neg P \wedge \neg Q \wedge R)$

Exercici 26 Donat un conjunt de clàusules $\mathcal{C}$, demostrar que $\mathcal{C}$ és satisfactible si totes les clàusules de $\mathcal{C}$ tenen un literal negatiu.

Exercici 27 Trobar totes les possibles resolvents entre les clàusules de $\mathcal{C}$.

$$\mathcal{C} = \{\{\neg P, Q\}, \{\neg P, S\}, \{\neg Q, R\}, \{\neg S, R\}, \{P\}\}$$

Exercici 28 Calcular $Res^*(\mathcal{C})$, on

$$\mathcal{C} = \{\{P\}, \{Q, \neg S\}, \{\neg P, Q\}, \{\neg P, \neg Q, R\}\}$$

Exercici 29 Trobar una refutació per resolució de $\mathcal{C}$.

1. $\mathcal{C} = \{\{\neg P_2\}, \{\neg P_1, P_2\}, \{P_1, \neg P_3\}, \{P_3\}\}$
2. $\mathcal{C} = \{\{\neg P_1, P_2\}, \{P_1, P_2\}, \{P_3, \neg P_2\}, \{\neg P_3\}\}$
3. $\mathcal{C} = \{\{\neg P_1\}, \{P_1, P_2\}, \{P_3, P_1\}, \{\neg P_2, \neg P_3\}\}$

Exercici 30 Si plou el concert es suspendrà i si el concert es suspén anirem al cinema. Si anem al cinema i plou agafarem un taxi i si agafem un taxi he d'anar a treure diners del caixer. Hauré d'anar al caixer si plou?.

Exercici 31 Demostrar per resolució que l'enunciat A és una tautologia.

$$A = (\neg P \wedge \neg Q \wedge R) \vee (\neg P \wedge \neg R) \vee (Q \wedge R) \vee P$$

Exercici 32 Demostrar les següents conseqüències lògiques emprant resolució:

1. $P, P \to Q \models P \wedge Q$
2. $Q, P \to (Q \to R) \models P \to R$
3. $(P \vee Q) \wedge R \models (P \wedge R) \vee (Q \wedge R)$
4. $P \to (Q \to \neg P) \models P \to \neg Q$
5. $(P \to Q) \to (P \to R) \models P \to (Q \to R)$

Capítol 3

Lògica de Predicats

3.1 Introducció

No tots els arguments vàlids els podem representar en CP_0. Per exemple, si tenim l'argument:

	Tots els nens mengen caramels
	Joan és un nen
per tant	Joan menja caramels

el representariem en CP_0 com $P, Q \models R$, on P = Tots els nens menjen caramels, Q = Joan es un nen i R = Joan menja caramels. Com podeu comprovar R no és una conseqüència lògica de P i Q. En CP_0 estudiem els enunciats declaratius simples com un tot, i hi ha raonaments amb els quals cal tenir en compte l'estructura dels enunciats simples. En el cas anterior tenim la següent estructura:

	Tots els a mengen b
	c és a
per tant	c menja b

La Lògica de Predicats (CP_1) es una extensió de la Lògica Proposicional que afegeix nous conceptes com quantificadors, funcions i predicats que permeten considerar l'estructura dels enunciats declaratius simples. Aquests conceptes augmenten la riquesa expressiva respecte al CP_0 permetent expressar que certs objectes pertanyen a una relació, que una propietat la tenen tots els objectes que considerem o que existeixen objectes que la tenen. En contrapartida, el CP_1 té una sintaxi, una semàntica i una teoria de la prova més complexes.

3.2 El llenguatge del CP_1

Definició 3.1 alfabet *l'alfabet $\sum$ del llenguatge del CP_1 té els següents components:*

1. *Un conjunt de símbols de variables* $\{x_1, x_2, \ldots, x_n, \ldots\}$
2. *Un conjunt de símbols de constants* $\{a_1, a_2, \ldots, a_n, \ldots\}$
3. *Un conjunt de símbols de predicats* $\{P_1^{n_1}, P_2^{n_2}, \ldots\}$

4. *Un conjunt de símbols de funcions* $\{f_1^{n_1}, f_2^{n_2}, \ldots\}$
5. *Un conjunt de símbols de connectives* $\{\neg, \rightarrow, \wedge, \vee\}$
6. *Un conjunt de símbols de quantificadors* $\{\forall, \exists\}$
7. *Un conjunt de símbols de puntuació* $\{(,),$ ",$"\}$

$$\begin{aligned}\sum \quad = \quad & \{x_1, x_2, \ldots, x_n, \ldots, a_1, a_2, \ldots, a_n, \ldots, P_1^{n_1}, P_2^{n_2}, \ldots, \\ & f_1^{n_1}, f_2^{n_2}, \ldots, \neg, \rightarrow, \wedge, \vee, \forall, \exists, (,), \text{“,”}\}\end{aligned}$$

El superíndex dels símbols de funcions i predicats és un número sencer positiu, anomenat *aritat*, que indica el número d'arguments de la funció o predicat. Quan es posen explícitament els arguments d'una funció o predicat s'acostuma a omitir aquest superíndex, així doncs, quan escrivim $f_1(x_1, a_3)$ ens referim a $f_1^2(x_1, a_3)$.

Definició 3.2 quantificadors *són els símbols* $\forall$ *i* $\exists$*, llegits per a tot i per a algun. Són operadors unaris. El seu efecte s'estén a la fórmula que els segueix immediatament, anomenada camp del quantificador.*

Definició 3.3 termes

1. *Variables i constants són termes.*
2. *Si* f_i^n *és un símbol de funció i* $t_1, \ldots, t_n$ *són termes, llavors* $f_i^n(t_1, \ldots, t_n)$ *és un terme.*
3. *No hi ha cap altre terme.*

Definició 3.4 fórmula atòmica *Si* P_i^n *és un símbol de predicat i* $t_1, \ldots, t_n$ *són termes, llavors* $P_i^n(t_1, \ldots, t_n)$ *és una fórmula atòmica.*

Definició 3.5 literal *Un literal és una fórmula atòmica o la negació d'una fórmula atómica. Un literal positiu és una fórmula atómica i un literal negatiu és la negació d'una fórmula atómica.*

Les fórmules ben formades (fbf's) del CP_1 es generen mitjançant les següents regles:

1. Tota fórmula atòmica és una fbf.
2. Si A i B són fbf's i x_1 és un símbol de variable, llavors $(\neg A), (A \wedge B), (A \rightarrow B), (A \vee B), (\forall x_1 A), (\exists x_1 A)$ són fbf.
3. No hi ha cap altra fbf.

Noteu que de fet no definim un llenguatge únic sinó una familia de llenguatges ja que els conjunts de símbols de constants, predicats i funcions no estan prefixats. Notarem $\mathcal{L}(\mathcal{C}, \mathcal{F}, \mathcal{P})$ aquell llenguatge del CP_1 que podem generar, aplicant les regles anteriors, a partir de l'alfabet que conté a C com a símbols de constants, a F com a símbols de funcions i a P com a símbols de predicats. Els llenguatges del CP_1 s'anomenen llenguatges de primer ordre, i els únics conjunts de símbols que poden ser buits són el de constants i el de funcions.

Definició 3.6 variables lliures i lligades *Una ocurrència d'una variable x_1 a una fbf direm que és lligada ssi o bé és la variable que segueix immediatament el símbol d'un quantificador, o bé cau dins del camp d'un quantificador de la forma $\forall x_1$ o $\exists x_1$. Tota ocurrència d'una variable que no és lligada direm que és lliure.*

Exemple 3.1

- Donada la fbf $\forall x_1 P(x_2)$, on el camp del quantificador és $P(x_2)$, es té que l'ocurrència de la variable x_2 és lliure, mentre que l'ocurrència de la variable x_1 és lligada.
- A la fbf $\forall x_1 \forall x_2 (P_1(x_1, x_2) \rightarrow P_2(x_2))$ totes les ocurrències de les variables x_1 i x_2 són lligades. Noteu que el camp del quantificador $\forall x_1$ és $\forall x_2 (P_1(x_1, x_2) \rightarrow P_2(x_2))$, mentre que el camp del quantificador $\forall x_2$ és $(P_1(x_1, x_2) \rightarrow P_2(x_2))$.
- A la fbf $\forall x_1 (P_1(x_1, x_2) \rightarrow (\forall x_2 P_2(x_2)))$ les dues ocurrències de la variable x_1 són lligades, mentre que per la variable x_2, la primera ocurrència és lliure i la segona és lligada.

S'anomena *fórmula oberta* tota fórmula que té variables lliures i *fórmula tancada* tota fórmula que no té variables lliures. Els *enunciats* del CP_1 són les fbf's que són tancades. Notació: Les lletres majúscules $A_1, A_2, \ldots$ noten els enunciats i les lletres majúscules $F_1, F_2, \ldots$ noten les fbf's.

3.2.1 Substitucions

Definició 3.7 Substitució *Una substitució és un conjunt finit de la forma $\{t_1/x_1, \ldots, t_n/x_n\}$ on cada t_i és un terme i cada x_i és una variable tals que $x_i \neq t_i$ i $x_i \neq x_j, \forall i \neq j$. La substitució buida es nota per ε.*

Definició 3.8 Aplicació *Sigui $\sigma = \{t_1/x_1, \ldots, t_n/x_n\}$ una substitució i E un terme o una fbf. L'aplicació de σ a E, notada $E\sigma$, és el terme o fbf obtingut al reemplaçar simultàneament a E cada ocurrència de la variable x_i per t_i. $E\sigma$ s'anomena una instància de E.*

Exemple 3.2

$$P_1(f_1(x_1, x_2), f_2(x_3, a_1))\{a_1/x_1, x_3/x_2\} \quad = \quad P_1(f_1(a_1, x_3), f_2(x_3, a_1))$$

Definició 3.9 composició de substitucions *Siguin $\sigma = \{t_1/x_1, \ldots, t_n/x_n\}$ i $\tau = \{u_1/y_1, \ldots, u_n/y_n\}$ dues substitucions. La composició de σ i τ, notada $\sigma\tau$, és la substitució que s'obté del conjunt*

$$\{t_1\tau/x_1, \ldots, t_n\tau/x_n, u_1/y_1, \ldots, u_m/y_m\}$$

on hem eliminat tots els elements u_i/y_i tals que $y_i \in \{x_1, \ldots, x_n\}$ i tots els elements $t_j\tau/x_j$ per als quals $t_j\tau = x_j$.

Propietats de la composició de substitucions:

- associativitat

$$(\theta\sigma)\tau = \theta(\sigma\tau)$$

- identitat per l'esquerra i per la dreta

$$\varepsilon\theta = \theta\varepsilon = \theta$$

-

$$(E\theta)\sigma = E(\theta\sigma)$$

Noteu que, en general, la composició de substitucions no és commutativa.

Exemple 3.3 *Siguin* $\sigma = \{t_1/x_1, t_2/x_2\} = \{f(y)/x, z/y\}$ *i* $\tau = \{u_1/y_1, u_2/y_2, u_3/y_3\} = \{a/x, b/y, y/z\}$ *dues substitucions. La seva composició* $\sigma\tau$ *és* $\{f(b)/x, y/z\}$ *ja que* $\{t_1\tau/x_1, t_2\tau/x_2, u_1/y_1, u_2/y_2, u_3/y_3\} = \{f(b)/x, y/y, a/x, b/y, y/z\}$ *i eliminem* y/y *per ser* $t_2\tau = x_2$ *i* $a/x = u_1/y_1$ *i* $b/y = u_2/y_2$ *ja que* $y_1 = x_1$ *i* $y_2 = x_2$.

3.3 Semàntica

Hem definit els llenguatges de primer ordre com objectes sintàctics sense significat. Amb aquests llenguatges volem expressar que certs objectes pertanyen a una relació (p.e. Joan és estudiant), que una propietat la tenen tots els objectes que considerem (p.e. tots els nombres parells són divisibles per 2), que existeixen objectes que tenen una propietat (p.e algunes persones estudien lògica), ... Per tant, hem de definir amb precisió la manera d'atribuir significat als símbols d'un llenguatge, i ho farem mitjançant el concepte d'interpretació. Després estudiarem els conceptes de validesa, satisfactibilitat, conseqüència lògica i equivalència.

Definició 3.10 interpretació *Una interpretació* $\mathcal{I}$ *per a un llenguatge de primer ordre* $\mathcal{L}(\mathcal{C}, \mathcal{F}, \mathcal{P})$ *està formada per un conjunt no buit* D*, anomenat domini, i una funció* I*, anomenada funció d'interpretació, que assigna*

- *a cada símbol de constant* $a_i \in \mathcal{C}$ *un element* $I(a_i)$ *de* D
- *a cada símbol de funció* $f_i^n \in \mathcal{F}$ *una funció* $I(f_i^n)$ *amb domini* D^n *i rang* D
- *a cada símbol de predicat* $P_i^n \in \mathcal{P}$ *una relació* $I(P_i^n)$ *sobre* D *(*$I(P_i^n) \subseteq D^n$*).*

Definició 3.11 valoració *Sigui* $\mathcal{I}$ *una interpretació d'un llenguatge de primer ordre* $\mathcal{L}(\mathcal{C}, \mathcal{F}, \mathcal{P})$*. Una valoració respecte a* $\mathcal{I}$*, notada* $\varphi_{\mathcal{I}}$*, és una funció que assigna a cada símbol de variable de* $\mathcal{L}(\mathcal{C}, \mathcal{F}, \mathcal{P})$ *un element del domini de* $\mathcal{I}$*.*

Definició 3.12 semàntica dels termes *Sigui* $\mathcal{I}$ *una interpretació,* $\varphi_{\mathcal{I}}$ *una valoració respecte a* $\mathcal{I}$ *i* t *un terme. El significat de* t*, notat* $\varphi^*_{\mathcal{I}}(t)$*, és un element del domini de* $\mathcal{I}$ *definit de la següent forma:*

- *Si* t *és un símbol de constant* a_i *llavors* $\varphi^*_{\mathcal{I}}(t) = I(a_i)$*.*
- *Si* t *és un símbol de variable* x_i *llavors* $\varphi^*_{\mathcal{I}}(t) = \varphi(x_i)$*.*
- *Si* t *és un terme de la forma* $f_i(t_1, \ldots, t_n)$ *llavors* $\varphi^*_{\mathcal{I}}(t) = I(f_i)(\varphi^*_{\mathcal{I}}(t_1), \ldots, \varphi^*_{\mathcal{I}}(t_n))$*.*

Exemple 3.4 Donat el llenguatge $L(\{a_1\},\{f_1^1,f_2^2\},\{P_1^2\})$ considerem la interpretació $\mathcal{I}$ on el domini són els naturals ($D=\mathcal{N}$), el símbol de constant a_1 s'interpreta com el zero dels naturals, els símbols de funció f_1^1 i f_2^2 s'interpreten com la funció successor i la suma, i el símbol de predicat P_1^2 com la relació d'igualtat. Si per aquesta interpretació definim una valoració que assigna el nombre natural 2 a la variable x_1, el significat del terme $f_2(f_1(a_1),x_1)$ és:

$$\begin{aligned}\varphi^*_{\mathcal{I}}(f_2(f_1(a_1),x_1)) &= \varphi^*_{\mathcal{I}}(f_1(a_1))+\varphi^*_{\mathcal{I}}(x_1)\\ &= succ(\varphi^*_{\mathcal{I}}(a_1))+2\\ &= succ(0)+2\\ &= 1+2\\ &= 3\end{aligned}$$

Un cop donada semàntica als termes podem donar significat a les fbf's. El fet de que una fbf F sigui veritat per a una interpretació $\mathcal{I}$ i una valoració $\varphi_{\mathcal{I}}$ ho notem $\models^{\varphi}_{\mathcal{I}} F$, i si és falsa ho notem $\not\models^{\varphi}_{\mathcal{I}} F$.

Definició 3.13 semàntica de les fbf's *Sigui $\mathcal{I}$ una interpretació i $\varphi_{\mathcal{I}}$ una valoració, el significat de una fbf és defineix com:*

- $\models^{\varphi}_{\mathcal{I}} P_i(t_1,\ldots,t_n)$ *ssi* $(\varphi^*_{\mathcal{I}}(t_1),\ldots,\varphi^*_{\mathcal{I}}(t_n))\in I(P_i)$.
- $\models^{\varphi}_{\mathcal{I}} (\neg F)$ *ssi* $\not\models^{\varphi}_{\mathcal{I}} F$
- $\models^{\varphi}_{\mathcal{I}} (F\wedge G)$ *ssi* $\models^{\varphi}_{\mathcal{I}} F$ *i* $\models^{\varphi}_{\mathcal{I}} G$
- $\models^{\varphi}_{\mathcal{I}} (F\vee G)$ *ssi* $\models^{\varphi}_{\mathcal{I}} F$ *o* $\models^{\varphi}_{\mathcal{I}} G$
- $\models^{\varphi}_{\mathcal{I}} (F\rightarrow G)$ *ssi* $\not\models^{\varphi}_{\mathcal{I}} F$ *o* $\models^{\varphi}_{\mathcal{I}} G$
- $\models^{\varphi}_{\mathcal{I}} (\forall x_i F)$ *ssi* $\models^{\vartheta}_{\mathcal{I}} F$ *per a tota valoració* $\vartheta_{\mathcal{I}}$ *tal que* $\vartheta_{\mathcal{I}}(x)=\varphi_{\mathcal{I}}(x)$ *per a tota* $x\neq x_i$.
- $\models^{\varphi}_{\mathcal{I}} (\exists x_i F)$ *ssi* $\models^{\vartheta}_{\mathcal{I}} F$ *per alguna valoració* $\vartheta_{\mathcal{I}}$ *tal que* $\vartheta_{\mathcal{I}}(x)=\varphi_{\mathcal{I}}(x)$ *per a tota* $x\neq x_i$.

Noteu que si una fbf és tancada el valor de veritat sols depén de la interpretació i no de la valoració.

Definició 3.14 model d'un enunciat *Sigui $\mathcal{I}$ una interpretació d'un llenguatge de primer ordre $\mathcal{L}(\mathcal{C},\mathcal{F},\mathcal{P})$ i A un enunciat. Direm que $\mathcal{I}$ és un model de A o que $\mathcal{I}$ satisfà A, notat $\models_{\mathcal{I}} A$, ssi A és veritat per aquesta interpretació. Altrament, direm que $\mathcal{I}$ falsifica A.*

Una interpretació $\mathcal{I}$ és un model d'un conjunt d'enunciats Γ ssi és un model de cadascún dels enunciats de Γ. Noteu que si $\Gamma=\{A_1,\ldots,A_n\}$, $\mathcal{I}$ és un model de Γ ssi $\mathcal{I}$ és un models de $A_1\wedge\ldots\wedge A_n$.

Exemple 3.5 Donat l'enunciat $\forall x P(x,a)$ si prenem com a domini els naturals ($D=\mathcal{N}$) i interpretem a com el zero i P com la relació $\geq$, l'enunciat anterior és satisfet per aquesta interpretació, ja que tot nombre natural sempre és més gran o igual que zero. Donat l'enunciat $\forall x P(x,a)$ si prenem com a domini els sencers ($D=Z$) i interpretem a com el zero i P com la

relació $\geq$, l'enunciat anterior no és satisfet per aquesta interpretació, ja que tot nombre sencer no sempre és més gran o igual que zero. La excepció són els nombres negatius. Donat l'enunciat $\forall x P(x, a)$ si prenem com a domini el conjunt $\{Joan, Pep, Margarida\}$ i interpretem que la constant a denota Margarida i el predicat P la relació estimar donada pels parells següents $\{(Joan, Margarida), (Pep, Margarida), (Margarida, Joan)\}$ (aquesta relació expressa que en Joan i en Pep estimem a na Margarida i na Margarida estima a en Joan), l'enunciat anterior no és satisfet per aquesta interpretació, ja que no es compleix que na Margarida s'estima a si mateixa.

Exemple 3.6 Donat l'enunciat $\forall x_1 \exists x_2 P(x_1, x_2)$ si prenem com a domini els sencers ($D = \mathcal{Z}$) i interpretem P com la relació $<$, l'enunciat anterior és satisfet per aquesta interpretació, ja que per a tot nombre sencer sempre podem trobar un nombre sencer més petit. Si prenem com a domini els naturals ($D = N$) i interpretem P com la relació $<$, l'enunciat anterior no és satisfet per aquesta interpretació, ja que per a tot nombre natural no sempre podem trobar un nombre natural més petit. L'excepció és el zero. Donat l'enunciat $\forall x_1 \exists x_2 P(x_1, x_2)$ si prenem com a domini el conjunt $\{Joan, Pep, Margarida\}$ i interpretem que el predicat P denota la relació estimar donada pels parells següents $\{(Joan, Margarida), (Pep, Margarida), (Margarida, Joan)\}$, l'enunciat anterior és satisfet per aquesta interpretació, ja que tots els individus són estimats per algun individu del domini.

Un cop definits el conceptes anteriors podem definir els conceptes de validesa, satisfactibilitat, conseqüència lògica i equivalència lògica de la mateixa manera que ho vàrem fer pel CP_O.

Definició 3.15 enunciat vàlid *Un enunciat A direm que és vàlid ssi és veritat sota totes les possibles interpretacions. També es diu que és una tautologia i es nota per $\models A$. En cas contrari es diu que A és invàlid i es nota per $\not\models A$.*

Definició 3.16 enunciat insatisfactible *Un enunciat A direm que és insatisfactible ssi és fals sota totes les possibles interpretacions i es nota per $\models \neg A$. En cas contrari es diu que A és satisfactible i es nota per $\not\models \neg A$.*

Un conjunt d'enunciats Γ és satisfactible si existeix una interpretació $\mathcal{I}$ que és un model de Γ. Γ és vàlid si tota interpretació $\mathcal{I}$ és un model de Γ. Γ és insatisfactible si no té cap model.

Definició 3.17 conseqüència lògica *Siguin $A_1, A_2, ..., A_n$ i B enunciats. Direm que B és una conseqüència lògica de $A_1, A_2, ..., A_n$ ssi per a qualsevol interpretació I que satisfà $A_1 \wedge A_2 \wedge ... \wedge A_n$, I també satisfà B.*

Definició 3.18 Enunciats lògicament equivalents *Siguin A i B dos enunciats. Direm que A i B són lògicament equivalents si i només si els models de A i B coincideixen. Es nota $A \equiv B$. L'expressió $A \equiv B$ s'anomena equivalència.*

Les equivalències que vàrem donar pel CP_0 també es verifiquen pel CP_1. Tot seguit donem algunes equivalències del CP_1 on apareixen quantificadors. F_1 i F_2 són fbf's.

1.
$$\neg\forall x_1 F_1 \equiv \exists x_1 \neg F_1$$
$$\neg\exists x_1 F_1 \equiv \forall x_1 \neg F_1$$

2. Si x_1 no ocorre lliure a F_2 llavors:
$$(\forall x_1 F_1 \wedge F_2) \equiv \forall x_1 (F_1 \wedge F_2)$$
$$(\forall x_1 F_1 \vee F_2) \equiv \forall x_1 (F_1 \vee F_2)$$
$$(\exists x_1 F_1 \wedge F_2) \equiv \exists x_1 (F_1 \wedge F_2)$$
$$(\exists x_1 F_1 \vee F_2) \equiv \exists x_1 (F_1 \vee F_2)$$

3.
$$(\forall x_1 F_1 \wedge \forall x_1 F_2) \equiv \forall x_1 (F_1 \wedge F_2)$$
$$(\exists x_1 F_1 \vee \exists x_1 F_2) \equiv \exists x_1 (F_1 \vee F_2)$$

4. Si x_1 no ocorre lliure a F_1 i $x_1 \neq x_2$ llavors:
$$\forall x_1 F_1 \equiv \forall x_2 F_1\{x_2/x_1\}$$
$$\exists x_1 F_1 \equiv \exists x_2 F_1\{x_2/x_1\}$$

5.
$$\forall x_1 \forall x_2 F_1 \equiv \forall x_2 \forall x_1 F_1$$
$$\exists x_1 \exists x_2 F_1 \equiv \exists x_2 \exists x_1 F_1$$

3.4 Indecidibilitat

Al 1936, Alonzo Church i Alan Turing varen demostrar des de dos enfocs diferents que el problema de la validesa i el problema de la insatisfactibilitat per la lògica de predicats eren indecidibles, i.e., no existeix cap procediment efectiu que permeti decidir, en un temps finit, si un conjunt finit d'enunciats és insatisfactible. És a dir, la lògica de primer ordre és *indecidible*. Malgrat tot, la situació no és tan greu com pot semblar a primer cop d'ull, ja que existeixen procediments que detecten en un temps finit la insatisfactibilitat d'un conjunt finit d'enunciats de la lògica de predicats si aquest conjunt és insatisfactible, però quan no ho és (o sigui, és satisfactible), el procediment pot entrar en un bucle i no aturar-se. Per tant, la lògica de predicats (més concretament, el problema de la validesa i el problema de la insatisfactibilitat) és *semi-decidible*, i.e, hi ha procediments efectius que permeten decidir en temps finit les respostes positives (insatisfactible), però no sempre les respostes negatives (satisfactible).

Exercicis

Exercici 33 Dissenyar un procediment per decidir si una expressió és un enunciat de la lògica de predicats.

Exercici 34 Formalitzar els enunciats declaratius següents com enunciats del CP_1:

1. Tot català estima alguna catalana.
2. Alguns catalans estimen totes les catalanes.
3. Cada catalana es estimada per algun català.

4. Alguna catalana es estimada per tots els catalans.
5. Els catalans que ballen sardanes porten barretina.
6. Totes les catalanes, i alguns catalans, saben fer escudella.
7. Els catalans, si tenen fills, visiten per Nadal la fira de Santa Llucia.
8. En Joan no coneix ningú que l'admiri.
9. No tots els estudiants admiren als seus professors.
10. Les amigues de les meves admiradores són les meves amigues.
11. Na Margarida admira tots els professors, tret del professor de gimnàstica.
12. No tots els professors que admiren na Margarida admiren en Joan, però ell si que els admira.

Exercici 35 Expressar cadascun dels enunciats següents com un enunciat declaratiu (interpreteu el predicat $P(x, y)$ com "x enveja y").

1. $\forall x \exists y P(x, y)$
2. $\exists y \forall x P(x, y)$
3. $\exists x \forall y P(x, y)$
4. $\forall y \exists x P(x, y)$
5. $\forall x \forall y P(x, y)$
6. $\exists x \exists y P(x, y)$

Exercici 36 Calcular la composició de les substitucions σ i τ:

1. $\sigma = \{f(x_3)/x_1, x_4/x_2\}$ i $\tau = \{a/x_1, a/x_3, x_2/x_4\}$
2. $\sigma = \{f(a, x_2)/x_1, f(x_6, x_3)/x_4\}$ i $\tau = \{f(a, a)/x_1, b/x_2, c/x_5\}$
3. $\sigma = \{f(x_1, x_2)/x_1, h(a)/x_2, g(c, h(x_1))/x_3\}$ i
   $\tau = \{b/x_1, g(a, x_1)/x_2, x_3/x_4\}$

Exercici 37 Dir quines de les interpretacions següents són models de l'enunciat

$$\exists x \exists y \exists z (P(x, y) \wedge P(z, y) \wedge P(x, z) \wedge \neg P(z, x))$$

1. $D = \mathcal{N}$, $I(P) = \{(m, n) | m, n \in \mathcal{N}, m < n\}$
2. $D = \mathcal{N}$, $I(P) = \{(m, m + 1) | m \in \mathcal{N}\}$
3. $D = 2^{\mathcal{N}}$, $I(P) = \{(A, B) | A, B \subseteq \mathcal{N}, A \subseteq B\}$

Exercici 38 Definir un llenguatge de primer ordre i una interpretació sobre aquest llenguatge que tingui com a domini els essers humans. A més a més, teniu en compte que en aquest llenguatge volem formalitzar les propietas de ser home i ser dona, i les relacions d'un individu amb els seus progenitors, amb el seu avi i la seva àvia. Formalitzar amb el llenguatge que acaves de definir la relació avi i àvia d'un individu a partir de les altres relacions i propietats.

Exercici 39 Donar interpretacions que siguin models i interpretacions que no ho siguin per cadascún dels enunciats següents:

1. $P_1(a_1, a_2) \rightarrow P_1(a_2, a_1)$
2. $\forall x_1 \forall x_2 (P_1(x_1, x_2) \rightarrow P_1(x_2, x_1))$
3. $\exists x_1 \forall x_2 (P_1(x_1, x_2) \wedge \neg P_1(a_1, x_2))$
4. $\forall x_1 (P_2(x_1) \rightarrow \exists x_2 (P_3(x_2) \wedge P_1(x_2, x_1)))$

Exercici 40 Dir quins dels següents enunciats són lògicament equivalents:

1. $\forall x (P_1(x) \rightarrow P_2(x))$
2. $\forall x (P_1(x) \wedge P_2(x))$
3. $\neg \exists x (P_1(x) \wedge \neg P_2(x))$
4. $\exists x \neg (P_1(x) \wedge P_2(x))$
5. $\forall x \neg (P_1(x) \rightarrow P_2(x))$
6. $\neg \exists x \neg (P_1(x) \wedge P_2(x))$
7. $\neg \forall x (P_1(x) \wedge P_2(x))$

Exercici 41 Donar algun contraexemple per demostrar que les següents fbf's no són equivalents:

$$(\forall x_1 F_1 \vee \forall x_1 F_2) \not\equiv \forall x_1 (F_1 \vee F_2)$$
$$(\exists x_1 F_1 \wedge \exists x_1 F_2) \not\equiv \exists x_1 (F_1 \wedge F_2)$$

Exercici 42 Demostrar les següents equivalències lògiques:

1. $\big(\forall x_1 P_1(a_3, x_1)\big) \rightarrow \big(\forall x_2 \exists x_3 P_1(x_2, x_3)\big) \equiv$
   $\exists x_1 \forall x_2 \exists x_3 \big(P_1(a_3, x_1) \rightarrow P_1(x_2, x_3)\big)$
2. $\exists x_1 (P_2(x_1) \rightarrow P_1(x_1)) \equiv \forall x_2 P_2(x_2) \rightarrow \exists x_1 P_1(x_1)$

Capítol 4

Procediments de prova de primer ordre

4.1 Introducció

En aquest tema estudiem dos tipus de procediments de prova per la lògica de primer ordre, els procediments de Herbrand i la resolució.

4.2 Forma clausal

Una característica del principi de resolució, així com de la majoria de mètodes de deducció automàtica, és fer inferència sobre fòrmules estructuralment molt simples, anomenades clàusules. Qualsevol enunciat A del CP_1 es pot expressar com un conjunt de clàusules $\mathcal{C}$, de manera que A és satisfactible ssi $\mathcal{C}$ ho és. En aquesta secció definirem els conceptes de clàusula i forma clausal pel CP_1, i donarem un algorisme per transformar un enunciat en la seva forma clausal.

Definició 4.1 clàusula *Una clàusula és un enunciat de la forma*

$$\forall x_1 \ldots \forall x_n (L_1 \vee \ldots \vee L_m)$$

on $L_1, \ldots, L_m$ *són literals i* $x_1, \ldots, x_n$ *són les variables que ocorren a* $L_1, \ldots, L_m$.

La clàusula $\forall x_1 \ldots \forall x_n (L_1 \vee \ldots \vee L_m)$ s'acostuma a representar pel conjunt $\{L_1, \ldots, L_m\}$, on les comes representen disjuncions i es sobreentén que les variables estan quantificades universalment.

Definició 4.2 forma clausal *Un enunciat està expressat en forma clausal ssi és de la forma*

$$\forall x_1 \ldots \forall x_k ((L_{11} \vee \ldots \vee L_{1m_1}) \wedge \ldots \wedge (L_{n1} \vee \ldots \vee L_{nm_n}))$$

on $L_{11}, \ldots, L_{1m_1}, \ldots, L_{n1}, \ldots, L_{nm_n}$ *són literals i* $x_1, \ldots, x_k$ *són les variables que ocorren a* $L_{11}, \ldots, L_{1m_1}, \ldots, L_{n1}, \ldots, L_{nm_n}$.

La forma clausal $\forall x_1 \ldots \forall x_k((L_{11} \vee \ldots \vee L_{1m_1}) \wedge \ldots \wedge (L_{n1} \vee \ldots \vee L_{nm_n}))$ s'acostuma a representar pel conjunt de clàusules $\{\{L_{11}, \ldots, L_{1m_1}\}, \ldots, \{L_{n1}, \ldots, L_{nm_n}\}\}$. Tot seguit donem l'algorisme de transformació a forma clausal:
Pas 1. Eliminar les connectives $\rightarrow$ i $\leftrightarrow$ usant les equivalències:

$$A_1 \leftrightarrow A_2 \equiv (A_1 \rightarrow A_2) \wedge (A_2 \rightarrow A_1) \quad (4.1)$$
$$A_1 \rightarrow A_2 \equiv \neg A_1 \vee A_2 \quad (4.2)$$

Pas 2. Reduir al màxim l'àmbit del símbol de negació usant les següents equivalències:

$$\neg(\neg A_1) \equiv A_1 \quad (4.3)$$
$$\neg(A_1 \vee A_2) \equiv \neg A_1 \wedge \neg A_2 \quad (4.4)$$
$$\neg(A_1 \wedge A_2) \equiv \neg A_1 \vee \neg A_2 \quad (4.5)$$
$$\neg\forall x_1 A_1 \equiv \exists x_1 \neg A_1 \quad (4.6)$$
$$\neg\exists x_1 A_1 \equiv \forall x_1 \neg A_1 \quad (4.7)$$

Pas 3. Rebatejar les variables lligades de manera que cada quantificador lligui una variable diferent. Per exemple, l'enunciat $\forall x P(x) \vee \forall x Q(x)$ el reemplacem per l'enunciat lògicament equivalent $\forall x P(x) \vee \forall y Q(y)$.
Pas 4. Eliminar els quantificadors existencials de la següent manera: Si el quantificador existencial que volem eliminar no cau dins del camp d'un quantificador universal llavors eliminem el quantificador i substituïm totes les ocurrències de la variable que quantifica per un nou símbol de constant. Per exemple, si tenim l'enunciat $\exists x_1 P_1(x_1)$ el reemplacem per l'enunciat $P_1(b_1)$. El nou símbol de constant, en aquest cas b_1, s'anomena *constant de Skolem*. Si el quantificador existencial que volem eliminar cau dins del camp d'algun quantificador universal llavors eliminem el quantificador existencial i substituïm totes les ocurrències de la variable que quantifica pel terme format per un nou símbol de funció que té com arguments les variables dels quantificadors universals que afectaven el quantificador existencial que hem eliminat. Per exemple, si tenim l'enunciat $\forall x_1 \forall x_2 \exists x_3 P_1(x_1, x_2, x_3)$ el reemplacem per l'enunciat $\forall x_1 \forall x_2 P_1(x_1, x_2, g_1(x_1, x_2))$. Aquesta nova funció, en aquest cas g_1, s'anomena *funció de Skolem*.
Pas 5. Portar els quantificadors universals a l'esquerra. Per exemple, reemplacem l'enunciat $\forall x_1 P_1(x_1) \vee \forall x_2 P_2(x_2)$ per l'enunciat lògicament equivalent $\forall x_1 \forall x_2 (P_1(x_1) \vee P_2(x_2))$.
Pas 6. Expressar la fbf que queda a la dreta dels quantificadors en FNC usant les següents equivalències:

$$A_1 \vee (A_2 \wedge A_3) \equiv (A_1 \vee A_2) \wedge (A_1 \vee A_3) \quad (4.8)$$
$$(A_1 \wedge A_2) \vee A_3 \equiv (A_1 \vee A_3) \wedge (A_2 \vee A_3) \quad (4.9)$$

Pas 7. Expressar la forma clausal obtinguda al *Pas 6* com un conjunt de clàusules.

Noteu que tots els passos, excepte el d'eliminació de quantificadors existencials, de l'algorisme de transformació a forma clausal preserven l'equivalència lògica entre l'enunciat generat al pas anterior i el generat en el propi pas. Això és així ja que l'únic que fem en aquests passos és aplicar equivalències lògiques.

Aquest no és el cas de la introducció de constants i funcions de Skolem: Si tenim l'enunciat $\exists x_1 P_1(x_1)$ a l'eliminar el quantificador existencial obtenim l'enunciat $P_1(b)$, on b és una constant nova. Si considerem la interpretació que té com a domini el conjunt $\{0,1\}$, i que interpreta a b com 0 i a P_1 com la relació $\{x|x>0\}$, es té que aquesta interpretació és un model de $\exists x_1 P_1(x_1)$ i no ho és de $P_1(b)$. Quan apliquem skolemització no tenim equivalència lògica, però tenim un tipus d'equivalència més feble respecte a la satisfactibilitat (un enunciat és satisfactible ssi ho és la seva forma clausal) com enuncia el següent teorema:

Teorema 4.1 *Sigui A un enunciat i $\mathcal{C}$ el conjunt de clàusules obtingut a l'aplicar l'algorisme de transformació de forma clausal a A.*

1. *$\mathcal{C}$ és satisfactible ssi A és satisfactible.*
2. *$\mathcal{C} \models A$*

Si a l'aplicar l'algorisme no hem introduit cap constant o funció de Skolem es té que $A \equiv \mathcal{C}$. Aquest teorema és de gran utilitat a l'hora de fer demostracions per refutació, ja que estableix que per demostrar la insatisfactibilitat d'un enunciat n'hi ha prou amb demostrar la insatisfactibilitat de la seva forma clausal.

Exemple 4.1 Trobar la forma clausal del següent enunciat:

$$\exists x_1 P_1(x_1) \rightarrow (\forall x_1(P_2(x_1) \vee P_3(x_1)) \rightarrow \forall x_1 \exists x_2 P_4(x_1, x_2))$$

Apliquem l'algorisme anterior:

$$\begin{aligned}
Pas\ 1: &\quad \neg\exists x_1 P_1(x_1) \vee (\forall x_1(P_2(x_1) \vee P_3(x_1)) \rightarrow \forall x_1 \exists x_2 P_4(x_1, x_2)) \\
&\quad \neg\exists x_1 P_1(x_1) \vee \neg\forall x_1(P_2(x_1) \vee P_3(x_1)) \vee \forall x_1 \exists x_2 P_4(x_1, x_2) \\
Pas\ 2: &\quad \forall x_1 \neg P_1(x_1) \vee \exists x_1(\neg P_2(x_1) \wedge \neg P_3(x_1)) \vee \forall x_1 \exists x_2 P_4(x_1, x_2) \\
Pas\ 3: &\quad \forall x_1 \neg P_1(x_1) \vee \exists x_3(\neg P_2(x_3) \wedge \neg P_3(x_3)) \vee \forall x_4 \exists x_2 P_4(x_4, x_2) \\
Pas\ 4: &\quad \forall x_1 \neg P_1(x_1) \vee (\neg P_2(b_1) \wedge \neg P_3(b_1)) \vee \forall x_4 P_4(x_4, g_1(x_4)) \\
Pas\ 5: &\quad \forall x_1 \forall x_4(\neg P_1(x_1) \vee (\neg P_2(b_1) \wedge \neg P_3(b_1)) \vee P_4(x_4, g_1(x_4)) \\
Pas\ 6: &\quad \forall x_1 \forall x_4(\neg P_1(x_1) \vee \neg P_2(b_1) \vee P_4(x_4, g_1(x_4))) \wedge \\
&\quad (\neg P_1(x_1) \vee \neg P_3(b_1) \vee P_4(x_4, g_1(x_4)))) \\
Pas\ 7: &\quad \{\{\neg P_1(x_1), \neg P_2(b_1), P_4(x_4, g_1(x_4))\}, \\
&\quad \{\neg P_1(x_1), \neg P_3(b_1), P_4(x_4, g_1(x_4))\}\}
\end{aligned}$$

4.3 Teorema de Herbrand

En aquesta secció estudiarem l'anomenat teorema de Herbrand que és un dels resultats més importants en el camp de la demostració automàtica de teoremes, ja que redueix el problema de demostrar la insatisfactibilitat d'un enunciat de la lògica de predicats a demostrar la insatisfactibilitat d'un conjunt d'enunciats

de la lògica proposicional. Abans, però, cal donar els conceptes de domini de Herbrand i instàncies bàsiques. Un terme, una fòrmula atòmica, un literal o una clàusula que no conté variables diem que és un terme bàsic, un àtom bàsic, un literal bàsic o una clàusula bàsica. El *domini de Herbrand* per un conjunt de clàusules $\mathcal{C}$ és el conjunt de tots els termes bàsics que es poden formar a partir dels símbols de constant i de funció que apareixen a $\mathcal{C}$ (si no apareix cap constant, n'afegim una, per exemple a, per formar termes bàsics). Anem a definir-ho més formalment:

Definició 4.3 domini de Herbrand *Sigui $\mathcal{C}$ un conjunt de clàusules, $SC(\mathcal{C})$ els símbols de constant que apareixen a $\mathcal{C}$ i $SF(\mathcal{C})$ els símbols de funció que apareixen a $\mathcal{C}$. El domini de Herbrand $H(\mathcal{C})$ per a $\mathcal{C}$ es defineix com:*

$$\cup_{i=0}^{\infty} H_i(\mathcal{C})$$

on H_i es defineix com:

$$\begin{array}{lcl} H_0(\mathcal{C}) & = & \begin{cases} SC(\mathcal{C}) & si\ SC(\mathcal{C}) \neq \emptyset \\ \{a\} & si\ SC(\mathcal{C}) = \emptyset \end{cases} \\ H_{n+1}(\mathcal{C}) & = & H_n(\mathcal{C}) \cup \{f(t_1, \ldots, t_n) | f \in SF(\mathcal{C}), t_i \in H_n(\mathcal{C})\} \end{array}$$

Exemple 4.2 *Si $\mathcal{C} = \{\{P_1(a_1), P_2(f(x_1))\}, \{\neg P_2(x_2)\}\}$ llavors $SC(\mathcal{C}) = \{a_1\}$, $SF(\mathcal{C}) = \{f\}$ i*

$$\begin{array}{lcl} H_0 & = & \{a_1\} \\ H_1 & = & \{a_1, f(a_1)\} \\ H_2 & = & \{a_1, f(a_1), f(f(a_1))\} \\ & \vdots & \\ H_\infty & = & \{a_1, f(a_1), f(f(a_1)), f(f(f(a_1))), f(f(f(f(a_1)))), \ldots\} \end{array}$$

Definició 4.4 conjunt d'instàncies bàsiques *Sigui $\mathcal{C}$ un conjunt de clàusules. El conjunt d'instàncies bàsiques $IB(\mathcal{C})$ de $\mathcal{C}$ és el conjunt format per les clàusules bàsiques de $\mathcal{C}$ i per totes les clàusules bàsiques que es poden obtenir a partir de les clàusules de $\mathcal{C}$ substituint les variables per elements del domini de Herbrand de $\mathcal{C}$.*

Exemple 4.3 *Si $\mathcal{C} = \{\{\neg català(x), agrada(x, escudella)\}, \{català(joan)\}\}$ llavors*

$$\begin{array}{lcl} IB(\mathcal{C}) = & \{\{català(joan)\} \\ & \{\neg català(joan), agrada(joan, escudella)\} \\ & \{\neg català(escudella), agrada(escudella, escudella)\}\} \end{array}$$

$IB(\mathcal{C})$ serà un conjunt de cardinalitat infinita, excepte quan $\mathcal{C}$ no contingui símbols de funció o tots els literals que apareixen a $\mathcal{C}$ siguin bàsics. Noteu que un conjunt d'instàncies bàsiques $IB(\mathcal{C})$ el podem tractar com un conjunt de clàusules proposicionals, ja que els literals no contenen variables. Ho podem fer associant a cada àtom bàsic diferent que apareix a $IB(\mathcal{C})$ una variable proposicional diferent.

Teorema 4.2 Teorema de Herbrand *Un conjunt de clàusules $\mathcal{C}$ és insatisfactible ssi algun subconjunt finit de $IB(\mathcal{C})$ és insatisfactible.*

Noteu que el teorema de Herbrand redueix el problema de demostrar la insatisfactibilitat d'un conjunt de clàusules $\mathcal{C}$ que sigui insatisfactible a demostrar la insatisfactibilitat d'un subconjunt *finit* del conjunt de clàusules $IB(\mathcal{C})$. Per a demostrar la insatisfactibilitat d'un subconjunt finit de $IB(\mathcal{C})$ podem emprar, per exemple, el procediment de resolució estudiat pel cas proposicional ja que els àtoms que apareixen a $IB(\mathcal{C})$ no contenen variables.

Teorema 4.3 (Corolari) *Un conjunt de clàusules $\mathcal{C}$ és satisfactible ssi tot subconjunt finit de $IB(\mathcal{C})$ és satisfactible.*

Per aquest corolari, que és conseqüència immediata del teorema de Herbrand, es té que per demostrar la satisfactibilitat d'un conjunt de clàusules $\mathcal{C}$ que sigui satisfactible tindrem que demostrar la satisfactibilitat d'infinits subconjunts finits, sempre que $IB(\mathcal{C})$ sigui infinit.

4.4 Procediments de Herbrand

Basats en el teorema de Herbrand s'han trobat diferents procediments de demostració per refutació. Els més coneguts són els de Davis i Putnam i Gilmore. Tot seguit donarem un procediment de demostració senzill basat en el teorema de Herbrand. Donat un conjunt de clàusules $\mathcal{C}$, calculem $IB(\mathcal{C})$ i l'ordenem, obtenint un conjunt possiblement infinit de clàusules bàsiques:

$$IB(\mathcal{C}) = \{C_1, C_2, \ldots, C_n, \ldots\}$$

Aquest conjunt ordenat serà l'entrada al nostre procediment.

Procediment Herbrand
Entrada: $IB(\mathcal{C}) = \{C_1, C_2, \ldots, C_n, \ldots\}$
begin
$n := 0$
repeat
$n := n + 1$
until $\square \in Res^*(\{C_1, , C_2, \ldots, C_n\})$
return ("$\mathcal{C}$ insatisfactible")
end;

Aquest és un procedimet de *semi-decisió* ja que sempre s'atura quan $\mathcal{C}$ és insatsifactible i altrament no s'atura. Observeu que aquest procediment permet comprovar la insatisfactibilitat de qualsevol enunciat A de la lògica de predicats. En primer lloc, apliquem a A l'algorisme de transformació a forma clausal i obtenim un conjunt de clàusules $\mathcal{C}$ (pel teorema 4.1 sabem que per a demostrar la insatisfactibilitat de A n'hi ha prou amb demostrar la de $\mathcal{C}$). Llavors generem $IB(\mathcal{C})$ i apliquem el procediment. A la pràctica, en lloc de primer calcular $IB(\mathcal{C})$ (que normalment té cardinalitat infinita), el que fariem seria calcular el nivell i del domini de Herbrand (començant per $i = 0$), generariem les clàusules bàsiques que podem construir amb els termes obtinguts i comprovariem la satisfactibilitat d'aquestes clàusules bàsiques. Aquest procés s'aturaria en el cas que detectessim un subconjunt de $IB(\mathcal{C})$ insatisfactible, o en el cas que $\mathcal{C}$ fos satisfactible i el domini de Herbrand de $\mathcal{C}$ fos finit. En el cas que $\mathcal{C}$ fos satisfactible i el seu domini de Herbrand fos infinit el procediment no s'aturaria. Com

ja vàrem veure al parlar de la semidecidibilitat de la lògica de predicats, no podem construir un procediment capaç de detectar la satisfactibilitat de qualsevol conjunt de clàusules que sigui satisfactible. Per altra banda, cal notar que per comprovar la possible insatisfactibilitat d'un subconjunt d'instàncies bàsiques emprem el procediment de demostració de resolució proposicional, però també hauriem pogut emprar altres mètodes com els tableaux semàntics o les taules de veritat. La principal diferència entre els diferents procediments de demostració basats en el teorema de Herbrand, com són el Davis i Putnam o Gilmore, està en el procediments de demostració proposicional emprat. El principal inconvenient dels procediments de demostració basats en el teorema de Herbrand és que generen explícitament el conjunt d'instàncies bàsiques i això té un cost computacional gran. Al 1965 Robinson va trobar un procediment de demostració que arreglava aquest problema, la resolució per la lògica de predicats, que constitueix la base per la programació lògica.

4.5 Unificació

En aquesta secció estudiem els conceptes d'unificador i unificador més general, i donem un algorisme per trobar l'unificador més general d'un conjunt d'expressions. Tot i que nosaltres ho emprarem per treballar amb resolució, aquest concepte l'utilitzen altres procediments de demostració de la lògica de predicats.

Definició 4.5 unificador *Una substitució θ direm que és un unificador pel conjunt d'expressions $\{E_1, \ldots, E_k\}$ ssi $E_1\theta = \ldots = E_k\theta$. El conjunt d'expressions $\{E_1, \ldots, E_k\}$ direm que és unificable si té un unificador.*

Exemple 4.4 *El conjunt $\{P(a, y), P(x, f(b))\}$ és unificable ja que la substitució $\{a/x, f(b)/y\}$ és un unificador pel conjunt.*

Definició 4.6 unificador més general *Un unificador σ per un conjunt d'expressions $\{E_1, \ldots, E_k\}$ direm que és un unificador més general (umg) ssi per a cada unificador θ del conjunt d'expressions existeix una substitució τ tal que $\theta = \sigma \circ \tau$.*

Tot seguit donarem un algorisme, anomenat algorisme d'unificació, per trobar l'unificador més general d'un conjunt finit d'expressions simples. Definirem el concepte de conjunt de diferències emprat a l'algorisme.

Definició 4.7 conjunt de diferències *Sigui E un conjunt finit d'expressions simples. El conjunt de diferències D s'obté trobant la posició del primer símbol en el que no totes les expressions d'E coincideixen i treient de cada expressió de E la subexpressió que comença en aquella posició. El conjunt de totes aquestes subexpressions és el conjunt de diferències.*

Exemple 4.5 *Donat el següent conjunt d'expressions*

$$E = \{P_1(f_1(a_1), f_2(x_4), x_1), P_1(f_1(a_1), x_2, a_2), P_1(f_1(a_1), x_2, a_3)\}$$

el conjunt de diferències és $D = \{f_2(x_4), x_2\}$.

Algorisme d'unificació

Sigui E un conjunt finit d'expressions simples no buit.

1. $i = 0, \sigma_0 = \varepsilon, E_0 = E$
2. Si $|E_i| = 1$ llavors σ_i és l'umg de E i parar, sinó sigui D_i el conjunt de diferències de E_i.
3. Si existeixen elements $v_i, t_i \in D_i$ tals que v_i és una variable que no ocorre a t_i llavors anar a 4, sino E no és unificable i parar.
4. Sigui $\sigma_{i+1} = \sigma_i \circ \{t_i/v_i\}$ i $E_{i+1} = E_i\{t_i/v_i\}$
5. $i = i + 1$ i anar a 2

Exemple 4.6 Trobar un umg del següent conjunt d'expressions:

$$E = \{P_1(a_1, f_1(x_1), f_1(f_2(x_2))), P_1(x_3, f_1(x_3), f_1(x_4))\}$$

1. $\sigma_0 = \varepsilon$ i $E_0 = E$. Com la cardinalitat de E_0 és 2, σ_0 no és un umg i hem de calcular el conjunt de diferències D_0.
2. $D_0 = \{a_1, x_3\}$. Com $v_0 = x_3$ és una variable que no ocorre a $t_0 = a_1$, llavors

$$\begin{aligned} \sigma_1 &= \sigma_0 \circ \{t_0/v_0\} = \varepsilon \circ \{a_1/x_3\} = \{a_1/x_3\} \\ E_1 &= E_0\{t_0/v_0\} \\ &= \{P_1(a_1, f_1(x_1), f_1(f_2(x_2))), P_1(x_3, f_1(x_3), f_1(x_4))\}\{a_1/x_3\} \\ &= \{P_1(a_1, f_1(x_1), f_1(f_2(x_2))), P_1(a_1, f_1(a_1), f_1(x_4))\} \end{aligned}$$

3. Com la cardinalitat de E_1 és 2, σ_1 no és un umg i hem de calcular el conjunt de diferències D_1.
4. $D_1 = \{x_1, a_1\}$. Com $v_1 = x_1$ és una variable que no ocorre a $t_1 = a_1$, llavors

$$\begin{aligned} \sigma_2 &= \sigma_1 \circ \{t_1/v_1\} = \{a_1/x_3\} \circ \{a_1/x_1\} = \{a_1/x_3, a_1/x_1\} \\ E_2 &= E_1\{t_1/v_1\} \\ &= \{P_1(a_1, f_1(x_1), f_1(f_2(x_2))), P_1(a_1, f_1(a_1), f_1(x_4))\}\{a_1/x_1\} \\ &= \{P_1(a_1, f_1(a_1), f_1(f_2(x_2))), P_1(a_1, f_1(a_1), f_1(x_4))\} \end{aligned}$$

5. Com la cardinalitat de E_2 és 2, σ_2 no és un umg i hem de calcular el conjunt de diferències D_2.
6. $D_2 = \{f_2(x_2), x_4\}$. Com $v_2 = x_4$ és una variable que no ocorre a $t_2 = f_2(x_2)$, llavors

$$\begin{aligned} \sigma_3 &= \sigma_2 \circ \{t_2/v_2\} = \{a_1/x_3, a_1/x_1\} \circ \{f_2(x_2)/x_4\} = \\ &= \{a_1/x_3, a_1/x_1, f_2(x_2)/x_4\} \\ E_3 &= E_2\{t_2/v_2\} \\ &= \{P_1(a_1, f_1(a_1), f_1(f_2(x_2))), P_1(a_1, f_1(a_1), f_1(x_4))\}\{f_2(x_2)/x_4\} \\ &= \{P_1(a_1, f_1(a_1), f_1(f_2(x_2)))\} \end{aligned}$$

7. Com la cardinalitat de E_3 és 1, σ_3 és un umg.

Teorema 4.4 teorema d'unificació *Sigui E un conjunt finit d'expressions simples no buit. Si E és unificable l'algorisme acaba i la substitució calculada és un umg. En cas contrari, l'algorisme acaba i ens informa que E no és unificable.*

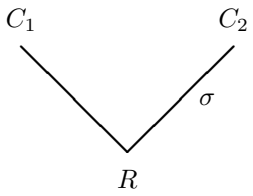
4.6 Resolució de primer ordre

Definició 4.8 Principi de resolució *Siguin C_1 i C_2 dues clàusules que no tenen variables en comú, $L = \{L_1^d, \ldots, L_m^d, L'_1, \ldots, L'_n\}$ un conjunt unificable de literals tals que $L_1, \ldots, L_m \in C_1 (m \geq 1)$ i $L'_1, \ldots, L'_n \in C_2 (n \geq 1)$, i σ un umg de L. La clàusula*

$$R = (C_1 - \{L_1, \ldots, L_m\})\sigma \cup (C_2 - \{L'_1, \ldots, L'_n\})\sigma$$

s'anomena resolvent de C_1 i C_2.

L'aplicació del principi de resolució a dues clàusules C_1 i C_2, anomenades *clàusules pare*, es representa gràficament de la següent manera:



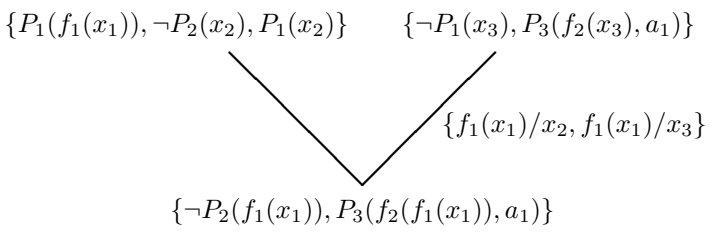
Exemple 4.7 Trobar una resolvent de les clàusules C_1 i C_2.

$$\begin{aligned} C_1 &= \{P_1(f_1(x_1)), \neg P_2(x_2), P_1(x_2)\} \\ C_2 &= \{\neg P_1(x_3), P_3(f_2(x_3), a_1)\} \end{aligned}$$

Sigui $L = \{P_1(f_1(x_1)), P_1(x_2), \neg P_1(x_3)\}$, on $P_1(f_1(x_1)) \in C_1$, $P_1(x_2) \in C_1$ i $\neg P_1(x_3) \in C_2$. La substitució $\sigma = \{f_1(x_1)/x_2, f_1(x_1)/x_3\}$ és un umg de L. Per tant, podem aplicar el principi de resolució i obtenim:

$$\begin{aligned} R &= (C_1 - \{P_1(f_1(x_1)), P_1(x_2)\})\sigma \cup (C_2 - \{\neg P_1(x_3)\})\sigma \\ &= \{\neg P_2(x_2)\}\sigma \cup \{P_3(f_2(x_3), a_1)\}\sigma \\ &= \{\neg P_2(f_1(x_1)), P_3(f_2(f_1(x_1)), a_1)\} \end{aligned}$$

La clàusula $\{\neg P_2(f_1(x_1)), P_3(f_2(f_1(x_1)), a_1)\}$ és una resolvent de C_1 i C_2.



Definició 4.9 prova *Siguin $\mathcal{C}$ un conjunt de clàusules i C' una clàusula. Una prova per resolució de C' a partir de $\mathcal{C}$ és una seqüència finita $C_1, C_2, \ldots, C_n$ de clàusules tals que cada C_i, $1 \leq i \leq n$, és una clàusula de $\mathcal{C}$ o és una resolvent de dues clàusules C_j, C_k, $1 \leq j, k < i$, i $C_n = C'$. Una prova de $\square$ a partir de $\mathcal{C}$ s'anomena una refutació de $\mathcal{C}$.*

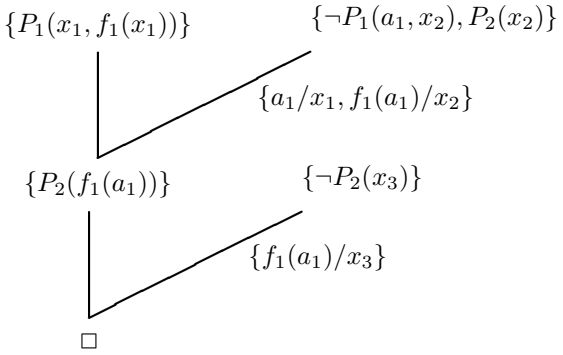
Si existeix una prova per resolució de C' a partir de $\mathcal{C}$ ho notem $\mathcal{C} \vdash_{Res} C'$. Quan no hi ha perill de confusió que el càlcul que emprem és el de resolució ho notem simplement $\mathcal{C} \vdash C'$.

Exemple 4.8 Trobar una refutació per resolució de $\mathcal{C}$.

$$\mathcal{C} = \{\{P_1(x_1, f_1(x_1))\}, \{\neg P_1(a_1, x_2), P_2(x_2)\}, \{\neg P_2(x_3)\}\}$$

$$\begin{array}{ll} C_1 = \{P_1(x_1, f_1(x_1))\} & (\textit{clàusula de } C) \\ C_2 = \{\neg P_1(a_1, x_2), P_2(x_2)\} & (\textit{clàusula de } C) \\ C_3 = \{P_2(f_1(a_1))\} & (\textit{resolvent de } C_1, C_2) \\ C_4 = \{\neg P_2(x_3)\} & (\textit{clàusula de } C) \\ C_5 = \square & (\textit{resolvent de } C_3, C_4) \end{array}$$

C_1, C_2, C_3, C_4, C_5 és una refutació de $\mathcal{C}$.



4.6.1 Completesa i solidesa

El següent teorema, anomenat teorema de resolució, estableix les propietats de solidesa i completesa del càlcul de resolució pel CP_1.

Teorema 4.5 *Sigui $\mathcal{C}$ un conjunt de clàusules. $\mathcal{C}$ és insatisfactible ssi existeix una refutació per resolució de $\mathcal{C}$.*

4.6.2 Refinaments

Des d'un punt de vista lògic hem vist que la resolució és un càlcul adient ja que té les propietats de solidesa i completesa. Ara bé, des del punt de vista de les aplicacions pràctiques presenta alguns problemes alhora d'implementar demostradors automàtics de teoremes. Per una banda, tenim l'obstacle que donat un conjunt de clàusules a refutar, en general, hi ha moltes possibles clàusules que permeten derivar una nova resolvent, i entre totes aquestes posibles aplicacions del principi de resolució sols n'hi ha unes quantes que ens permeten derivar la clàusula buida. Per altra banda, la resolució és semidecidible pel CP_1

i, per tant, la reiterada aplicació del principi de resolució pot produir infinites resolvents. Per subsanar aquests problemes ha sorgit el concepte de refinament. Distinguim entre dos tipus de refinaments: les estratègies i les restriccions. Les estratègies són regles heurístiques que, quan tenim varis possibles parells de clàusules als que podem aplicar el principi de resolució, determinen en quin ordre ho hem de fer. Les estratègies no redueixen l'espai de búsqueda, però, en general, fan que haguéssim de recorrer una part més petita de l'espai de búsqueda abans de trobar una solució. Un exemple és l'estratègia de preferència unitaria. Aquesta estratègia estableix que, sempre que sigui possible, al aplicar el principi de resolució una de les clàusules sigui unitaria, o sigui, que tingui un sol literal. Les restriccions prohibeixen l'aplicació del principi de resolució a clàusules que no tinguin una determinada forma sintàctica. Per tant, en aquest cas reduïm el nombre de possibles resolvents i l'espai de búsqueda. El perill de les restriccions es que podem perdre la propietat de completesa al no generar totes les possibles resolvents. Un exemple de restricció és la N- resolució (P-resolució) que sols permet aplicar el principi de resolució a dues clàusules quan al menys una de les clàusules sols té literals negats (no negats). Tant la N-resolució com la P-resolució són completes. Altres exemples de restriccions són la resolució amb conjunt suport, la resolució semàntica, la resolució unitària i la resolució lineal.

Exercicis

Exercici 43 Trobeu la forma clausal dels enunciats següents:

1. $\exists x_2 \forall x_3 \forall x_4 \exists x_1 P_1(x_1, f_2(x_2, x_3, x_4))$
2. $\neg(\forall x_1 P_1(x_1) \rightarrow \exists x_1 \forall x_2 P_2(x_1, x_2))$
3. $(\forall x_1 \exists x_2 P_1(x_1, x_2) \wedge \exists x_3 \forall x_4 (P_1(x_3, x_4) \rightarrow P_2(x_4))) \rightarrow \exists x_2 P_2(x_2)$

Exercici 44 Trobeu el domini de Herbrand pels conjunts de clàusules següents:

1. $\{\{\neg P_1(x_1), P_2(f(x_1, x_2))\}\}$
2. $\{\{\neg P_1(x_1), P_2(x_1, x_2)\}, \{P_2(a_1), P_3(a_1, a_2)\}\}$
3. $\{\{\neg P_1(x_1), P_2(f(x_1, x_2))\}, \{P_2(a_1), P_3(a_2, a_2)\}\}$
4. $\{\{\neg P_1(x_1, f(x_1))\}, \{\neg P_1(a_1, x_1), P_2(x_1)\}, \{\neg P_2(a_2)\}\}$

Exercici 45 Trobeu el conjunt d'instàncies bàsiques dels conjunts de clàusules següents:

1. $\mathcal{C} = \{\{sencer(0)\}, \{\neg sencer(x), sencer(succ(x))\}\}$
2. $\{\{\neg P_1(x_1), P_2(x_1, x_2)\}, \{P_2(a_1), P_3(a_1, a_2)\}\}$
3. $\{\{\neg P_1(x_1), P_2(f(x_1, x_2))\}, \{P_2(a_1), P_3(a_2, a_2)\}\}$
4. $\{\{\neg P_1(x_1, f(x_1))\}, \{\neg P_1(a_1, x_1), P_2(x_1)\}, \{\neg P_2(a_2)\}\}$

Exercici 46 Expresseu els enunciats següents com enunciats del CP_1 i obteniu la seva forma clausal.

1. Tots els divisors de 2 són diferents de zero.
2. No tot número és igual a zero.
3. Tot número no primer té algun divisor.

Exercici 47 Trobeu un umg pels següents conjunts d'expressions que siguin unificables:

1. $\{P_1(f(x_1), a_3), P_1(x_2, f(x_3))\}$
2. $\{P_1(f(x_1), f(a_3)), P_1(f(x_2), f(x_3))\}$
3. $\{P_2(a_1, x_1, f_1(f_2(x_2))), P_2(x_2, f_1(x_3), f_1(x_3))\}$
4. $\{P_1(f_1(x_1), x_2, f_2(x_3)), P_1(x_4, x_4, x_5)\}$
5. $\{P_1(f_1(x_1), x_2, f_2(x_3)), P_1(x_5, x_4, x_5)\}$
6. $\{P_3(a_5, x_1, f_1(f_2(x_4))), P_3(x_2, f_4(x_2, x_3), f_1(x_3))\}$

Exercici 48 Trobeu, si existeix, una resolvent de les clàusules $\{P(x_1, x_1)\}$ i $\{\neg P(f(x_2), x_2)\}$.

Exercici 49 Trobeu totes les possibles resolvents entre les clàusules C_1 i C_2.

$$\begin{aligned} C_1 &= \{\neg P_1(f_1(a_1), f_2(x_1, a_2)), P_2(x_2, x_1), \neg P_1(x_2, x_3)\} \\ C_2 &= \{P_1(f_1(x_2), f_2(a_1, a_2)), \neg P_2(a_1, a_2), \neg P_2(f_1(a_1), b_1)\} \end{aligned}$$

Exercici 50 Tots els artistes són uns presumits. Per tant, la casa d'un artista és la casa d'un presumit.
Demostreu per un procediment de Herbrand i per resolució de primer ordre que l'argument anterior és vàlid (empreu els següents predicats: A(x): x és un artista, P(x): x és un presumit, C(x,y): x és la casa de y).

Exercici 51 Siguin R_1 una relació binaria antisimètrica i R_2 una relació binaria simètrica. Definim una relació binaria R_3 de la següent manera:

$$\forall x_1 \forall x_2 (R_3(x_1, x_2) \leftrightarrow (R_1(x_1, x_2) \vee R_2(x_1, x_2)))$$

demostreu, emprant resolució, que R_3 és antisimètrica.

Exercici 52 Si dos carrers són perpendiculars a un tercer llavors són paral.lels entre si. Si un carrer és perpendicular a un altre, aquest és perpendicular del primer. Per tant, si dos carrers no són paralels no n'hi ha cap que sigui perpendicular a tots dos.

Capítol 5

Programació Lògica

5.1 Introducció

Un algoritme conté tant la representació del coneixement necessari per a solucionar un problema com la manera de manipular aquest coneixement per tal d'obtenir una solució. La idea de la Programació lógica és que el programador representi aquest coneixement mitjançant un conjunt finit d'enunciats, que constitueixen una especificació del problema, i que un procediment de prova manipuli aquest coneixement per trobar una solució. D'aquesta forma, el programador comunica a l'ordinador qué ha de fer i el procediment de prova se n'encarrega de com fer-ho. Entre els avantatges del paradigma de programació lògica està el fet de que programem de manera independent de la màquina. Els programes lògics están formats per enunciats que expressen coneixement sobre el problema a resoldre en lloc d'instruccions que han d'esser seguides pas a pas com passa amb els llenguatges de programació imperatius. Un programa és una especificació amb el llenguatge de la lògica i la seva correcció es deriva de la del procediment de prova emprat. Dit d'una altra manera, un programa és una especificació executable. El llenguatge de programació lògica més conegut és el Prolog. El Prolog utilitza el subconjunt del llenguatge de primer ordre de les clàusules de Horn i un procedimet de prova basat en un refinament del càlcul de resolució, anomenat resolució SLD. En aquest capítol estudiarem els fonaments teòrics de l'anomenat Prolog pur.

5.2 Programes lògics

En programació lògica una clàusula de la forma

$$\forall x_1 \ldots \forall x_n (P_1 \vee \ldots \vee P_k \vee \neg Q_1 \vee \ldots \vee \neg Q_m)$$

on $P_1, \ldots, P_k, Q_1, \ldots, Q_m$ són àtoms i $x_1, \ldots, x_n$ són les variables que ocorren en aquests àtoms, es representa per

$$P_1, \ldots, P_k \leftarrow Q_1, \ldots, Q_m$$

En aquesta notació, les comes del antecedent representen conjuncions i les comes del conseqüent representen disjuncions. A més a més, es sobreentén que les variables estan quantificades universalment.

Definició 5.1 clàusula de Horn *Una clàusula que conté com a màxim un literal positiu s'anomena clàusula de Horn.*

Les clàusules de Horn poden ser dels següents tipus:

- **regles** hi ha un literal positiu i un o més literals negatius

  $$\{P, \neg Q_1, \ldots, \neg Q_m\}$$

  Les regles corresponen a enunciats de la forma $\forall((Q_1 \wedge \ldots \wedge Q_n) \rightarrow P)$. En programació lògica es representen per $P \leftarrow Q_1, \ldots, Q_n$. P s'anomena el cap i $Q_1, \ldots, Q_n$ el cos.

- **fets** hi ha sols un literal positiu

  $$\{P\}$$

  Els fets corresponen a enunciats de la forma $\forall P$. En programació lògica es representen per $P \leftarrow$.

- **objectius** no hi ha cap literal positiu i hi ha un o més literals negatius

  $$\{\neg Q_1, \ldots, \neg Q_m\}$$

  Els objectius corresponen a enunciats de la forma $\neg\exists(Q_1 \wedge \ldots \wedge Q_n)$. En programació lògica es noten per $\leftarrow Q_1, \ldots, Q_n$.

- **objectiu buit** no hi ha cap literal. Es representa per la clàusula buida $\square$.

Definició 5.2 programa lògic *Un programa lògic és un conjunt finit de regles i fets, anomenats clàusules del programa.*

Exemple 5.1 El següent conjunt de clàusules és un programa lògic que permet sumar dos nombres naturals si interpretem la constant 0 com el zero dels naturals, la funció *succ* com la funció successor i el predicat *suma* com la relació en que el tercer argument és la suma dels dos arguments anteriors.

1. $suma(x, 0, x) \leftarrow$
2. $suma(x, succ(y), succ(z)) \leftarrow suma(x, y, z)$

A cada clàusula del programa li assignem un nombre per poder-la referenciar.

5.3 Resolució SLD

En aquesta secció estudiarem un refinament de la resolució anomenat resolució SLD. Aquest refinament l'aplicarem a clàusules de Horn i es caracteritza perquè a cada pas de resolució una de les clàusules pare és la resolvent obtinguda al pas anterior. A més a més, tenim una funció de selecció que escolleix l'àtom a eliminar en cada pas de resolució.

Definició 5.3 funció de selecció *Una funció de selecció és una funció amb domini un conjunt d'objectius i amb rang un conjunt d'àtoms tal que el valor de la funció per un objectiu és un àtom de l'objectiu, anomenat àtom seleccionat.*

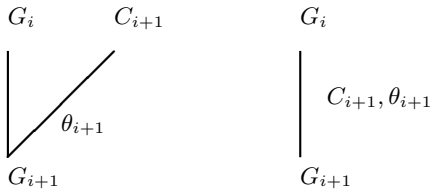
La idea de la funció de selecció és escollir l' àtom que eliminarem a l'aplicar el principi de resolució, evitant l'indeterminisme que es produeix quan podem escollir entre varis àtoms. Exemples típics de funció de selecció són la que agafa l'àtom de més a la dreta o de més a l'esquerra.

Definició 5.4 SLD-resolvent *Siguin G_i un objectiu de la forma $\leftarrow P_1, \ldots, P_m, \ldots, P_k (k > 0)$, i C_{i+1} una clàsula d'un programa lògic de la forma $P \leftarrow Q_1, \ldots, Q_q (q \geq 0)$ de manera que G_i i C_{i+1} no tenen variables en comú, R una funció de selecció, P_m l'àtom que retorna R quan l'apliquem a G_i i θ_{i+1} un umg de P_m i P. L'objectiu*

$$\leftarrow (P_1, \ldots, P_{m-1}, Q_1, \ldots, Q_q, P_{m+1}, \ldots, P_k)\theta_{i+1}$$

s'anomena resolvent SLD de G_i i C_{i+1}.

Aquesta situació la podem representar gràficament per qualsevol d'aquests dos gràfics:



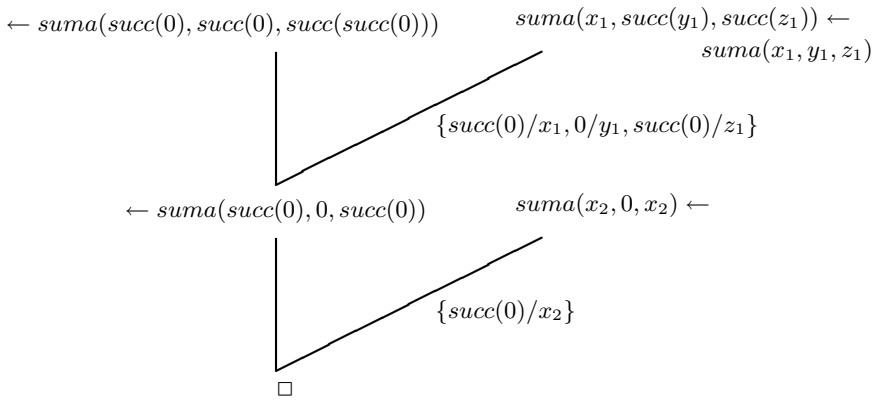
Per trobar una resolvent les clàusules pare no tenen que tenir variables en comú. Per aconseguir-ho n'hi ha prou amb renombrar una de les clàusules pare. Nosaltres sempre aplicarem el renombrament a la clàusula del programa i no a l'objectiu. Això s'anomena resolució SLD standaritzada.

Definició 5.5 SLD-derivació *Siguin P un programa, G un objectiu i R una funció de selecció. Una SLD-derivació de $P \cup \{G\}$ via R consisteix de una seqüència (finita o infinita) $G_0 = G, G_1, \ldots$ d'objectius, una seqüència $C_1, C_2, \ldots$ de variants de clàusules del programa P i una seqüència $\theta_1, \theta_2, \ldots$ de umg's tals que cada G_{i+1} és una resolvent de G_i i C_{i+1} usant θ_{i+1} via R.*

Definició 5.6 SLD-refutació *Una SLD-refutació de $P \cup \{G\}$ via R és una SLD-derivació on les seqüències d'objectius, clàusules i umg's són finites i el darrer objectiu és $\square$.*

Les SLD-proves poden ser finites o infinites, depenent de la finitud o infinitud de les seqüències d'objectius, variants de clàusules de programa i unificadors. Una SLD- derivació finita direm que és exitosa si el darrer objectiu és la clàusula buida, o sigui, una SLD-derivació que és una refutació. Una SLD-derivació finita direm que és fallida si no acaba amb la clàusula buida i l'àtom que retorna la funció de selecció no unifica amb el cap de cap clàusula del programa.

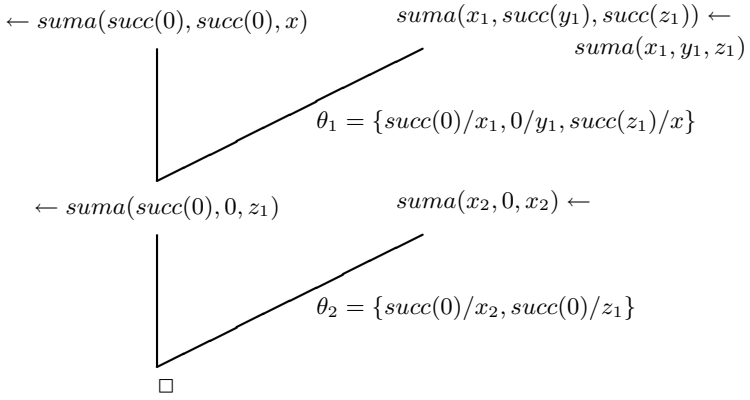
Exemple 5.2 *Donat el programa de la suma de naturals i l'objectiu $\leftarrow suma(succ(0), succ(0), succ(succ(0)))$ és té que la següent derivació és una refutació.*



Donat un objectiu i un programa, quan trobem una refutació, tenim que la negació de l'objectiu és una conseqüència lògica del programa ja que, com veurem més endavant, la resolució SLD té les propietats de solidesa i completesa. Recordem que la negació d'un objectiu correspon a un enunciat de la forma $\exists(Q_1 \wedge \ldots \wedge Q_n)$, on les variables lliures de l'objectiu queden existencialment quantificades. La resolució SLD, a més a més, ens permet trobar les instàncies d'aquestes variables, que al aplicar-les a l'enunciat obtenim un nou enunciat que és conseqüència lògica del programa.

Definició 5.7 resposta generada *Siguin P un programa, G un objectiu i R una funció de selecció. Una resposta generada via R és una substitució θ obtinguda al restringir la substitució $\theta_1 \circ \ldots \circ \theta_n$ a les variables de G, on $\theta_1, \ldots, \theta_n$ és la sequència de umg's obtinguda a una SLD- refutació de $P \cup \{G\}$ via R.*

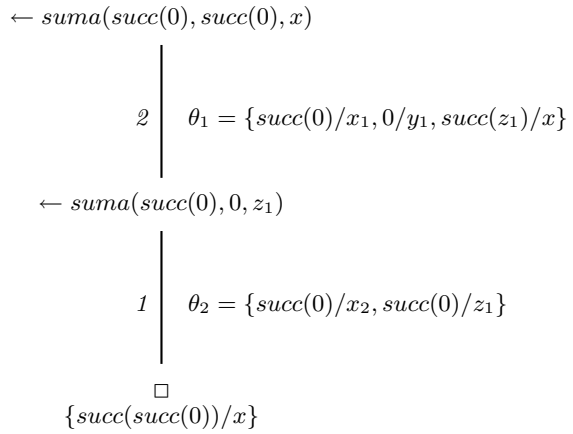
Exemple 5.3 *Considerem el programa de suma de naturals i l'objectiu $\leftarrow suma(succ(0), succ(0), x)$. Tot seguit donem una refutació i una resposta generada.*



Noteu que per poder aplicar el principi de resolució SLD hem tingut que renombrar les variables de les clàusules del programa. La composició de les substitucions θ_1 i θ_2 és:

$$\theta_1 \circ \theta_2 = \{succ(0)/x_1, 0/y_1, succ(succ(0))/x, succ(0)/x_2, succ(0)/z_1\}$$

Com la única variable que apareix a l'objectiu és x, la resposta generada és $\{succ(succ(0))/x\}$. Aquesta situació la podem representar de manera més compacta de la forma:



5.3.1 Completesa i solidesa de la resolució SLD

Teorema 5.1 solidesa *Siguin P un programa, R una funció de selecció, $\leftarrow A_1, \ldots, A_m$ un objectiu pel qual existeix una refutació via R i θ una resposta generada. Llavors,*

$$P \models \forall((A_1 \wedge \ldots \wedge A_m)\theta)$$

Teorema 5.2 completesa *Siguin P un programa, R una funció de selecció i $\leftarrow B$ un objectiu. Per a cada instància B' de B que és una conseqüència lògica de P existeix una refutació de $\leftarrow B$ via R amb resposta generada θ tal que B' és una instància de $B\theta$.*

5.4 Procediments de refutació

Noteu que donat un programa, un objectiu i una funció de selecció, l'àtom seleccionat es pot unificar amb el cap de més d'una clàusula del programa. No obstant, el número de clàusules diferents amb les que pot unificar és finit, ja que un programa conté un nombre finit de clàusules. Si a cada pas trobem totes les posibles resolvents podem construir un arbre, on les branques infinites corresponen a proves infinites. Entre les branques finites distingim les que acaben en la clàusula buida, que són les proves exitoses o refutacions, i les que acaben amb una clàusula no buida que són les proves fallides, per les quals ja no podem trobar més resolvents. Les respostes generades per les refutacions ens donen consequències lògiques del programa al aplicar-les al objectiu.

Definició 5.8 arbre de resolució SLD *Siguin P un programa, G un objectiu i R una funció de selecció. Un SLD-arbre per $P \cup \{G\}$ via R es defineix de la següent manera:*

1. *l'arrel de l'arbre és G.*
2. *cada node de l'arbre és un objectiu, eventualment buit.*

3. *Siguin* $\leftarrow A_1, \ldots, A_m, \ldots, A_k (k \geq 1)$ *un node de l'arbre i* A_m *l'àtom seleccionat per R. Aquest node té un descendent per cada clàusula de P de la forma* $A \leftarrow B_1, \ldots, B_q$ *tal que* A_m *i A són unificables. El descendent és*

$$\leftarrow (A_1, \ldots, A_{m-1}, B_1, \ldots, B_q, A_{m+1}, \ldots, A_k)\theta$$

on θ *és un umg de* A_m *i A.*

4. *els nodes que són la clàusula buida no tenen descendents.*

Cada branca d'un arbre SLD és una derivació de $P \cup \{G\}$. Una branca és exitosa si correspon a una derivació exitosa, és fallida si correspon a una derivació fallida i és infinita si correspon a una derivació infinita.

Exemple 5.4 El següent programa lògic permet saber si hi ha un camí entre dos punts d'un graf dirigit. La primera clàusula expressa que s'hi ha una aresta que uneix 2 punts, x i z, llavors hi ha un camí entre aquests dos punts. La segona clàusula expressa que si hi ha una aresta que uneix dos punts, x i y, i hi ha un camí de y a z llavors hi ha un camí entre x i z. Les dues darreres clàusules expressen que hi ha una aresta que uneix a i b, i una aresta que uneix b i c.

1. $cami(x, z) \leftarrow aresta(x, z)$
2. $cami(x, z) \leftarrow aresta(x, y), cami(y, z)$
3. $aresta(a, b) \leftarrow$
4. $aresta(b, c) \leftarrow$

La figura 5.1 mostra l'arbre de resolució SLD per aquest programa, on l'objectiu és $\leftarrow cami(a, z)$ i la funció de selecció retorna l'àtom de més a l'esquerra.

Definició 5.9 procediment de refutació SLD *Un procediment de refutació SLD ve donat per una funció de selecció i una estratègia per trobar branques exitoses a un arbre SLD, anomenada regla de cerca.*

El Prolog és un procediment de refutació que utilitza la funció de selecció que sempre selecciona l'àtom de més a l'esquerra i l'estrategia de fondaria prioritària com a regla de cerca. Des del punt de vista de trobar totes les refutacions, quan l'arbre té branques infinites, és millor l'estratègia d'amplada prioritària. Ara bè, sacrifiquem la completesa del procediment en ares de la eficiència. O sigui, si treballem en amplada prioritària podem trobar en temps finit totes les refutacions, mentre que si treballem en fondaria prioritària tenim el perill de recorrer una branca infinita abans de trobar totes les refutacions. En aquest cas el procediment diem que no és complet.

Exercicis

Exercici 53 Expressar els següents enunciats com clàusules de Horn:

1. $\forall x(p(x) \vee \neg q(x))$
2. $\forall x(p(x) \rightarrow (q(x) \rightarrow r(x)))$
3. $\forall x(p(x) \vee \neg \exists y(q(x, y) \wedge r(x)))$
4. $\forall x(\neg p(x) \vee (q(x) \rightarrow r(x)))$

Exercici 54 Demostrar que tot conjunt insatisfactible de clàusules de Horn conté almenys una clàusula unitària positiva i almenys una clàusula negativa.

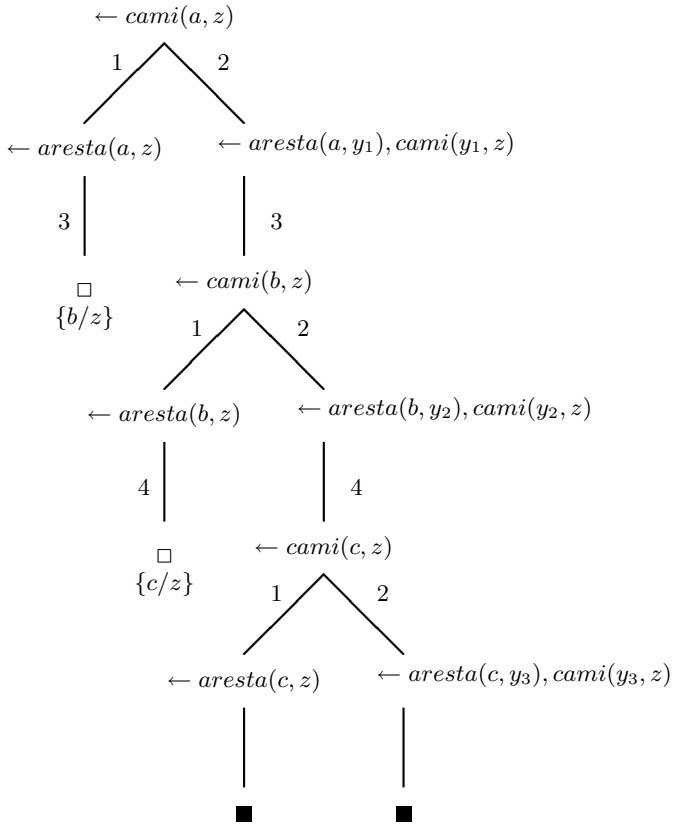


Figura 5.1: arbre de resolució SLD

Exercici 55 Es pot utilitzar el programa de l'exemple 5.1 per restar?

Exercici 56 Afegir noves clàusules al programa de l'exemple 5.1 per obtenir un programa que permeti fer el producte de dos nombres naturals.

Exercici 57 Afegir noves clàusules al programa de l'exercici 56 per obtenir un programa que permeti fer l'exponenciació de dos nombres naturals.

Exercici 58 Trobar l'arbre de resolució SLD, usant la funció de selecció que retorna l'àtom de més a la dreta, pel programa del camí d'un graf i l'objectiu $\leftarrow cami(a, z)$.

Exercici 59 Donat el programa de l'exercici 56 i l'objectiu $\leftarrow producte(succ(succ(0)), succ(succ(0)), x)$ donar una refutació i la resposta generada per aquesta refutació.

Exercici 60 El següent programa lògic concatena dues llistes:

$$concatenar(nil, X, X)$$
$$concatenar(cons(V, X), Y, cons(V, Z)) \leftarrow concatenar(X, Y, Z)$$

nil denota la llista buida i $cons(X, Y)$ és una funció que afegeix l'element X a la llista Y. Usant aquesta notació, la llista $[1, 2]$ es representa per $cons(1, cons(2, nil))$ i la llista [3] es representa per $cons(3, nil)$. Calculeu, usant l'anterior programa, la llista que s'obté al concatenar les llistes $[1, 2]$ i [3].

Exercici 61 Donat el següent programa:

$$p(y) \leftarrow q(x, y), r(y)$$
$$p(x) \leftarrow q(x, x)$$
$$q(x, x) \leftarrow s(x)$$
$$r(b) \leftarrow$$
$$s(a) \leftarrow$$
$$s(b) \leftarrow$$

Trobar l'arbre SLD per l'objectiu $\leftarrow p(x)$ usant com a funció de selecció la que agafa l'àtom de més a l'esquerra. Trobar les respostes generades. Dir quina regla de cerca és millor per aquest cas.

Exercici 62 Donat el següent programa:

$$p(x, y) \leftarrow q(x, z), r(z, y)$$
$$p(x, y) \leftarrow r(y, x)$$
$$q(x, y) \leftarrow r(y, x)$$
$$r(a, b) \leftarrow$$

Trobar una refutacio de $\leftarrow p(b, x)$ usant la funció de selecció de més a la dreta. Trobar l'arbre SLD per l'objectiu $\leftarrow p(b, x)$. Dir quina regla de cerca és millor per aquest cas.

Exercici 63 Donat el següent programa:

$$p(x, z) \leftarrow p(y, z), q(x, y)$$
$$p(x, x) \leftarrow$$
$$q(x, y) \leftarrow r(y, x)$$
$$q(a, b) \leftarrow$$

Trobar totes les refutacions de l'objectiu $\leftarrow p(x, b)$ usant la funció de selecció de més a l'esquerra i calcular les respostes generades per aquestes refutacions.

Exercici 64 El següent programa lògic permet calcular derivades:

$$derivada(t, 1) \leftarrow$$
$$derivada(U + V, U' + V') \leftarrow derivada(U, U'), derivada(V, V')$$
$$derivada(U * V, U * V' + V * U') \leftarrow derivada(U, U'), derivada(V, V')$$
$$derivada(sin(U), U' * cos(U)) \leftarrow derivada(U, U')$$

calcular la derivada de $t * sin(t)$, usant resolució SLD prenent com a funció de selecció la que escolleix l'àtom de més a l'esquerra. Observacions: 1 i t són símbols de constant, U, V, U' i V' són símbols de variable, sin, cos, $*$ i $+$ són símbols de funció i *derivada* és un símbol de predicat. Les funcions $*$ i $+$ les expressem en notació infixa.

Bibliografia

[AAD77] J. Allwood, L. G. Andersson, and O. Dahl. *Logic in Linguistics*. Cambridge University Press, 1977.

[AP93] G. Aguilera and I. Pérez de Guzman. *Lógica para la Computación I.* Editorial Agora, 1993.

[Apt90] K. R. Apt. Logic programming. In J. Van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 493–574. Elsevier, Amsterdam, 1990.

[Bra90] I. Bratko. *Prolog Programming for Artificial Intelligence (2nd. ed.)*. Addison-Wesley, 1990.

[Bun83] Alan Bundy. *The Computer Modelling of Mathematical Reasoning*. Academic Press. Inc., 1983.

[Chu36] A. Church. A note on the entscheidungsproblem. *Journal of Symbolic Logic*, 1:40–41, 1936.

[CL73] C. L. Chang and R. C. T. Lee. *Symbolic Logic and Mechanical Theorem Proving*. Academic Press, Inc., 1973.

[Cla78] K. L. Clark. Negation as failure. In H. Gallaire and J. Minker, editors, *Logic and Data Bases*, pages 293–322. Plenum Press, New York, 1978.

[CM81] W. Clocksin and C. Mellish. *Programming in Prolog*. Springer-Verlag, 1981.

[Cue85] J. Cuena. *Lógica Informática*. Alianza Editorial, Madrid, 1985.

[Dea74] A. Deaño. *Introducción a la Lògica Formal*. Alianza Universidad, 1974.

[DP60] M. Davis and H. Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7:201–215, 1960.

[End72] H. B. Enderton. *A Mathematical Introduction to Logic*. Academic Press, New York, 1972.

[Fit90] M. Fitting. *First-Order Logic and Automated Theorem Proving*. Springer-Verlag, New York, 1990.

[Gal86] J. H. Gallier. *Logic for Computer Science: Foundations of Automatic Theorem Proving*. Harper and Row, New York, 1986.

[Gal90] A. Galton. *Logic for Information Technology*. John Wiley and Sons, 1990.

[Gar86] M. Garrido. *Lógica Simbólica*. Editorial Tecnos, Madrid, 1986.

[Gil60] P. C. Gilmore. A proof method for quantification theory: its justification and realization. *IBM J. Res. Develop.*, 4:28–35, 1960.

[GN87] M. R. Genesereth and N. J. Nilsson. *Logical Foundations of Artificial Intelligence*. Morgan Kaufmann Publishers, Inc., 1987.

[Göd30] K. Gödel. Die vollständigkeit der axiome des logischen funktionenkalküls. *Monatshefte für Mathematik und Physik*, 37:349–360, 1930.

[Göd31] K. Gödel. Über formal unentscheidbare stätze der principia mathematica und verwandter systeme i. *Monatshefte für Mathematik und Physik*, 38:173–198, 1931.

[Ham78] A. G. Hamilton. *Logic for Mathematicians*. Cambridge University Press, 1978.

[Her30] J. Herbrand. Recherches sur la théorie de la démonstration. *Travaux de la Societé des Sciences et des Lettres de Varsovie*, 33, 1930.

[Hod77] W. Hodges. *Logic*. Penguin Books, London, 1977.

[Hod83] W. Hodges. Elementary predicate logic. In D. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic*. D. Reidel Publishing Company, 1983.

[Hof87] D. R. Hofstadter. *Gödel, Escher, Bach. Un Eterno y Grácil Bucle*. Tusquets Editores, 1987.

[Hog90] C. J. Hogger. *Essentials of Logic Programming*. Oxford University Press, 1990.

[JRvH89] P. Jackson, H. Reichgelt, and F. van Harmelen. *Logic-Based Knowledge Representation*. The MIT Press, Cambridge, MA., 1989.

[Kle67] S. C. Kleene. *Mathematical Logic*. John Wiley and Sons, 1967.

[Kow79] R. Kowalski. *Logic for Problem Solving*. Elsevier North-Holland, Amsterdam, 1979.

[Kum93] S. Kumar. *Deductive Databases and Logic Programming*. Addison-Wesley, 1993.

[Lei92] A. Leitsch. *Resolution Theorem Proving*. Fourth European Summer School in Logic, Language and Information. University of Essex, Colchester, 1992.

[Llo87] J.W. Lloyd. *Foundations of Logic Programming.* Springer-Verlag, second edition, 1987.

[Lov78] D. W. Loveland. *Automated Theorem Proving: A Logical Basis.* North Holland, New York, 1978.

[LRA92] J. Leach and M. Rodríguez-Artalejo. *Lógica Matemática. Notas del Curso.* Facultad de Matemáticas. Universidad Complutense, Madrid, 1992.

[Man89] M. Manzano. *Teoría de Modelos.* Alianza Editorial, 1989.

[Man93] F. Manyà. *Notes de Lògica.* Departament d' Informàtica i Enginyeria Industrial. Universitat de Lleida, 1993.

[Men87] E. Mendelson. *Introduction to Mathematical Logic 3rd. edn.* Wadsworth and Brooks, California, 1987.

[MW85] Z. Manna and R. Waldinger. *The Logical Basis for Computer Programming.* Addison-Wesley, 1985.

[NM90] U. Nilsson and J. Małuszyński. *Logic, Programming and Prolog.* John Wiley and Sons, 1990.

[Pla91] J. Pla. *Lliçons de Lògica Matemàtica.* Promociones y Publicaciones Universitarias, Barcelona, 1991.

[Pra60] D. Prawitz. An improved proof procedure. *Theoria*, 26:102–139, 1960.

[Ram88] A. Ramsay. *Formal Methods in Artificial Intelligence.* Cambridge University Press, 1988.

[RC90] S. Reeves and M. Clarke. *Logic for Computer Science.* Addison-Wesley, 1990.

[Rob65] J. A. Robinson. A machine oriented logic based on the resolution principle. *J. ACM*, 12(1):23–41, 1965.

[Rob91] D. Robertson. *Quick Prolog.* Department of Artificial Intelligence. University of Edinburgh, 1991.

[Rob92] J. A. Robinson. Logic and logic programming. *Communications of the ACM*, 35(3):40–65, 1992.

[SA91] V. Sperschneider and G. Antoniou. *Logic: A Foundation for Computer Science.* Addison-Wesley, 1991.

[Sal90] T. Sales. *Apunts de Lògica Matemàtica.* Centre Publicacions d'Abast, Facultat de Informàtica de Barcelona, 1990.

[Sch89] U. Schöning. *Logic for Computer Scientists.* Birkhäuser, Boston, 1989.

[Sco81] D. Scott. *Notes on the Formalization of Logic.* Sub-faculty of Philosophy, Oxford, 1981.

[Sho67] J. R. Shoendfield. *Mathematical Logic.* Adisson-Wesley, 1967.

[Smu68] R. M. Smullyan. *First Order Logic.* Springer-Verlag, 1968.

[Smu78] R. M. Smullyan. *What is the name of this Book ?* Prentice-Hall, Inc., 1978.

[SS86] L. Sterling and E. Shapiro. *The Art of Prolog.* The MIT Press, Cambridge, MA., 1986.

[TBA+90] A. Tucker, B. Barnes, R. Aiken, K. Barker, K. Bruce, T. Cain, S. Conry, G. Engel, R. Epstein, D. Lidtke, M. Mulder, J. Rogers, E. Spafford, and A. Turner. *Computing Curricula 1991.* ACM, IEEE Computer Society Press, 1990.

[Tur36] A. M. Turing. On computable numbers, with an application to the entscheidungsproblem. *Proc. London Math. Soc.*, 42:230–265, 1936.

[vD89] D. van Dalen. *Logic and Structure (2nd. ed.).* Springer-Verlag, 1989.