Curso de Lógica
1998

Roberto Cignoli y Guillermo Martínez

Chapter 1

Cálculo Proposicional

1.1 El lenguaje del cálculo proposicional

Una *proposición* es un enunciado del que tiene sentido decir que es verdadero o falso. Es decir, una proposición es lo que en gramática se llama una *oración declarativa* (en contraste con las oraciones interrogativas, exclamativas o imperativas).

Una oración debe ser expresada en un cierto lenguaje, oral o escrito, y su forma dependerá de las leyes gramaticales de ese lenguaje.

Pero en general, las leyes gramaticales de los lenguajes que usamos corrientemente no son suficientes para caracterizar completamente a las proposiciones.

Por ejemplo "ayer llovió" es una proposición que depende del lugar y fecha en que es enunciada. Así podría ser la proposición "el 25 de agosto de 1930 llovió en Buenos Aires" o también la proposición "el 3 de junio de 1995 llovió en Bahía Blanca".

Otro ejemplo está dado por "esta mesa es marrón", que supone que el que la enuncia esta indicando algún objeto no explicitado en el lenguaje.

Al referirnos a proposiciones *supondremos que hemos eliminado todas estas ambigüedades.*

Las proposiciones pueden combinarse por medio de las expresiones lógicas "o", "y", "si ..., entonces ...", y también podemos negar una proposición. Una propiedad básica de estas combinaciones es que *la verdad o falsedad de la proposición resultante depende sólo de la verdad o falsedad de las proposiciones de partida.*

Así, por ejemplo, la proposición compuesta "el aire contiene nitrógeno y 'Los Beatles' visitaron la Argentina en 1963" será verdadera si y solamente si es verdadera cada una de las proposiciones "el aire contiene nitrógeno" y "'Los Beatles' visitaron la Argentina en 1963".

El *cálculo proposicional* es un modelo matemático de los razonamientos que se pueden hacer en base a estas formas de combinar proposiciones.

Este modelo matemático nos ayuda a comprender mejor las formas básicas del pensamiento racional, y también nos permite simular esquemas de deducción por medio de computadoras.

El primer paso para establecer un cálculo proposicional es encontrar un sistema de notación que nos permita expresar simbólicamente las combinaciones de proposiciones por medio de los conectivos lógicos "o", "y", "si ..., entonces ...", "no".

Para comprender la importancia de un sistema de notación, basta observar que los cálculos aritméticos (esto es: suma, resta, multiplicación y división de números naturales) se pueden presentar en la forma simple que aprendemos en los primeros grados de la escuela primaria gracias al sistema de numeración decimal, aparentemente inventado en la India entre los siglos 9 y 10 de nuestra era, y difundido por los árabes.

La numeración decimal no es más que un sistema de notación que nos permite escribir (y leer) cualquier número natural usando sólo las cifras $0, 1, 2, \ldots, 9$.

Las operaciones aritméticas se aprenden en la escuela como reglas para manipular estas listas de símbolos.

Por ejemplo, dadas las listas "5643" y "97", aplicando las reglas para sumar obtenemos la lista "5740".

Con el tiempo se comprendió que la característica esencial de este sistema es su carácter posicional, y que el número 10 puede ser sustituido por cualquier número $b > 1$, obteniéndose los sistemas de numeración en base b, que permiten escribir cualquier número con b símbolos básicos.

Por ejemplo, en el sistema binario ($b = 2$), todo número se puede escribir como una lista de ceros y unos.

Esta representación de los números es similar a la que se tiene con los sistemas de escritura alfabéticos de lenguajes naturales, como el castellano: podemos representar las palabras por listas de símbolos, que son las letras del alfabeto.

Para intentar comprender mejor las analogías y diferencias entre los sistemas de numeración y los de escritura de lenguajes naturales, vamos a in-

troducir los siguientes conceptos.

Definición 1.1.1 Dado un conjunto finito y no vacío A, indicaremos con A^* al conjunto de todas las listas finitas de elementos de A. En particular, la lista vacía será indicada por $\square$. Llamaremos *lenguaje sobre el alfabeto* A a cualquier subconjunto $L \subseteq A^*$. Los elementos de un lenguaje L son llamados *palabras*.

Ejemplo 1.1.2 Sea $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Los números naturales se representan por el lenguaje $L = A \setminus \{\square\}$. Como los ceros a la izquierda no son significativos, cada número se representa por infinitas palabras de L: 5, 05, 005, 0005, ... representan el mismo número.

Ejemplo 1.1.3 Sea A como en ejemplo anterior y sea M el conjunto de todas las listas de $A \setminus \{\square\}$ que no comienzan con 0. Entonces $L = M \cup \{0\}$ es un lenguaje sobre A, y cada número puede ser escrito de una manera única.

Ejemplo 1.1.4 Sea A el alfabeto castellano (al que le agregamos, como nuevas letras, las vocales acentuadas). El idioma castellano es un lenguaje sobre A. Podemos definirlo como el conjunto de listas de A^* que figuran en la última edición del diccionario de la Real Academia Española de la Lengua.

Sea A un alfabeto. Para $s \in A^*$, llamaremos *longitud de* s, al número de símbolos de A que figuran en la lista s, contados tantas veces como aparezcan. Así, si A es como en el Ejemplo 1.1.2, la longitud de "000430" es seis.

Los elementos de A serán identificados con los elementos de A^* de longitud uno. La longitud de $s \in A$ será abreviada por $long(s)$.

Para todo número natural n, indicaremos con A^n al subconjunto de A^* formado por los elementos de longitud n.

Observemos que esta notación es coherente con la usual de producto cartesiano, pues una lista de n símbolos del alfabeto puede ser identificada con la n-upla cuya primera componente es el primer símbolo de la lista, la segunda, el segundo símbolo, etc.

En particular, $A^0 = \{\square\}$ y $A = A^1$. Además, $A^* = \bigcup_{n=0}^{\infty} A^n$.

Dados s y t en A^*, indicaremos con st la lista de elementos de A que se obtiene escribiendo la lista t a continuación de la lista s.

Así, si A es como en el Ejemplo 1.1.4 y $s =$ "casa" y $t =$ "caballo", entonces $st =$ "casacaballo". Observemos que si bien s y t pertenecen al castellano, la lista st no pertenece a este lenguaje.

Es claro que para todo $s, t \in A^*$ se tiene que

$$long(st) = long(s) + long(t). \tag{1.1}$$

Una primera diferencia que conviene destacar entre los lenguajes definidos en los Ejemplos 1.1.2, 1.1.3 y 1.1.4 es la siguiente: En los dos primeros, basta inspeccionar una lista para saber si pertenece o no al lenguaje.

En efecto, en el primer caso basta con verificar que la lista s no es vacía y que los únicos símbolos que figuran en s son cifras entre 0 y 9. En el segundo caso, debemos verificar también que si $long(s) > 1$, entonces el primer símbolo de s no es 0.

En cambio, para saber si una lista de símbolos s pertenece o no en el idioma castellano debemos recurrir a un diccionario. Por ejemplo, sin recurrir al Diccionario de la Real Academia no podemos saber si "ñaca" pertenece o no al idioma castellano, como fue definido en el Ejemplo 1.1.4.

En resumen, en los dos primeros ejemplos, los lenguajes están definidos por propiedades *intrínsecas* de las listas, mientras que en el tercero el lenguaje está definido por un *elemento externo*, el diccionario.

La condición de estar definidos por propiedades intrínsecas de las listas de símbolos es una característica de los lenguajes artificiales, como el lenguaje matemático y los lenguajes de programación utilizados en computación, en contraste con los lenguajes naturales, como el castellano, el inglés, etc.

No siempre es posible dar una descripción tan directa de las listas de símbolos que forman un lenguaje como fue hecho en los casos de los Ejemplos 1.1.2 y 1.1.3. El método que seguiremos ahora para definir el lenguaje del cálculo proposicional sirve de modelo para la definición de muchos otros lenguajes.

Para desarrollar el cálculo proposicional necesitaremos símbolos para denotar proposiciones, que jugarán el mismo papel que las expresiones literales en algebra, símbolos para los conectivos lógicos "o", "y", "si ..., entonces ..." y negación, con un rol similar a los símbolos de operaciones algebraicas +,-,..., y, finalmente, los paréntesis, con el mismo empleo que en álgebra.

Teniendo en cuenta esto, definimos como alfabeto del cálculo proposicional al conjunto $\mathbf{A} = \{p, |, \neg, \vee, \wedge, \rightarrow, (,)\}$.

Los símbolos $\neg, \vee, \wedge, \rightarrow$ son llamados *conectivos proposicionales*.

Llamaremos *variables proposicionales* a las listas de $\mathbf{A}^*$ formadas por el símbolo p seguido de un número finito de barras $|$. Para simplificar la escritura, usaremos p_n como abreviatura de p seguido de n barras. Así $p_0 = p$, $p_1 = p|$, $p_2 = p||$, etc.

El subconjunto de $\mathbf{A}^*$ formado por las variables proposicionales será denotado por **Var**.

Vamos a definir ahora el lenguaje **Form** $\subseteq \mathbf{A}^*$. Sus elementos son llamados *fórmulas proposicionales* o, simplemente, *fórmulas*, y constituyen las palabras del lenguaje del cálculo proposicional.

Definición 1.1.5 Una lista de símbolos de **A** es una *fórmula* si y solo si se la puede obtener aplicando un número finito de veces las siguientes reglas:

FP1) Las variables proposicionales son fórmulas.

FP2) Si $P \in \mathbf{A}^*$ es una fórmula, entonces $\neg P$ es una fórmula.

FP3) Si $P, Q \in \mathbf{A}^*$ son fórmulas, entonces $(P \vee Q)$, $(P \wedge Q)$ y $(P \to Q)$ son fórmulas.

¿ Qué significa que una lista de símbolos se obtenga aplicando un número finito de veces las reglas FP1), FP2) y FP3)? La respuesta precisa es la siguiente:

Definición 1.1.6 Una *cadena de formación de longitud n* es una sucesión finita $X_1, \ldots, X_n$ de elementos de $\mathbf{A}^*$ que satisface las condiciones siguientes:

(CF) Para cada i, con $1 \leq i \leq n$, se tiene que o bien X_i es una variable proposicional, o bien existe un j tal que $1 \leq j \leq i-1$ y $X_i = \neg X_j$, o bien existen j y k, ambos entre 1 e $i-1$, tales que $X_i = (X_j \vee X_k)$, $X_i = (X_j \wedge X_k)$ o $X_i = (X_j \to X_k)$.

Cada uno de los X_i, $1 \leq i \leq n$, será llamado un *eslabón* de la cadena de formación.

La Definición 1.1.5 puede ahora traducirse en la siguiente forma:

Una lista $P \in \mathbf{A}^*$ es una fórmula si y sólo si existe una cadena de formación $X_1, \ldots, X_n$ tal que $P = X_n$.

Observación 1.1.7 Observemos que si $X_1, \ldots, X_n$ es una cadena de formación, entonces para cada $i \leq n$, $X_1, \ldots, X_i$ es una cadena de formación de longitud i. Por lo tanto *todos los eslabones de una cadena de formación son fórmulas.*

A título de ejemplo consideremos la expresión:

$$P = (((p_0 \vee p_1) \rightarrow \neg p_3) \wedge p_0).$$

La sucesión

$$X_1 = p_0,$$

$$X_2 = p_1,$$

$$X_3 = p_3,$$

$$X_4 = \neg X_3,$$

$$X_5 = (X_1 \vee X_2),$$

$$X_6 = (X_5 \rightarrow X_4), \text{ y}$$

$$X_7 = (X_6 \wedge X_1)$$

es una cadena de formación de P. Esto significa que P es una fórmula. Notemos todavía que la cadena de formación de una fórmula no es única. Por ejemplo, la sucesión:

$$Y_1 = p_3,$$

$$Y_2 = \neg Y_1,$$

$$Y_3 = p_1,$$

$$Y_4 = p_0,$$

$$Y_5 = (Y_4 \vee Y_3),$$

$$Y_6 = (Y_5 \rightarrow Y_2) \text{ y}$$

$$Y_7 = (Y_6 \wedge Y_4).$$

es otra cadena de formación de la fórmula P.
También es una cadena de formación de P la sucesión:

$$Z_1 = p_0,$$

$$Z_2 = p_1,$$

$$Z_3 = p_2,$$

$$Z_4 = p_3,$$

$$Z_5 = \neg Z_0,$$
$$Z_6 = \neg Z_4,$$
$$Z_7 = (Z_1 \vee Z_2),$$
$$Z_8 = (Z_1 \to Z_3),$$
$$Z_9 = (Z_8 \to Z_6) \text{ y}$$
$$Z_{10} = (Z_9 \wedge Z_1).$$

Esto muestra que una cadena de formación de una fórmula puede tener eslabones superfluos, como Z_3, Z_5 y Z_7.

Teniendo en cuenta la posibilidad de estos eslabones superfluos, vemos que cada fórmula tiene cadenas de formación de longitud arbitrariamente grande.

Conviene insistir en que por el momento *estamos considerando a las fórmulas como meras listas de símbolos*. Por lo tanto cuando decimos que $P = Q$ queremos significar que *en P y en Q aparecen exactamente los mismos símbolos y en el mismo orden.*

Por ejemplo, las fórmulas $(p_0 \vee p_1)$ y $(p_1 \vee p_0)$ no son iguales, pues tienen los mismos símbolos pero en diferente orden.

Observemos también que hay una cantidad infinita de fórmulas. Por ejemplo, con la variable proposicional p_0 y el conectivo $\neg$ ya se pueden generar infinitas fórmulas, de longitud creciente:

$$p_0, \neg p_0, \neg\neg p_0, \neg\neg\neg p_0, \ldots$$

Nuestro próximo objetivo es describir un método inductivo que nos permitirá asegurar que ciertas propiedades las tienen todas las fórmulas. Comenzaremos por la siguiente

Definición 1.1.8 Un subconjunto $S \subseteq \mathbf{A}^*$ se dice *cerrado por los conectivos* si satisface las siguientes condiciones:

(C1) Si la lista $X \in S$, entonces también la lista $\neg X \in S$,

(C2) Si las listas X e Y pertenecen a S, entonces también las listas $(X \vee Y)$, $(X \wedge Y)$ y $(X \to Y)$ pertenecen a S.

La importancia de los conjuntos cerrados por los conectivos estriba en el siguiente teorema:

Teorema 1.1.9 *Sea S un subconjunto de $\mathbf{A}^*$ cerrado por los conectivos. Si S contiene a todas las variables proposicionales, entonces S contiene a todas las fórmulas.*

Demostración: Sea S un subconjunto de $\mathbf{A}^*$ cerrado por los conectivos y que contiene a todas las variables proposicionales. Queremos ver que para toda fórmula P, se tiene que $P \in S$. Para ello razonaremos por inducción en la longitud de las cadenas de formación de P. Supongamos primero que P tiene una cadena de formación de longitud 1. Esto sólo es posible si P es una variable proposicional, esto es $P = p_k$ para algún número natural k, y en este caso $P \in S$ por hipótesis.

Sea $n > 1$. Como hipótesis inductiva supongamos que: *para toda fórmula Q que admita una cadena de formación de longitud $< n$, se tiene que $Q \in S$.*

Supongamos que P admite una cadena de formación de longitud n, digamos $X_1, \ldots, X_n$, con $X_n = P$. Si X_n es una variable proposicional, entonces ya sabemos que $P \in S$. Si no lo es, se pueden presentar los siguientes casos:

(1) Existe i tal que $1 \leq i \leq n$ y $P = \neg X_i$ ó

(2) Existen i, j entre 1 y n tales que $P = (X_i \vee X_j)$, $P = (X_i \wedge X_j)$ ó $P = (X_i \rightarrow X_j)$.

En el caso (1), $X_1, \ldots, X_i$ es una cadena de formación de la fórmula X_i de longitud $i < n$.

Luego, por la hipótesis inductiva, $X_i \in S$, y por la propiedad (C1), $P \in S$.

En el caso (2), $X_1, \ldots, X_i$ y $X_1, \ldots, X_j$ son cadenas de formación de las fórmulas X_i y X_j respectivamente, ambas de longitud $< n$. Por la hipótesis inductiva $X_i \in S$ y $X_j \in S$, y por la propiedad (C2) resulta que $P \in S$.

Por el principio de inducción resulta entonces que para todo número natural n, si la fórmula P tiene una cadena de formación de longitud n, entonces $P \in S$.

Como toda fórmula tiene una cadena de formación de longitud finita, resulta que $\mathbf{Form} \subseteq S$, c.q.d.

El teorema anterior nos da el método que mencionamos antes de la Definición 1.1.8 para probar que las fórmulas tienen una determinada propiedad: basta verificar que el subconjunto de $\mathbf{A}^*$ formado por las listas de símbolos que satisfacen esa propiedad es cerrado por los conectivos y contiene a todas las variables proposicionales.

A continuación veremos una primera aplicación de este método, que nos permitirá concluir que cada fórmula puede ser leída de una única manera.

Definición 1.1.10 Sea $X \in \mathbf{A}^*$. Llamaremos *peso de* X, y lo denotaremos por $peso(X)$, al número entero que se obtiene restando el número de paréntesis izquierdos "(" del de paréntesis derechos ")" que figuran en la lista X.

Observemos que si en la lista X no figuran ni paréntesis izquierdos ni paréntesis derechos, entonces $peso(X) = 0$. En particular, tenemos que:

(1.2) Para toda variable proposicional p_n, $peso(p_n) = 0$

También resulta inmediatamente de la definición anterior, que cualesquiera que sean X e Y pertenecientes a $\mathbf{A}^*$:

$$peso(\neg X) = peso(X) \tag{1.3}$$

y si $\star$ denota uno cualquiera de los conectivos $\vee$, $\wedge$ ó $\rightarrow$, entonces

$$peso((X \star Y)) = peso(X) + peso(Y) \tag{1.4}$$

Lema 1.1.11 *Para toda fórmula P valen las siguientes propiedades:*

(i) *El número de paréntesis "(" que figuran en P es igual al número de paréntesis ")". Esto es, $peso(P) = 0$.*

(ii) *A la izquierda de cualquiera de los conectivos $\vee$,$\wedge$ ó $\rightarrow$ que figuren en P, hay un número estrictamente mayor de paréntesis "(" que de paréntesis ")". Esto es, si $P = X \star Y$, donde X e Y pertenecen a $\mathbf{A}^*$ y $\star$ representa uno de los conectivos $\vee$,$\wedge$ ó $\rightarrow$, entonces $peso(X) > 0$.*

Demostración: Llamemos S al conjunto de todas las listas de $\mathbf{A}^*$ de peso 0:

$$S = \{X \in \mathbf{A}^* \mid peso(X) = 0\}$$

Del enunciado (1.2) resulta que todas las variables proposicionales pertenecen a S, y de (1.3) y (1.4) se deduce que S es cerrado por los conectivos. Luego por el Teorema 1.1.9, debe ser $\mathbf{Form} \subseteq S$, esto es, todas las fórmulas tienen peso cero, lo que prueba el enunciado (i).

Pasemos ahora a probar el enunciado (ii). Llamemos T al conjunto de todas las fórmulas tales que a la izquierda de cualquiera de los símbolos $\vee$,$\wedge$ ó $\rightarrow$ que figuren en X hay un número mayor de paréntesis "(" que de parentesis ")".

Esto es, una fórmula P pertenece al conjunto T si y sólo si $P = X \star Y$, con X, $Y \in \mathbf{A}^*$, implica que $peso(X) > 0$.

Como en las variables proposicionales sólo figuran los símbolos p y $|$, éstas pertenecen trivialmente al conjunto T.

Para probar que T es cerrado por los conectivos, supongamos que $P \in T$ y que $\neg P = X \star Y$. Esta última condición implica que existe $Z \in \mathbf{A}^*$ tal que $X = \neg Z$, y por lo tanto que $P = Z \star Y$. Como $P \in T$, debe ser $peso(Z) > 0$, y por (1.3) se tiene que también $peso(X) > 0$. Luego $\neg P \in S$. Por lo tanto T satisface la condición (C1) de la Definición 1.1.8.

Para probar que también satisface (C2), supongamos que P y Q pertenezcan a T y sea $\star$ uno de los símbolos $\vee$,$\wedge$ ó $\rightarrow$ que figuran en la lista $(P \vee Q)$. Si $\star$ figura a la izquierda de $\vee$, entonces deben existir X, Y en $\mathbf{A}^*$ tales que $P = X \star Y$, y $(P \vee Q) = (X \star Y \vee Q)$. Como $P \in T$, debe ser $peso(X) > 0$, y como $peso((X) = 1 + peso(X)$, resulta que la expresión a la izquierda de $\star$ en $(P \vee Q)$ tiene peso mayor que cero.

Si $\star$ coincide con $\vee$, entonces a la izquierda de $\star$ figura (P, y como por la parte (i), $peso(P) = 0$, resulta que $peso((P) = 1$.

Finalmente, supongamos que $\star$ figure a la derecha de $\vee$. Entonces existen X, Y en $\mathbf{A}^*$ tales que $Q = X \star Y$, y $(P \vee Q) = (P \vee X \star Y)$. Como $Q \in T$, debe ser $peso(X) > 0$, y se tiene que $peso((P \vee X) = 1 + peso(X) > 2$.

Acabamos de ver así que la expresión a la izquierda de cualquiera de los símbolos $\vee$,$\wedge$ ó $\rightarrow$ que figuran en la lista $(P \vee Q)$, tiene siempre peso mayor que cero, lo que significa que $(P \vee Q) \in T$.

En forma enteramente análoga se prueba que si P y Q pertenecen a T, entonces también $(P \wedge Q)$ y $(P \rightarrow Q)$ pertenecen a al conjunto T. Luego T es cerrado por los conectivos.

Como T es un subconjunto de $\mathbf{A}^*$ que contiene a las variables proposicionales y es cerrado por los conectivos, debe ser $\mathbf{Form} \subseteq T$, es decir todas las fórmulas satisfacen las condiciones que definen al conjunto T, c.q.d.

Notemos que como el conjunto T en la demostración anterior fue definido como un subconjunto de **Form**, en realidad hemos probado que $\mathbf{Form} = T$.

Teorema 1.1.12 (Unicidad de la lectura de una fórmula.) *Sean P, Q, R y S fórmulas, y supongamos que $\star$ y $\circ$ representan algunos de los conec-*

tivos $\vee$, $\wedge$ *ó* $\rightarrow$. *Si* $(P \star Q) = (R \circ S)$, *entonces debe ser* $P = R$, $Q = S$ *y* $\star = \circ$.

Demostración: Supongamos, por el absurdo, que $(P \star Q) = (R \circ S)$ y que $P \neq R$. En estas condiciones o bien existe $X \in \mathbf{A}^*$ tal que $(R \circ S) = (P \star X \circ S)$ o bien existe $Y \in \mathbf{A}^*$ tal que $(P \star Q) = (R \circ Y \star Q)$.

En el primer caso tendríamos que $R = P \star X$, y en el segundo, que $P = R \circ Y$. Pero por (ii) del lema anterior esto implicaría que $peso(P) > 0$ o $peso(R) > 0$, y ambas desigualdades son imposibles por la parte (i) del mismo lema.

Luego hemos probado que la igualdad $(P \star Q) = (R \circ S)$ implica que $P = R$.

Pero de las igualdades $(P \star Q) = (R \circ S)$ y $P = R$ resulta inmediatamente que debe ser $\star = \circ$ y $Q = S$, lo que concluye la demostración del teorema.

Definición 1.1.13 Llamaremos *grado de complejidad de una fórmula* P, y lo denotaremos por $comp(P)$, al número de conectivos que figuran en P, contados tantas veces como aparezcan.

Por ejemplo, $comp(\neg\neg\neg p_0) = 3$, $comp((p_5 \vee p_{17})) = 1$.

Corolario 1.1.14 *Para toda fórmula* P *se tiene que:*

(i) $comp(P) = 0$ *si y sólo si* P *es una variable proposicional.*

(ii) *Si* $comp(P) > 0$, *entonces se cumple uno, y sólo uno, de los casos siguientes:*

- **(1)** *Existe una única fórmula* Q *tal que* $P = \neg Q$.
- **(2)** *Existe un único par de fórmulas* Q, R *tal que* $P = (Q \vee R)$.
- **(3)** *Existe un único par de fórmulas* Q, R *tal que* $P = (Q \wedge R)$.
- **(4)** *Existe un único par de fórmulas* Q, R *tal que* $P = (Q \rightarrow R)$.

Demostración: La parte (i) del enunciado es obvia. La parte (ii) es una consecuencia inmediata de la definición de fórmula y del Teorema 1.1.12.

Como una aplicación de lo anterior, definiremos inductivamente la noción de *subfórmula* de una fórmula.

Sea P una fórmula proposicional. Si $comp(P) = 0$ (esto es, si P es una variable proposicional), entonces P es la única subfórmula de P. Supongamos

ahora que $comp(P) = n > 0$ y que hemos definido subfórmulas para toda fórmula de complejidad menor que n. Por el Corolario 1.1.14 sabemos que o bien existe una única fórmula Q tal que $P = \neg Q$ o bien existen un único par de fórmulas Q, R y un único conectivo $* \in \{\vee, \wedge, \rightarrow\}$ tales que $P = (Q * R)$. En el primer caso, $comp(Q) = n-1$, luego por la hipótesis inductiva tenemos definidas las subfórmulas de Q. Definimos las subfórmulas de P como P y todas las subfórmulas de Q.

En el otro caso, se tiene que $comp(Q) < n$ y $comp(R) < n$. Luego por la hipótesis inductiva están definidas las subfórmulas de Q y de R. Definimos las subfórmulas de P como P y todas las subfórmulas de Q y de R.

Esto completa la definición de subfórmula para cualquier fórmula proposicional P.

Ejemplo 1.1.15 Si $P = ((p_0 \vee \neg p_1) \wedge (p_3 \rightarrow p_2))$, utilizando la definición inductiva obtenemos que las subfórmulas de P son P, $(p_0 \vee \neg p_1)$, $(p_3 \rightarrow p_1)$, p_0, $\neg p_1$, p_3, p_2 y p_1.

Con la noción de complejidad hemos asociado un número natural con cada fórmula proposicional, y esto nos permitió hacer razonamientos inductivos sobre las fórmulas. Una propiedad importante que se utiliza en estos razonamientos inductivos es que las fórmulas de complejidad 0 son exactamente las variables proposicionales.

La complejidad no es el único número que se puede asociar con una fórmula. Por ejemplo, también tenemos la longitud. Pero la longitud no es adecuada para razonamientos inductivos, pues *las variables proposicionales pueden tener longitud arbitrariamente grande*. En efecto, recordemos que p_n abrevia al símbolo p seguido de n barras $|$, por lo que $long(p_n) = n + 1$.

Para evitar este inconveniente, podemos modificar la definición de longitud para fórmulas, asignándole a las variables proposicionales longitud 1.

Estas consideraciones nos conducen a definir la *longitud modificada* de una fórmula P, que denotamos $long^*(P)$ como el grado de complejidad de P más el número de variables que figuran en P, contadas tantas veces como aparezcan repetidas.

En otras palabras, la longitud modificada es la función

$$long^*\colon \mathbf{Form} \rightarrow \mathbf{N} \setminus \{0\}$$

caracterizada por las siguientes propiedades:

(GCE1) Para toda variable proposicional p_n, $long^*(p_n) = 1$,

(GCE2) Para toda fórmula P, $long^*(\neg P) = 1 + long^*(P)$,

(GCE3) Para todo par de fórmulas P, Q, $long^*(P \vee Q) = long^*(P \wedge Q) = long^*(P \to Q) = 1 + long^*(P) + long^*(Q)$.

Dejamos al cuidado del lector verificar que tanto en la definición de subfórmula como en demostraciones que haremos en lo sucesivo, se puede reemplazar la inducción en $comp(P)$ por inducción en $long^*(P)$.

1.2 Semántica del cálculo proposicional.

Hasta el momento hemos considerado las fórmulas como meras listas de símbolos, construidas a partir de un alfabeto mediante ciertas reglas bien determinadas. Este punto de vista es llamado *sintáctico*.

Ahora vamos a interpretar las fórmulas como proposiciones, esto es, como enunciados a los que se le pueden asignar uno de los dos valores de verdad "verdadero" o "falso". Este nuevo punto de vista es llamado *semántico*.

La siguiente analogía puede ayudar a comprender la diferencia entre los dos puntos de vista: en la escuela primaria, aprendemos las reglas para efectuar las operaciones aritméticas en forma mecánica, considerando los números escritos en el sistema decimal. Esto corresponde al punto de vista sintáctico. Para resolver problemas, los números son interpretados como valores de mercaderías, longitudes, volúmenes, etc. Es decir, a las listas de cifras de la notación decimal les atribuímos un significado concreto, lo mismo que a los símbolos $+, -, \times, \div$, lo que nos permite aplicar las reglas aritméticas para obtener la solución del problema. Este es el punto de vista semántico.

Siguiendo la tradición, vamos a representar "verdadero" por el 1, y "falso" por 0.

Comenzaremos por interpretar los conectivos: $\vee$ será interpretado como "o", $\wedge$ como "y", $\to$ como "si ..., entonces ...", y $\neg$ como "negación".

De acuerdo con esta interpretación, el comportamiento de los conectivos binarios con respecto a los valores de verdad se puede resumir en las siguientes tablas:
y el comportamiento del conectivo unario de negación está dado por:

$$\neg 0 = 1 \text{ y } \neg 1 = 0. \tag{1.5}$$

$\vee$	0	1
0	0	1
1	1	1

$\wedge$	0	1
0	0	0
1	0	1

$\rightarrow$	0	1
0	1	1
1	0	1

Por ejemplo, de la primera tabla resulta que:

falso o falso es falso y *falso o verdadero es verdadero.*

y de la tercera tabla que:

falso implica verdadero es verdadero.

Este comportamiento de los conectivos proposicionales con respecto a los valores de verdad es el habitual en matemática, y se remonta por lo menos al siglo IV A.C., cuando fue especificada por los filósofos griegos de la escuela estoica, en especial Crisipo.

Vamos a indicar con $\mathbf{B}$ al conjunto $\{0, 1\}$. Si pensamos a 0 y a 1 como números, con el orden y las operaciones ordinarias, las tablas anteriores pueden interpretarse del siguiente modo, donde x e y representan elementos de $\mathbf{B}$:

$$\neg x = 1 - x, \tag{1.6}$$

$$x \vee y = \max(x, y), \tag{1.7}$$

$$x \wedge y = \min(x, y) = xy, \tag{1.8}$$

$$x \rightarrow y = \max(1 - x, y). \tag{1.9}$$

Además se tiene que

$$x \vee \neg x = 1 \ \text{ y } \ x \wedge \neg x = 0. \tag{1.10}$$

El lector familiarizado con las álgebras de Boole, reconocerá que $\mathbf{B}$ con las operaciones indicadas *es el álgebra de Boole con dos elementos.*

Interpretaremos ahora las fórmulas a través del concepto de valuación, que pasamos a definir.

Definición 1.2.1 Una *valuación booleana*, o, simplemente, una *valuación*, es cualquier función $v\colon \mathbf{Form} \rightarrow \mathbf{B}$ que satisfaga las siguientes condiciones, donde P y Q denotan fórmulas arbitrarias:

(V1) $v(\neg P) = 1 - v(P)$,

(V2) $v(P \vee Q) = \max(v(P), v(Q))$,

(V3) $v(P \wedge Q) = \min(v(P), v(Q))$, y

(V4) $v(P \to Q) = \max(1 - v(P), v(Q))$.

Nótese que las valuaciones le asignan a cada fórmula un valor de verdad, pero no de cualquier modo, sino respetando el significado de los conectivos proposicionales. De aquí resulta – y es un buen ejercicio detenerse un instante a pensarlo – que fijada una valuación las listas de símbolos que llamamos fórmulas pueden interpretarse como un conjunto de enunciados, algunos verdaderos, algunos falsos, sujetos a las operaciones lógicas usuales.

Una pregunta importante, directamente ligada con la expresividad del lenguaje formal que hemos definido, es la de si habrá "pocas" o "muchas" valuaciones.

Si nuestro lenguaje pretende representar combinaciones de conjuntos arbitrarios de proposiciones, sería conveniente que hubiera "muchas" valuaciones: debería haber, por lo menos, una para cada asignación posible de valores de verdad a las variables proposicionales.

Afortunadamente, esto es lo que ocurre, como lo muestra el siguiente teorema.

Teorema 1.2.2 *Sea f una función cualquiera del conjunto* **Var** *de variables proposicionales en* **B**. *Entonces existe una única valuación v_f: $\mathbf{Form} \to \mathbf{B}$ que extiende a f, es decir, tal que $v_f(p_n) = f(p_n)$ para toda variable proposicional p_n.*

Demostración: Vamos a definir la valuación v_f por inducción en el grado de complejidad de las fórmulas (ver la Definición 1.1.13).

Si P es una fórmula tal que $comp(P) = 0$, entonces por (i) del Corolario 1.1.14 existe un n tal que $P = p_n$. Definimos $v_f(P) = v_f(p_n) = f(p_n)$.

Sea ahora $n \geq 1$. Como hipótesis inductiva, supongamos que hemos definido v_f para toda fórmula de grado de complejidad estrictamente menor que n.

Si P es una fórmula tal que $comp(P) = n$, entonces por (ii) del Corolario 1.1.14 se puede presentar uno, y sólo uno de los casos siguientes:

(1) Existe una única fórmula Q tal que $P = \neg Q$.

(2) Existe un único par de fórmulas Q, R tal que $P = (Q \vee R)$.

(3) Existe un único par de fórmulas Q, R tal que $P = (Q \wedge R)$.

(4) Existe un único par de fórmulas Q, R tal que $P = (Q \to R)$.

En el caso (1), $comp(Q) = n - 1 < n$, y por la hipótesis inductiva, está definida $v_f(Q)$. Definimos $v_f(P) = 1 - v_f(Q)$.

En los casos (2), (3) y (4) se tiene que $comp(Q) < comp(P)$ y $comp(R) < comp(P)$, luego por la hipótesis inductiva están unívocamente definidos los valores $v_f(Q)$ y $v_f(R)$. En el caso (2), definimos $v_f(P) = \max(v_f(Q), v_f(R))$, en el caso (3), $v_f(P) = \min(v_f(Q), v_f(R))$ y en el caso (4), $v_f(P) = \max(1 - v_f(Q), v_f(R))$.

De esta forma definimos unívocamente el valor $v_f(P)$ para toda fórmula de grado de complejidad n.

Obtuvimos así una función $v_f\colon \mathbf{Form} \to \mathbf{B}$, y de la definición resulta obviamente que esta función v_f es una valuación y que $v_f(p_n) = f(p_n)$ para toda variable proposicional p_n.

Para terminar la demostración sólo nos queda probar que la v_f que acabamos de definir es la única valuación que extiende a la función f.

Supongamos que hubiese otra, digamos w, y consideremos el conjunto $I = \{P \in \mathbf{Form} \mid v_f(P) = w(P)\}$.

Como w también extiende a f, I contiene a las variables proposicionales, y como v_f y w son ambas valuaciones, es fácil ver que I es cerrado por los conectivos. Luego por el Teorema 1.1.9 resulta que $\mathbf{Form} \subseteq I$, es decir, $v_f(P) = w(P)$ para toda fórmula P, c.q.d.

Observación 1.2.3 La construcción anterior nos permite obtener todas las posibles valuaciones. En efecto, si $v\colon \mathbf{Form} \to \mathbf{B}$ es una valuación cualquiera y llamamos f a la restricción de v a **Var** (es decir, $f\colon \mathbf{Var} \to \mathbf{B}$ está definida por $f(p_n) = v(p_n)$ para todo n), resulta de la unicidad que $v = v_f$.

El resultado siguiente nos será útil más adelante.

Corolario 1.2.4 *Fijemos n variables proposicionales distintas, que denotaremos por $x_1, \ldots, x_n$. Para cada n-upla $\vec{a} = (a_1, \ldots, a_n) \in \mathbf{B}^n$, existe una valuación $v_{\vec{a}}$ tal que $v_{\vec{a}}(x_1) = a_1, v_{\vec{a}}(x_2) = a_2, \ldots, v_{\vec{a}}(x_n) = a_n$.*

Demostración: Consideremos una función $f\colon \mathbf{Var} \to \mathbf{B}$ tal que $f(x_i) = a_i$ para $i = 1, \ldots, n$. Por ejemplo, podemos definir f del modo siguiente:

$$f(p_k) = \begin{cases} a_i & \text{si} \quad p_k = x_i \\ 0 & \text{si} \quad p_k \notin \{x_1, \ldots, x_n\} \end{cases}$$

La valuación v_f del Teorema 1.2.2 satisface que $v_f(x_i) = f(x_i) = a_i$, $i = 1, \ldots, n$, luego podemos tomar $v_{\vec{a}} = v_f$, c.q.d.

Convención: En lo sucesivo, y para simplificar la escritura, en general omitiremos los paréntesis exteriores al escribir fórmulas. Además, al escribir una disyunción o una conjunción de varias fórmulas, eliminaremos los paréntesis, suponiendo que estamos asociando a la izquierda. En otras palabras, las fórmulas

$$(\ldots(P_1 \vee P_2) \vee \ldots) \vee P_r)$$

y

$$(\ldots(P_1 \wedge P_2) \wedge \ldots) \wedge P_r),$$

serán abreviadas por

$$P_1 \vee P_2 \vee \ldots \vee P_r$$

y

$$P_1 \wedge P_2 \wedge \ldots \wedge P_r,$$

respectivamente.

Por ejemplo, $((P_1 \vee P_2) \vee P_3)$ será abreviada por $P_1 \vee P_2 \vee P_3$.

La noción de valuación nos permite diferenciar a las fórmulas en las siguientes tres clases:

Definición 1.2.5 Diremos que una fórmula P del cálculo proposicional es una *tautología* si $v(P) = 1$ para toda valuación v. Diremos que P es una *falsedad* (o una *contradicción*) si $v(P) = 0$ para toda valuación v. Y diremos que P es *una contingencia* si P no es ni una tautología ni una contradicción.

Las tautologías son aquellas fórmulas que son verdaderas por la configuración de sus símbolos, independientemente de la valuación que se considere; se corresponden con las "verdades lógicas".

Por ejemplo: $p_0 \vee \neg p_0$.

Otro ejemplo, no tan obvio, es el siguiente. Sea P una fórmula de la forma

$$((x \wedge y) \to z) \to ((x \to z) \vee (y \to z)),$$

donde x, y, z denotan variables proposicionales cualesquiera. Si P no fuese una tautología debería existir una valuación v tal que $v(P) = 0$. Como P es de la forma $Q \to R$, con $Q = (x \wedge y) \to z$ y $R = (x \to z) \vee (y \to z)$, de acuerdo con la condición (V4) de la definición de valuación, debería ser

(a) $v((x \wedge y) \to z) = 1$ y

(b) $v((x \to z) \vee (y \to z)) = 0$.

De la igualdad (b) obtenemos, por la propiedad (V2), que $v(x \to z) = 0$ y $v(y \to z) = 0$ y de aquí, usando la propiedad (V4), concluímos que $v(x) = v(y) = 1$, y que $v(z) = 0$. Pero entonces $v((x \wedge y) \to z) = \max(1-(1\wedge 1), 0) = 0$, lo que contradice (a).

Del mismo modo, las falsedades son las fórmulas que son falsas por su configuración; por ejemplo: $p_0 \wedge \neg p_0$.

Es fácil ver que *una fórmula P es una falsedad si y sólo si su negación $\neg P$ es una tautología*.

En contraposición con estos dos casos, el valor de verdad de las contingencias depende de la valuación que se considere. Por ejemplo, si

$$P = (x \to \neg y) \to (y \wedge x),$$

para una valuación v que considere verdaderas a x y a y, P resultará verdadera, y para una valuación v que considere falsa a x y verdadera a y, P resultará falsa.

Las contingencias se corresponden con los enunciados que realmente interesan en un contexto dado, esto es, aquellos cuyo valor de verdad no está determinado trivialmente.

En los ejemplos anteriores, para calcular $v(P)$ usamos las propiedades (V1)–(V4) para poder aplicar la valuación v directamente a las variables proposicionales que figuran en P, obteniendo así elementos $\mathbf{B}$, y considerar los símbolos de los conectivos como representando las operaciones definidas por las ecuaciones (1.6), (2.8), (2.7) y (1.9). Formalizaremos este procedimiento en el próximo teorema.

Antes conviene introducir la siguiente notación: Para toda fórmula P, $var(P)$ denotará el conjunto de las variables proposicionales que figuran en P.

Teorema 1.2.6 *Supongamos que $x_1, \ldots, x_n$ representan variables proposicionales distintas y sea P una fórmula tal que $var(P) \subseteq \{x_1, \ldots, x_n\}$. Para toda valuación v, el siguiente procedimiento nos permite calcular el valor $v(P)$, conociendo los valores $v(x_1), \ldots, v(x_n)$:*

- *Sustituir en la expresión de P cada una de las variables x_i por $v(x_i)$, y evaluar la expresión así obtenida en $\mathbf{B}$, utilizando las tablas que interpretan los conectivos binarios y las relaciones (1.5).*

El resultado obtenido es $v(P)$.

Demostración: Por inducción en el grado de complejidad de P. Si $comp(P) = 0$, entonces P debe ser una variable proposicional, esto es $P = x_i$ para algún i entre 1 y n, y $v(P) = v(x_i) \in \mathbf{B}$.

Sea $comp(P) = n > 0$. Como hipótesis inductiva, supondremos que el procedimiento indicado nos da el valor $v(S)$ para toda fórmula S tal que $var(S) \subseteq \{x_1, \ldots, x_n\}$ y $comp(S) < n$. Por el Corolario 1.1.14 sabemos que se puede presentar uno (y sólo uno) de los casos siguientes:

(1) Existe una única fórmula Q tal que $P = \neg Q$.

(2) Existe un único par de fórmulas Q, R tal que $P = (Q \vee R)$.

(3) Existe un único par de fórmulas Q, R tal que $P = (Q \wedge R)$.

(4) Existe un único par de fórmulas Q, R tal que $P = (Q \to R)$.

Observemos que en todos los casos se tiene que $comp(Q) < comp(P)$ y $comp(R) < comp(P)$. Además, en el caso (1), $var(Q) = var(P)$, y en los casos (2), (3) y (4), $var(Q) \cup var(R) = var(P)$. Luego en todos los casos se tiene que $var(Q) \subseteq \{x_1, \ldots, x_n\}$ y $var(R) \subseteq \{x_1, \ldots, x_n\}$.

Sustituir cada una de las variables x_i que figuren en P por el valor $v(x_i)$ significa hacer estas sustituciones en Q y en R. Luego por la hipótesis inductiva, al efectuar estas sustituciones obtenemos los valores $v(Q)$ y $v(R)$.

Para completar la demostración, basta observar que de las propiedades (V1)–(V4) de las valuaciones resulta que $v(P)$ se obtiene de los valores $v(Q)$ y $v(R)$ considerando los conectivos como las respectivas operaciones en $\mathbf{B}$, c.q.d.

El siguiente resultado es una consecuencia inmediata del teorema anterior:

Corolario 1.2.7 *Sea P una fórmula tal que $var(P) \subseteq \{x_1, \ldots, x_n\}$. Si v y w son dos valuaciones tales que $v(x_i) = w(x_i)$ para $1 \leq i \leq n$, entonces $v(P) = w(P)$.*

Resulta de este corolario que *dos valuaciones pueden asignar diferentes valores de verdad a una fórmula P solamente si asignan diferentes valores de verdad a alguna de las variables proposicionales que intervienen en la escritura de P.*

Un tópico importante del cálculo proposicional es el de encontrar métodos que permitan determinar, de una manera sistemática y en una cantidad finita de pasos (de modo que puedan implementarse en una computadora) si una fórmula dada es o no es una tautología. La importancia de este problema va mucho mas allá de la Lógica, como se verá más adelante.

En principio, el problema de determinar si una fórmula dada es o no es una tautología involucra a todas las valuaciones, y éstas forman un conjunto infinito.

Veremos, sin embargo, que se trata de un problema esencialmente finito: para cada fórmula P, basta estudiar sólo una cantidad finita, bien determinada, de posibilidades.

Sea P una fórmula y supongamos que sus variables proposicionales son $x_1, \ldots, x_n$, esto es, $var(P) = \{x_1, \ldots, x_n\}$. Por el Corolario 1.2.4, para cada n-upla $\vec{a} = (a_1, \ldots, a_n) \in \mathbf{B}^n$ hay una valuación $v_{\vec{a}}$, tal que $v_{\vec{a}}(x_i) = a_i$, para $i = 1, \ldots, n$.

De acuerdo con el Teorema 1.2.6, $v_{\vec{a}}(P)$ se calcula sustituyendo en P las variables x_i por los valores a_i y efectuando las operaciones correspondientes a los conectivos.

Si listamos ahora todas las n-uplas $\vec{a} \in \mathbf{B}^n$, y a la derecha de cada una de ellas escribimos el valor $v_{\vec{a}}(P)$, obtenemos una tabla (o matriz) de 2^n filas (o renglones) y $n+1$ columnas, formada por elementos de $\mathbf{B}$.

Por ejemplo, la Tabla 1.1 corresponde a $P = (p_0 \vee p_7) \to p_9$, con $x_1 = p_0$, $x_2 = p_7$ y $x_3 = p_9$.

La tabla que acabamos de describir se llama *la tabla de verdad de la fórmula P*, y nos permite conocer el valor $v(P)$ para cualquier valuación v.

En efecto, sea v una valuación cualquiera y consideremos la n-upla $\vec{a} = (v(x_1), \ldots, v(x_n))$. Supongamos que esta n-upla corresponde a los n primeros lugares de la fila i-ésima. Por el Corolario 1.2.7, $v(P) = v_{\vec{a}}(P)$, y este valor es el que figura en la última columna de dicha fila, esto es, en el lugar $(i, n+1)$.

0	0	0	1
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

Table 1.1: Tabla de verdad de $(p_0 \vee p_7) \to p_9$

Tenemos entonces un primer método para determinar si una fórmula P dada es o no una tautología: escribimos su tabla de verdad y miramos la última columna. P es una tautología si y sólo si en esta columna figura únicamente el símbolo 1. Si figura únicamente 0, P es una falsedad y si figuran al menos un 0 y un 1, P es una contingencia.

Recordemos que los n primeros elementos de las filas de la tabla de verdad de una fórmula P tal que $var(P) = \{x_1, \ldots, x_n\}$ están en correspondencia con las funciones de $\{x_1, \ldots, x_n\}$ en $\mathbf{B}$. Por lo tanto dos filas distintas no pueden tener sus primeros n elementos iguales: si $a_1, \ldots, a_n, a_{n+1}$ y $b_1, \ldots, b_n, b_{n+1}$ son dos filas distintas de la tabla de verdad de P, debe existir al menos un $i \leq n$ tal que $a_i \neq b_i$.

De esta observación resulta que la tabla de verdad de una fórmula P en la que figuran n variables proposicionales distintas puede considerarse como una función $T_P\colon \mathbf{B}^n \to \mathbf{B}$.

En efecto, para toda n-upla $(a_1, \ldots, a_n) \in \mathbf{B}^n$, en la tabla de verdad de P hay una (única) fila cuyos primeros n elementos coinciden con los de esta n-upla, y si a_{n+1} es el elemento que figura en la columna $n+1$ de dicha fila, podemos definir sin ambigüedad $T_P(a_1, \ldots, a_n) = a_{n+1}$.

Por ejemplo, si $P = (p_0 \vee p_7) \to p_9$, de la tabla anterior resulta que $T_P(0,1,0) = 0$, $T_P(1,0,0) = 0$.

En resumen, a cada fórmula P del cálculo proposicional tal que $var(P) = \{x_1, \ldots, x_n\}$, le podemos hacer corresponder una función T_P de $\mathbf{B}^n$ en $\mathbf{B}$. Esta función está caracterizada por la siguiente propiedad:

(TV) $T_P(v(x_1), \ldots, v(x_n)) = v(P)$.

Las funciones de $\mathbf{B}^n$ en $\mathbf{B}$ se llaman *funciones booleanas de n variables.*

Los resultados anteriores muestran que podemos asociar con cada fórmula proposicional una función booleana. Una pregunta que se plantea naturalmente es si vale la recíproca, esto es, si dada una función booleana $f\colon \mathbf{B}^n \to \mathbf{B}$, existe una fórmula P tal que $T_P = f$.

La respuesta es afirmativa, como lo muestra el siguiente:

Teorema 1.2.8 *Sea n un número natural > 0 y f una función booleana de n variables. Entonces dadas n variables proposicionales distintas $x_1, \ldots, x_n$, existe una fórmula P tal que $var(P) = \{x_1, \ldots, x_n\}$ y $T_P = f$.*

Demostración: En esta demostración usaremos repetidas veces la convención de la página 19.

Sea $f\colon \mathbf{B}^n \to \mathbf{B}$ una función. Si f toma idénticamente el valor 0, est es, si $f(a_1, \ldots, a_n) = 0$ para toda n-upla $(a_1, \ldots, a_n) \in \mathbf{B}^n$, entonces escribiendo

$$P = (x_1 \wedge \neg x_1) \vee (x_2 \wedge \neg x_2) \vee \cdots \vee (x_n \wedge \neg x_n),$$

tenemos que $var(P) = \{x_1, \ldots, x_n\}$ y $T_P = f$.

Supongamos ahora que f no es idénticamente 0, y representemos a f por una tabla de 2^n filas por $n+1$ columnas, de modo que si el elemento de $\mathbf{B}$ que figura en la intersección de la i-ésima fila con la j-ésima columna se denota por t_{ij}, se tiene que $t_{in+1} = f(t_{i1}, \ldots, t_{in})$, para $i = 1, \ldots, 2^n$.

Para cada i entre 1 y 2^n y para cada j entre 1 y n, definamos la fórmula P_{ij} del modo siguiente:

$$P_{ij} = \begin{cases} x_j & \text{si } t_{ij} = 1, \\ \neg x_j & \text{si } t_{ij} = 0. \end{cases}$$

y para cada i entre 1 y 2^n, sea

$$P_i = P_{i1} \wedge P_{i2} \wedge \cdots \wedge P_{in}.$$

Es claro que $var(P_i) = \{x_1, \ldots, x_n\}$. Además se tiene que

$$T_{P_i}(t_{i1}, \ldots, t_{in}) = \min(T_{P_{i1}}(t_{i1}, \ldots, t_{in}), \ldots, T_{P_{in}}(t_{i1}, \ldots, t_{in}))$$

y como

$$T_{P_{ij}}(t_{i1}, \ldots, t_{in}) = \begin{cases} t_{ij} & \text{si } t_{ij} = 1, \\ 1 - t_{ij} & \text{si } t_{ij} = 0 \end{cases}$$

se tiene que

$$T_{P_{ij}}(t_{i1}, \ldots, t_{in}) = 1 \text{ para } 1 \leq j \leq n.$$

Por lo tanto vemos que

$$T_{P_i}(t_{i1}, \ldots, t_{in}) = 1 \text{ para } 1 \leq i \leq 2^n.$$

Por otro lado, si $k \neq i$, existe por lo menos un j entre 1 y n tal que $t_{kj} = 1 - t_{ij}$, y para este j se tiene que

$$T_{P_i}(t_{k1}, \ldots, t_{kn}) = \begin{cases} t_{kj} & \text{si } t_{ij} = 1, \\ 1 - t_{kj} & \text{si } t_{ij} = 0. \end{cases}$$

Por lo tanto:

$$T_{P_i}(t_{k1}, \ldots, t_{kn}) = 0 \text{ para } k \neq i.$$

En resumen, tenemos que

$$T_{P_i}(t_{k1}, \ldots, t_{kn}) = \begin{cases} 1 & \text{si } k = i, \\ 0 & \text{si } k \neq i. \end{cases} \tag{1.11}$$

Sea $I = \{i \,|\, 1 \leq i \leq 2^n \text{ y } t_{in+1} = 1\}$. Como por hipótesis f no es idénticamente 0, resulta que I es no vacío y podemos escribir

$$I = \{i_1, \ldots, i_r\},$$

donde i_1 es el primer i tal que $t_{in+1} = 1$, i_2 el segundo, etc.

Finalmente, definamos la fórmula P del modo siguiente:

$$P = P_{i_1} \vee P_{i_2} \vee \cdots \vee P_{i_r}.$$

Se tiene que

$$T_P(t_{i1}, \ldots, t_{in}) = \max(T_{P_{i_1}}(t_{i1}, \ldots, t_{in}), \ldots, T_{P_{i_r}}(t_{i1}, \ldots, t_{in})).$$

De esta última igualdad y de (1.11) resulta que $T_P(t_{i1}, \ldots, t_{in}) = 1$ si y sólo si $i \in I$, esto es, si y sólo si $f(t_{i1}, \ldots, t_{in}) = 1$. Luego las funciones T_P y f toman el valor 1 en exactamente las mismas n-uplas de $\mathbf{B}^n$, y como en las restantes deben ambas tomar el valor 0, se tiene que $T_p(t_{i1}, \ldots, t_{in}) = f(t_{i1}, \ldots, t_{in})$ para toda n-upla $(t_{i1}, \ldots, t_{in}) \in \mathbf{B}^n$, y esto significa que $T_P = f$, con lo que termina la demostración.

La demostración del teorema anterior nos da un procedimiento efectivo para encontrar una fórmula correspondiente a una función booleana. Ilustraremos este procedimiento con un ejemplo.

a_1	a_2	a_3	$f(a_1, a_2, a_3)$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

Table 1.2: Función booleana de tres variables

Sea $f\colon \mathbf{B}^3 \to \mathbf{B}$ la función dada por la Tabla 1.2.
En este caso tenemos $i_1 = 2$, $i_2 = 4$, $i_3 = 5$, $i_4 = 6$ e $i_5 = 8$ y:

$$P_2 = \neg x_1 \wedge \neg x_2 \wedge x_3,$$

$$P_4 = \neg x_1 \wedge x_2 \wedge x_3,$$

$$P_5 = x_1 \wedge \neg x_2 \wedge \neg x_3,$$

$$P_6 = x_1 \wedge \neg x_2 \wedge x_3,$$

$$P_8 = \neg x_1 \wedge \neg x_2 \wedge \neg x_3.$$

Podemos concluir entonces que f es la tabla de verdad de la fórmula

$$P = P_2 \vee P_4 \vee P_5 \vee P_6 \vee P_8,$$

donde, como siempre, estamos suponiendo que asociamos a la izquierda, según lo convenido en la página 19.

Pero observemos que f es también la tabla de verdad de la fórmula $Q = (x_1 \vee x_3) \wedge (\neg x_2 \vee x_3)$, y también la tabla de verdad de la fórmula $R = (x_1 \to x_2) \to x_3)$.

Esto muestra que en general fórmulas distintas, construidas con las mismas variables, pueden tener la misma tabla de verdad. Intuitivamente, estas fórmulas expresan lo mismo. Esta idea intuitiva se formaliza en la siguiente:

Definición 1.2.9 Las fórmulas P y Q se dicen *equivalentes*, y se escribe $P \equiv Q$, si para todo valuación v se tiene que $v(P) = v(Q)$.

Es fácil verificar que $\equiv$ es una relación de equivalencia en el conjunto de las fórmulas, esto es, que se satisfacen las tres propiedades siguientes, donde P, Q y R denotan fórmulas arbitrarias:

Reflexiva: $P \equiv P$,

Simétrica: Si $P \equiv Q$, entonces $Q \equiv P$, y

Transitiva: Si $P \equiv Q$ y $Q \equiv R$, entonces $P \equiv R$.

Lema 1.2.10 *Si $var(P) = var(Q)$, entonces $P \equiv Q$ si y sólo si tienen la misma tabla de verdad.*

Demostración: Supongamos que $var(P) = var(Q) = \{x_1, \ldots, x_n\}$.
Si $T_P = T_Q$, entonces para toda valuación v se tiene que

$$v(P) = T_P(v(x_1), \ldots, v(x_n)) = T_Q(v(x_1), \ldots, v(x_n)) = v(Q)$$

y por lo tanto $P \equiv Q$.
Supongamos ahora que $P \equiv Q$. Sea $(a_1, \ldots, a_n) \in \mathbf{B}$ y sea v una valuación tal que $v(x_1) = a_i$ para $i = 1, \ldots, n$. Entonces tendremos que

$$T_P(a_1, \ldots, a_n) = v(P) = v(Q) = T_Q(a_1, \ldots, a_n)$$

Como $(a_1, \ldots, a_n) \in \mathbf{B}$ es arbitraria, esto significa que $T_P = T_Q$, c.q.d.

Observación 1.2.11 La hipótesis que $var(P) = var(Q)$ es esencial para la validez del Lema 1.2.10. En efecto, consideremos las fórmulas $P = (p_0 \vee p_1)$ y $Q = (p_0 \vee p_2)$. Si v es una valuación tal que $v(p_0) = v(p_1) = 0$ y $v(p_2) = 1$ (una tal valuación existe por el Teorema 1.2.2), se tiene que $v(P) = 0$ y $v(Q) = 1$. Luego P no es equivalente a Q. Sin embargo, es fácil verificar que $T_P(a_1, a_2) = T_Q(a_1, a_2)$ para todo par $(a_1, a_2) \in \mathbf{B} \times \mathbf{B}$.

Con un conjunto finito de variables proposicionales $\{x_1, \ldots, x_n\}$ se pueden construir infinitas fórmulas. Por ejemplo, ya vimos que usando sólo una variable x y sólo el conectivo $\neg$ se pueden obtener las fórmulas:

$$x, \neg x, \neg\neg x, \ldots, \neg \cdots \neg x, \ldots$$

Pero ¿cuántas de estas fórmulas son esencialmente distintas, esto es, no equivalentes? Algunos autores se refieren a este problema como *el poder de expresión del cálculo proposicional*.

La respuesta se obtiene fácilmente a partir del Lema 1.2.10: cada clase de equivalencia de fórmulas construidas a partir de las variables proposicionales (que suponemos distintas) $x_1, \ldots, x_n$ está determinada por una única función booleana, y como por el Teorema 1.2.8 toda tal función proviene de una fórmula construida a partir de $x_1, \ldots, x_n$, vemos que hay una biyección entre las clases de equivalencia de las fórmulas construidas a partir de las variables $x_1, \ldots, x_n$ y las funciones booleanas de n variables.

De estas consideraciones resulta inmediatamente que:

Teorema 1.2.12 *El número de clases de equivalencia de fórmulas que se pueden construir a partir de n variables proposicionales distintas es* 2^{2^n}.

El siguiente criterio de equivalencia de fórmulas es conceptualmente importante:

Teorema 1.2.13 *Las fórmulas P y Q son equivalentes si y sólo si $P \to Q$ y $Q \to R$ son ambas tautologías.*

Demostración: El teorema es una consecuencia inmediata de la siguiente propiedad, que a su vez resulta de las igualdades (1.9) y (1.5): *Para toda valuación v, $v(P) \leq v(Q)$ si y sólo si $v(P \to Q) = 1$.*

Las fórmulas de longitud modificada 1 ó 2 (ver página 14) son llamadas *literales*. Esto es, un literal es o bien una variable proposicional o bien la negación de una variable proposicional.

Una fórmula P se dice *normal disyuntiva* si es una disyunción de conjunciones de literales. Esto es, P es normal disyuntiva si (usando la convención de la página 19)

$$P = P_1 \vee P_2 \vee \ldots \vee P_r,$$

donde cada P_i es, a su vez, de la forma $\ldots x_1' \wedge x_2' \wedge \ldots \wedge x_n'$, con $x_i' = x_i$ ó $x_i' = \neg x_i$.

Observemos que *la fórmula P que se obtiene a partir de una función booleana f de n variables por el procedimiento indicado en la demostración del Teorema 1.2.8 es normal disyuntiva.*

Teorema 1.2.14 *Toda fórmula es equivalente a una fórmula normal disyuntiva.*

Demostración: Sea P una fórmula y $var(P) = \{x_1, \ldots, x_n\}$. Tenemos que la tabla de verdad de P, T_P, es una función booleana de n variables. Luego podemos aplicar el procedimiento indicado en la demostración del Teorema 1.2.8 para obtener una fórmula normal disyuntiva Q tal que $var(Q) = var(P)$ y $T_Q = T_P$. Por el Lema 1.2.10, resulta entonces que $Q \equiv P$, c.q.d.

Observemos que el procedimiento indicado en la demostración del teorema anterior *nos da un procedimiento efectivo para encontrar la forma normal disyuntiva de una fórmula dada.*

1.3 La consecuencia lógica.

La noción más importante considerada en lógica es la de *consecuencia.*

Sea S un conjunto de fórmulas, esto es, $S \subseteq \mathbf{Form}$, y sea P una fórmula. Para cada asignación de valores de verdad a las variables proposicionales algunas de las fórmulas de S podrán resultar verdaderas y otras falsas, y lo mismo ocurre con P. Intuitivamente parece natural pedir que para que P sea una consecuencia lógica de S, se cumpla que cualquier asignaciń de valores de verdad que haga verdaderas *todas* las fórmulas de S también haga verdadera a P.

Esta idea intuitiva es la que tomaremos como definición semántica de la relación de consecuencia. Comenzaremos por la siguiente:

Definición 1.3.1 Diremos que *una valuación v satisface a una fórmula P* si $v(P) = 1$, y diremos que *v satisface a un conjunto $S \subseteq \mathbf{Form}$* si v satisface a todas las fórmulas de S, esto es, si $v(Q) = 1$ para toda $Q \in S$. Una fórmula (o un conjunto de fórmulas) se dice *satisfacible* si existe una valuación que la (o lo) satisfaga, e *insatisfacible* en caso contrario.

Observar que toda valuación v satisface (trivialmente) al conjunto vacío.

Es también claro que una fórmula es satisfacible si y sólo si es una tautología o una contingencia. Luego *las fórmulas insatisfacibles coinciden con las falsedades.*

Definición 1.3.2 Sean $S \subseteq \mathbf{Form}$ y $P \in \mathbf{Form}$. Diremos que *P es una consecuencia de S*, y escribiremos $S \models P$, en el caso que toda valuación v que satisface a S también satisface a P. Esto es, si $v(Q) = 1$ para toda $Q \in S$, entonces $v(P) = 1$. Indicaremos con $\mathbf{Con}(S)$ al conjunto de las consecuencias de S: $\mathbf{Con}(S) = \{P \in \mathbf{Form} \mid S \models P\}$.

Observación 1.3.3 De la definición anterior resulta que $\emptyset \models P$ si y sólo si P es una tautología. Luego $\mathbf{Con}(\emptyset)$ *es el conjunto de las tautologías.*

El resultado siguiente pone de manifiesto la relación existente entre el conectivo binario $\rightarrow$ y la noción de consecuencia.

Teorema 1.3.4 (de la deducción, forma semántica) *Sean P, Q fórmulas y $S \subseteq \mathbf{Form}$. Entonces $Q \in \mathbf{Con}(S \cup \{P\})$ si y solamente si $P \rightarrow Q \in \mathbf{Con}(S)$.*

Demostración: Sea v una valuación que satisface a S, y supongamos que $v(P \rightarrow Q) = \max(\neg v(P), v(Q)) = 0$. Entonces $v(P) = 1$ y $v(Q) = 0$, lo que implica que v satisface a $S \cup \{P\}$ y $v(Q) = 0$. Esto muestra que si $Q \notin \mathbf{Con}(S)$, entonces $Q \notin \mathbf{Con}(R \cup \{P\})$.
Para probar la recíproca, supongamos que $P \rightarrow Q \in \mathbf{Con}(S)$, y sea v una valuación que satisfaga a $S \cup \{P\}$. Entonces

$$v(Q) = \max(0, v(Q)) = \max(\neg v(P), v(Q)) = v(P \rightarrow Q) = 1$$

lo que muestra que $Q \in \mathbf{Con}(S \cup \{P\})$, c.q.d.

Tomando $S = \emptyset$ en el teorema anterior y teniendo en cuenta la Observación 1.3.3, se obtiene el siguiente:

Corolario 1.3.5 *Para todo par de fórmulas P y Q se tiene que $\{P\} \models Q$ si y sólo si $P \rightarrow Q$ es una tautología.*

La relación entre consecuencia y satisfacibilidad está dada por el teorema siguiente, cuya demostración es inmediata a partir de las respectivas definiciones:

Teorema 1.3.6 *Para todo $S \subseteq \mathbf{Form}$ y para toda fórmula P, se tiene que $S \models P$ si y sólo si $S \cup \{\neg P\}$ es insatisfacible.*

Notemos que si en este teorema hacemos $S = \emptyset$, obtenemos que *una fórmula P es una tautología si y sólo si $\neg P$ es insatisfacible.*

Observación 1.3.7 Un conjunto finito y no vacío de fórmulas es satisfacible si y sólo si la conjunción de las fórmulas que lo componen es satisfacible. Luego: $\{Q_1, \ldots, Q_r\} \models P$ *si y sólo si* $Q_1 \wedge \cdots \wedge Q_r \wedge \neg P$ *es insatisfacible.*

Observación 1.3.8 Una fórmula P tal que $var(P) = \{x_1, \ldots, x_n\}$ es satisfacible si y sólo si existe una n-upla $(a_1, \ldots, a_n) \in \mathbf{B}^n$ tal que $T_P(a_1, \ldots, a_n) = 1$.

De las observaciones precedentes resulta que tenemos un procedimiento efectivo para poder determinar si un conjunto finito y no vacío de fórmulas es o no satisfacible: hacemos primero la conjunción de las fórmulas del conjunto, y luego vamos construyendo la tabla de verdad de esta conjunción hasta obtener una fila cuyo último valor sea 1. Si encontramos tal fila, el conjunto es satisfacible. Si en la última columna de la tabla figura solamente 0, entonces el conjunto es insatisfacible.

Observemos que por el Teorema 1.3.6, podemos aplicar este procedimiento para determinar si una fórmula es o no consecuencia de un conjunto finito de fórmulas.

Si bien la construcción de una tabla de verdad es un procedimiento efectivo para determinar si una dada fórmula es o no satisfacible, en el sentido que se hace en un número finito de pasos bien determinados, este número puede ser muy grande y por lo tanto el método puede resultar impracticable.

Para convencerse de esto, basta observar que si en P figuran n variables proposicionales, su tabla de verdad tendra 2^n filas y $n+1$ columnas, y podría ocurrir que el valor 1 apareciese sólo al final de la última fila.

Nos proponemos ahora describir otro procedimiento para determinar si un conjunto finito (y no vacío) de fórmulas es o no satisfacible.

Este método, aunque también requiere una cantidad de pasos del orden anterior, es más natural, pues es más próximo a la forma en que razonamos habitualmente,y, sobre todo, puede generalizarse para el cálculo de predicados, lo que no ocurre con las tablas de verdad. La idea básica se remonta a los trabajos del matemático aleman G.Gentzen publicados en 1934. Nuestra exposición seguirá la presentación del método publicada en 1968 por R.M. Smullyan , quien tuvo en cuenta ideas introducidas por E.W.Beth y K.Hintikka a mediados de la decada del 50[1]

Comenzaremos por el siguiente lema, cuya demostración resulta inmediatamente de la Definición 1.2.1 y de las fórmulas (1.5)–(1.9):

Lema 1.3.9 *Sean P,Q y R fórmulas, y v una valuación. Entonces se tienen las siguientes propiedades:*

[1]R.M. Smullyan, *First Order Logic*, Springer-Verlag, Berlin–New York, 1968. Reimpreso por Dover, N. York, 1995

($\mathbf{R}_{\neg}$) *v satisface a $\neg\neg P$ si y sólo si v satisface a P,*

($\mathbf{R}_{\vee}$) *v satisface a $P \vee Q$ si y sólo si v satisface a P o v satisface a Q,*

($\mathbf{R}_{\neg\vee}$) *v satisface a $\neg(P \vee Q)$ si y sólo si v satisface a $\neg P$ y v satisface a $\neg Q$,*

($\mathbf{R}_{\wedge}$) *v satisface a $P \wedge Q$ si y sólo si v satisface a P y v satisface a Q,*

($\mathbf{R}_{\neg\wedge}$) *v satisface a $\neg(P \wedge Q)$ si y sólo si v satisface a $\neg P$ o v satisface a $\neg Q$,*

($\mathbf{R}_{\rightarrow}$) *v satisface a $P \rightarrow Q$ si y sólo si v satisface a $\neg P$ o v satisface a Q, y*

($\mathbf{R}_{\neg\rightarrow}$) *$v$ satisface a $\neg(P \rightarrow Q)$ si y sólo si v satisface a P y v satisface a $\neg Q$.*

Las propiedades enunciadas en el lema anterior nos dan las reglas del comportamiento de los conectivos con respecto a la satisfacibilidad, y se pueden esquematizar en la forma siguiente:

$$\mathbf{R}_{\neg} \quad \frac{\neg\neg P}{P}$$

$$\mathbf{R}_{\vee} \quad \frac{(P \vee Q)}{P \mid Q} \qquad\qquad \mathbf{R}_{\neg\vee} \quad \frac{\neg(P \vee Q)}{\neg P, \neg Q}$$

$$\mathbf{R}_{\wedge} \quad \frac{(P \wedge Q)}{P, Q} \qquad\qquad \mathbf{R}_{\neg\wedge} \quad \frac{\neg(P \wedge Q)}{\neg P \mid \neg Q}$$

$$\mathbf{R}_{\rightarrow} \quad \frac{(P \rightarrow Q)}{\neg P \mid Q} \qquad\qquad \mathbf{R}_{\neg\rightarrow} \quad \frac{\neg(P \rightarrow Q)}{P, \neg Q}$$

La fórmula que figura encima de la raya horizontal en cada una de las reglas se llama *la premisa* de dicha regla, y la o las fórmulas que figuran debajo, *la o las conclusiones* de la regla.

Observación 1.3.10 En todos los casos se tiene que las conclusiones son o bien subfórmulas o bien negaciones de subfórmulas de la correspondiente premisa.

Tenemos dos tipos de reglas: las que llamaremos de tipo A, que tienen una única conclusión o dos conclusiones separadas por una coma, sin una barra vertical, y las que llamaremos de tipo B, que tienen dos conclusiones separadas por una barra vertical.

Las reglas de tipo A son $\mathbf{R}_{\neg}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$ y $\mathbf{R}_{\neg\rightarrow}$, y las de tipo B son $\mathbf{R}_{\vee}$, $\mathbf{R}_{\neg\wedge}$ y $\mathbf{R}_{\rightarrow}$.

Las reglas de tipo A significan que una valuación satisface a la premisa si y sólo si satisface a todas las conclusiones, y las de tipo B, que una valuación satisface a la premisa si y sólo si satisface a por lo menos una de las conclusiones.

Lema 1.3.11 *Toda fórmula que no es una variable proposicional o negación de una variable proposicional, es de la forma de la premisa de una (y sólo una) de las reglas* $\mathbf{R}_{\neg}$, $\mathbf{R}_{\vee}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$, $\mathbf{R}_{\neg\wedge}$, $\mathbf{R}_{\rightarrow}$ *o* $\mathbf{R}_{\neg\rightarrow}$.

Demostración: Sea P una fórmula. Si P no es una variable proposicional, entonces del Corolario 1.1.14 resulta que se puede presentar uno y sólo uno de los siguientes casos:

(i) $P = \neg T$,

(ii) $P = (Q \vee R)$,

(iii) $P = (Q \wedge R)$ o

(iv) $P = (Q \rightarrow R)$.

En los casos (ii), (iii) y (iv) P es de la forma de la premisa de $\mathbf{R}_{\vee}$, $\mathbf{R}_{\wedge}$ y $\mathbf{R}_{\rightarrow}$, respectivamente.

Consideremos el caso (i). Si T es una variable proposicional, entonces P es la negación de una variable proposicional. En caso contrario, otra vez por el Corolario 1.1.14 resulta que se puede presentar uno, y sólo uno, de los casos siguientes:

(a) $P = \neg T = \neg\neg Q$,

(b) $P = \neg T = \neg(Q \vee R)$,

(c) $P = \neg T = \neg(Q \wedge R)$ o

(d) $P = \neg T = \neg(Q \rightarrow R)$,

de donde resulta que P es de la forma de la premisa de $\mathbf{R}_{\neg}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\neg\wedge}$ o $\mathbf{R}_{\neg\rightarrow}$, c.q.d.

La alternativa al método de las tablas de verdad para determinar si una fórmula es satisfacible que mencionamos anteriormente, consiste en una aplicación sistemática de las reglas para los distintos conectivos. Antes de describir el método formalmente, daremos un par de ejemplos.

Como primer ejemplo, utilizaremos las reglas para averiguar si la fórmula

$$P = ((p_0 \vee p_1) \rightarrow p_2) \rightarrow ((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2)) \tag{1.12}$$

es o no una tautología.

Sabemos que P es una tautología si y sólo si $\neg P$ es insatisfacible.

Supongamos que $\neg P$ sea satisfacible. Entonces existiría una valuación v que la satisface. Veamos si esto es posible.

Como P es de la forma $Q \rightarrow R$, con $Q = (p_0 \vee p_1) \rightarrow p_2$ y $R = (p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2)$, la regla $\mathbf{R}_{\neg\rightarrow}$ es aplicable a $\neg P$, lo que implica que $(p_0 \vee p_1) \rightarrow p_2$ y $\neg((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2)))$ deberían ser satisfechas por la valuación v.

El siguiente árbol nos ayudará a visualizar la situación:

$$\neg(((p_0 \vee p_1) \rightarrow p_2) \rightarrow ((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2)))$$

$$(p_0 \vee p_1) \rightarrow p_2$$

$$\neg((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2))$$

La regla $\mathbf{R}_{\rightarrow}$ es aplicable a $(p_0 \vee p_1) \rightarrow p_2$, y por lo tanto v debería satisfacer a $\neg(p_0 \vee p_1)$ o a p_2. Para tener en cuenta ambas posibilidades, agregamos a nuestro árbol dos nuevos nodos, a la misma altura:

$$\neg(((p_0 \vee p_1) \rightarrow p_2) \rightarrow ((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2)))$$

$$(p_0 \vee p_1) \rightarrow p_2$$

$$\neg((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2))$$

$$\neg(p_0 \vee p_1) \qquad\qquad p_2$$

La regla $\mathbf{R}_{\neg\wedge}$ es aplicable a $\neg((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2))$, por lo que v debería satisfacer a $\neg(p_0 \rightarrow p_2)$ o a $\neg(p_1 \rightarrow p_2)$. Para seguir teniendo en cuenta todas las posibilidades, agregamos estas fórmulas como nuevos nodos terminales a cada uno de los dos nodos terminales anteriores, obteniendo el siguiente árbol:

$$\neg(((p_0 \vee p_1) \rightarrow p_2) \rightarrow ((p_0 \rightarrow p_2) \wedge (p_1 \rightarrow p_2)))$$

$$\begin{array}{cccc} & & (p_0 \vee p_1) \to p_2 & \\ & & \neg((p_0 \to p_2) \wedge (p_1 \to p_2)) & \\ & \neg(p_0 \vee p_1) & & p_2 \\ \neg(p_0 \to p_2) & \neg(p_1 \to p_2) & \neg(p_0 \to p_2) & \neg(p_1 \to p_2) \end{array}$$

La regla $\mathbf{R}_{\neg\to}$ es aplicable tanto a $\neg(p_0 \to p_2)$ como a $\neg(p_1 \to p_2)$. Por lo tanto deberemos agregar sucesivamente las conclusiones de esta regla a los nodos terminales que se encuentran debajo de las respectivas fórmulas, obteniendo:

$$\neg(((p_0 \vee p_1) \to p_2) \to ((p_0 \to p_2) \wedge (p_1 \to p_2)))$$

$$\begin{array}{cccc} & & (p_0 \vee p_1) \to p_2 & \\ & & \neg((p_0 \to p_2) \wedge (p_1 \to p_2)) & \\ & \neg(p_0 \vee p_1) & & p_2 \\ \neg(p_0 \to p_2) & \neg(p_1 \to p_2) & \neg(p_0 \to p_2) & \neg(p_1 \to p_2) \\ p_0 & p_1 & p_0 & p_1 \\ \neg p_2 & \neg p_2 & \neg p_2 & \neg p_2 \end{array}$$

Observemos que en las dos ramas de la derecha del árbol que estamos construyendo figuran las fórmulas p_2 y $\neg p_2$. Como no puede haber ninguna valuación que las satisfaga simultáneamente, los caminos indicados por estas dos ramas se nos "cierran", por lo que seguiremos aplicando las reglas sólo a las fórmulas que figuran en las dos ramas izquierdas. Indicamos este hecho colocando la marca $\times$ debajo de los nodos terminales de las dos ramas derechas. La regla $\mathbf{R}_{\neg\vee}$ es aplicable a $\neg(p_0 \vee p_1)$, por lo que debemos agregar sucesivamente $\neg p_0$ y $\neg p_1$ como nuevos nodos a cada uno de los nodos terminales que figuran debajo de $\neg(p_0 \vee p_1)$, obteniendo:

$$\neg(((p_0 \vee p_1) \to p_2) \to ((p_0 \to p_2) \wedge (p_1 \to p_2)))$$

$$\begin{array}{cccc} & & (p_0 \vee p_1) \to p_2 & \\ & & \neg((p_0 \to p_2) \wedge (p_1 \to p_2)) & \\ & \neg(p_0 \vee p_1) & & p_2 \\ \neg(p_0 \to p_2) & \neg(p_1 \to p_2) & \neg(p_0 \to p_2) & \neg(p_1 \to p_2) \end{array}$$

p_0	p_1	p_0	p_1
$\neg p_2$	$\neg p_2$	$\neg p_2$	$\neg p_2$
$\neg p_0$	$\neg p_0$	$\times$	$\times$
$\neg p_1$	$\neg p_1$		

En la primera rama de la izquierda figuran p_0 y $\neg p_0$, y en la segunda, p_1 y $\neg p_1$, por lo que debemos poner una marca $\times$ debajo de los nodos terminales de cada una de ellas, obteniendo:

$$\neg(((p_0 \vee p_1) \to p_2) \to ((p_0 \to p_2) \wedge (p_1 \to p_2)))$$

$$(p_0 \vee p_1) \to p_2$$

$$\neg((p_0 \to p_2) \wedge (p_1 \to p_2))$$

$$\neg(p_0 \vee p_1) \qquad\qquad p_2$$

$$\neg(p_0 \to p_2) \quad \neg(p_1 \to p_2) \quad \neg(p_0 \to p_2) \quad \neg(p_1 \to p_2)$$

p_0	p_1	p_0	p_1
$\neg p_2$	$\neg p_2$	$\neg p_2$	$\neg p_2$
$\neg p_0$	$\neg p_0$	$\times$	$\times$
$\neg p_1$	$\neg p_1$		
$\times$	$\times$		

Ahora en todos los nodos terminales del árbol figura la marca $\times$, lo que significa que todos los posibles caminos a seguir se nos han "cerrado" o "bloqueado". Luego, partiendo de la hipótesis de que v satisface a la fórmula $\neg P$ y examinando los valores que v le debería asignar a todas las fórmulas que podemos derivar de $\neg P$ por aplicación de las reglas, siempre llegamos a una contradicción. Por lo tanto podemos concluir que no existe una valuación que satisfaga a $\neg P$, y, por ende, que P es una tautología.

Como segundo ejemplo de aplicación de las reglas, veamos si la fórmula

$$Q = (p_0 \wedge p_1) \to \neg(p_0 \vee \neg p_1) \tag{1.13}$$

es o no una tautología. Como en el caso de la fórmula P del ejemplo anterior, debemos averiguar si $\neg Q$ es o no satisfacible. Supongamos entonces que exista una valuación v tal que $v(\neg Q) = 1$. Como la regla $\mathbf{R}_{\neg\to}$ es aplicable a

$\neg Q$, v debe satisfacer a $p_0 \wedge p_1$ y a $\neg\neg(p_0 \vee \neg p_1)$. Podemos entonces dibujar el árbol:

$$\neg((p_0 \wedge p_1) \rightarrow \neg(p_0 \vee \neg p_1))$$

$$p_0 \wedge p_1$$

$$\neg\neg(p_0 \vee \neg p_1)$$

Aplicando ahora la regla $\mathbf{R}_\neg$ a la órmula $\neg\neg(p_0 \vee \neg p_1)$, podemos agregar $p_0 \vee \neg p_1$ como nuevo nodo terminal:

$$\neg((p_0 \wedge p_1) \rightarrow \neg(p_0 \vee \neg p_1))$$

$$p_0 \wedge p_1$$

$$\neg\neg(p_0 \vee \neg p_1)$$

$$p_0 \vee \neg p_1$$

y aplicando $\mathbf{R}_\wedge$ a $p_0 \wedge p_1$, obtenemos:

$$\neg((p_0 \wedge p_1) \rightarrow \neg(p_0 \vee \neg p_1))$$

$$p_0 \wedge p_1$$

$$\neg\neg(p_0 \vee \neg p_1)$$

$$p_0 \vee \neg p_1$$

$$p_0$$

$$p_1$$

Finalmente, aplicando $\mathbf{R}_\vee$ a $p_0 \vee p_1$, nos queda:

$$\neg((p_0 \wedge p_1) \rightarrow \neg(p_0 \vee \neg p_1))$$

$$p_0 \wedge p_1$$

$$\neg\neg(p_0 \vee \neg p_1)$$

$$p_0 \vee \neg p_1$$

$$p_0$$

$$p_1$$

$$p_0 \qquad\qquad \neg p_1$$

$\times$

La rama de la derecha aparece cerrada por la marca $\times$. La rama de la izquierda está "saturada", en el sentido de que todas las fórmulas que figuran arriba de su nodo terminal p_0 (y que no son variables proposicionales o negaciones de variables proposicionales), fueron utilizadas. De este modo hemos "completado" el procedimiento.

Sea v una valuación tal que $v(p_0) = v(p_1) = 1$ (una tal valuación existe por el Corolario 1.2.4). Entonces "subiendo" por la rama de la izquierda vemos que v va satisfaciendo las fórmulas que figuran en la misma, en particular al nodo inicial. Luego $v(\neg Q) = 1$. Esto significa que $\neg Q$ es satisfacible, y por lo tanto que Q no es una tautología.

Vamos a formalizar ahora la construcción de los árboles que vimos en los dos ejemplos anteriores[2].

Para abreviar, en lo sucesivo diremos que *una fórmula P es de tipo A (o de tipo B)*, en lugar de decir que es de la forma de la premisa de una regla de tipo A (de tipo B), y llamaremos *conclusiones de P* a las conclusiones de la única regla que tiene por premisa a P.

Definición 1.3.12 Sea $\mathcal{A}$ un árbol cuyos nodos son fórmulas. Un árbol $\mathcal{A}'$ se dice *una extensión inmediata de* $\mathcal{A}$ si $\mathcal{A}'$ se obtiene de una de las siguientes maneras, donde T representa un nodo terminal de $\mathcal{A}$:

1. Si T tiene como antecesor una fórmula P de tipo A, se agrega una de las conclusiones de P como sucesor inmediato de T.

2. Si T tiene como antecesor una fórmula P de tipo B, se agregan las dos conclusiones de P como dos sucesores inmediatos de T.

En cualquiera de estos casos, se dice que la fórmula P fue *usada* en el nodo T.

Definición 1.3.13 Un árbol de fórmulas es *un árbol de refutación de una fórmula P* si y sólo si se lo obtiene haciendo un número finito de extensiones inmediatas a partir del árbol cuyo único nodo es la fórmula P. Un *árbol de refutación de un conjunto finito S de fórmulas* es un árbol de refutación de la conjunción de las fórmulas de S.

[2]En el Apéndice de este Capítulo se dan todas las nociones sobre árboles que usaremos en lo que sigue.

De la definición anterior resulta que *un árbol de fórmulas $\mathcal{A}$ es un árbol de refutación de una fórmula P si y sólo si existe una sucesión finita de árboles $\mathcal{A}_0, \ldots, \mathcal{A}_n$ tal que $\mathcal{A}_0$ tiene por único nodo a la fórmula P, $\mathcal{A}_n = \mathcal{A}$ y $\mathcal{A}_{i+1}$ es una extensión inmediata de $\mathcal{A}_i$, para $i = 1, \ldots, n-1$.*

Una rama de un árbol de fórmulas se dice *cerrada* si contiene a la vez a una fórmula y a su negación. En caso contrario, la rama se dice *abierta*.

Un árbol se dice *cerrado* si todas sus ramas son cerradas.

Una rama de un árbol de fórmulas se dice *saturada* si el conjunto formado por las fórmulas de sus nodos es *saturado* en el sentido de la siguiente definición:

Definición 1.3.14 Un conjunto S de fórmulas se dice *saturado* si y sólo si satisface las siguientes condiciones:

(S0) Una fórmula y su negación no pueden estar simultáneamente en S.

(S1) Si una fórmula de tipo A está en S, entonces sus dos conclusiones están en S.

(S2) Si una fórmula de tipo B está en S, al menos una de sus conclusiones está en S.

Un árbol de refutación se dice *completo* si cada una de sus ramas es cerrada o saturada. Notemos que, en particular, *todo árbol de refutación cerrado es un árbol completo.*

En los dos ejemplos que desarrollamos anteriormente obtuvimos árboles de refutación completos de las fórmulas definidas en (1.12) y (1.13). El primero es cerrado, mientras que el segundo tiene una rama saturada.

Observemos que el grado de complejidad de las conclusiones no es necesariamente menor que el grado de complejidad de las premisas. Por ejemplo, si $P = p_0 \to p_1$, $comp(P) = comp(\neg p_0)$. Esto hace que no podamos argumentar por inducción en el grado de complejidad al pasar de una fórmula a sus conclusiones.

Lo que sí disminuye al pasar de una fórmula a sus conclusiones es la longitud modificada (ver página 14). En efecto, se tiene el siguiente lema, que es análogo al Corolario 1.1.14.

Lema 1.3.15 *Para toda fórmula P se tiene que:*

(i) *Si $long^*(P) = 1$, entonces P es una variable proposicional,*

(ii) *Si $long^*(P) = 2$, entonces P es la negación de una variable proposicional,*

(iii) *Si $long^*(P) \geq 3$, entonces P es de tipo A o B, y la longitud modificada de sus conclusiones es menor que $long^*(P)$.*

Demostración: Resulta de la definición de longitud modificada y de examinar cada una de las reglas $\mathbf{R}_{\neg}$, $\mathbf{R}_{\vee}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$, $\mathbf{R}_{\neg\wedge}$, $\mathbf{R}_{\rightarrow}$ o $\mathbf{R}_{\neg\rightarrow}$, c. q. d.

Estamos ahora en condiciones de probar el siguiente:

Lema 1.3.16 *Toda fórmula P tiene por lo menos un árbol de refutación completo.*

Demostración: Por inducción en la longitud modificada.

Si $long^*(P) = 1$ o $long^*(P) = 2$, entonces por el Lema 1.3.15 P es una variable proposicional o la negación de una variable proposicional, y por consiguiente, el árbol de nivel 0 en cuyo único nodo figura P es un árbol de refutación completo de P.

Supongamos que $long^*(P) \geq 3$, y que toda fórmula X tal que $long^*(X) < long^*(P)$ admite un árbol de refutación completo.

Por el Lema 1.3.15, P es de tipo A o de tipo B, y la longitud modificada de sus conclusiones es menor que $long^*(P)$.

Supongamos primero que P es de tipo A, y sea Q una conclusión de P. Sea $\mathcal{A}_1$ el árbol que se obtiene poniendo Q debajo de P, y luego desarrollando un árbol completo para Q (que existe por la hipótesis inductiva). Como $\mathcal{A}_1$ se puede obtener por un número finito de extensiones inmediatas a partir del nodo inicial P, es un árbol de refutación de P. Si es cerrado, o si Q es la única conclusión de P (esto es, $P = \neg\neg Q$), entonces es un árbol de refutación completo de P. En caso contrario, sea R la otra conclusión de P, y agreguémosla como nuevo nodo terminal de todas las ramas abiertas de $\mathcal{A}_1$. Llamemos $\mathcal{A}_2$ al árbol que se obtiene desarrollando a partir de cada uno de estos nuevos nodos un árbol de refutación completo para R (que existe por la hipótesis inductiva). Como $\mathcal{A}_2$ se puede obtener haciendo un número finito de extensiones inmediatas a partir de P, es un árbol de refutación de P. Veamos que es completo. Si es cerrado, no hay nada que probar. Supongamos entonces que S sea una rama abierta de $\mathcal{A}_2$, y sea $S_1 = S \cap \mathcal{A}_1$. Sea X una fórmula de tipo A o B en S. Si $X = P$, entonces por la construcción de $\mathcal{A}_2$, sus dos conclusiones Q y R están en S. Si $X \neq P$ y $X \in S_1$, entonces como

$S_1 \setminus \{P\}$ es una rama abierta de un árbol completo de Q, sus conclusiones (si es de tipo A), o una de ellas (si es de tipo B) deben estar en $S_1 \subseteq S$. Si $X \in S \setminus S_1$, se puede aplicar el mismo argumento, pues $S \setminus S_1$ es una rama abierta de un árbol de refutación completo de R. Luego en cualquier caso S es una rama completa de $\mathcal{A}_2$.

Supongamos ahora que P es de tipo B, y sean Q y R sus dos conclusiones. Llamemos $\mathcal{A}$ al árbol que se obtine del siguiente modo: partiendo del nodo inicial P, se agregan dos nodos terminales Q y R, y árboles completos con origen en cada uno de estos nodos, que existen por la hipótesis inductiva. Como $\mathcal{A}$ se puede obtener de P haciendo un número finito de extensiones inmediatas, es un árbol de refutación de P. Si es cerrado, entonces es completo. Supongamos que S es una rama abierta de $\mathcal{A}$, y que X es una fórmula de tipo A o B que está en S. Si $X = P$, entonces por la construcción de $\mathcal{A}$, una de sus conclusiones está en S. Si $X \neq P$, la demostración se termina observando que $S \setminus \{P\}$ es una rama abierta de un árbol completo para Q o de un árbol completo para R, c. q. d.

Lema 1.3.17 *Si P es una fórmula satisfacible, entonces todo árbol de refutación de P tiene tiene al menos una rama abierta.*

Demostración: Sea P satisfacible, y probemos, por inducción en la altura, que todo árbol de refutación de P tiene por lo menos una rama satisfacible.

Si la altura del árbol es 0, entonces tiene un único nodo y por lo tanto una única rama, formada por la fórmula P, que es satisfacible.

Como hipótesis inductiva, supongamos que todo árbol de refutación de P de altura n tiene por lo menos una rama satisfacible.

Sea $\mathcal{A}$ un árbol de refutación de P de altura $n+1$, y sea $\mathcal{A}'$ el árbol que se obtiene eliminando todos los nodos de nivel $n+1$ de $\mathcal{A}$. Es claro que $\mathcal{A}'$ es un árbol de refutación de P de altura n. Luego, por la hipótesis inductiva, $\mathcal{A}'$ tiene por lo menos una rama abierta. Elijamos una, que llamaremos S.

Podría ser que S fuese también una rama de $\mathcal{A}$, y en tal caso ya encontramos una rama abierta de $\mathcal{A}$.

En caso contrario, el nodo terminal N de S en $\mathcal{A}'$ no sería un nodo terminal de $\mathcal{A}$, y habría dos posibilidades: o hay un único nodo terminal N' de $\mathcal{A}$ que sigue a N, o hay una bifurcación en dos nodos terminales de $\mathcal{A}$, N' y N''.

En el primer caso, N' es una de las conclusiones de una fórmula Q de tipo A que pertenece a S. Como toda valuación que satisface a Q debe satisfacer a N', resulta que $S \cup \{N'\}$ es una rama satisfacible de $\mathcal{A}$.

En el segundo caso, los nodos N' y N'' son las conclusiones de una fórmula R de tipo B perteneciente a S. Como toda valuación que satisface a R también debe satisfacer a N' o a N'', resulta que por lo menos una de las dos ramas de A, $S \cup \{N'\}$ o $S \cup \{N''\}$ es satisfacible.

Vemos así que $\mathcal{A}$ tiene una rama satisfacible, y como las ramas satisfacibles no pueden ser cerradas, resulta que $\mathcal{A}$ tiene una rama abierta, c.q.d.

Toda rama abierta de un árbol de refutación completo es saturada. La importancia de los conjuntos saturados está dada por el siguiente:

Lema 1.3.18 *Todo conjunto saturado de fórmulas es satisfacible.*

Demostración: Sea S un conjunto saturado de fórmulas y definamos la función $f\colon \mathbf{Var} \to \mathbf{B}$ del modo siguiente, donde, como siempre, **Var** denota el conjunto de las variables proposicionales:

$$f(p_n) = \begin{cases} 1 & \text{si} \quad p_n \in S \\ 0 & \text{si} \quad p_n \notin S \end{cases}$$

Si $v = v_f$ es la valuación que extiende a f, vamos a ver que $v(P) = 1$ para toda fórmula $P \in S$, lo que probará que S es satisfacible.

Haremos la demostración por inducción en la longitud modificada.

Si $P \in S$ y $long^*(P) = 1$, entonces P es una variable proposicional, digamos $P = p_n$, y por la definición de V, $v(P) = v(p_n) = f(p_n) = 1$.

Si $long^*(P) = 2$, entonces P es la negación de una variable proposicional, digamos $P = \neg p_n$. Como $\neg p_n \in S$, por (S0) resulta que $p_n \notin S$, de donde resulta que $f(p_n) = 0$, y por consiguiente que $v(P) = 1$.

Supongamos ahora que $long^*(P) \geq 3$, y que $v(X) = 1$ para toda fórmula X tal que $X \in S$ y $long^*(X) < long^*(P)$.

Sabemos (Lema 1.3.15) que P es de tipo A o de tipo B. En el primer caso, de la propiedad (S1) resulta que sus conclusiones están en S, y por la hipótesis inductiva, v les asigna el valor 1, lo que implica que también $v(P) = 1$. Si P es de tipo B, de (S2) resulta que al menos una de sus conclusiones está en S, y por la hipótesis inductiva, v le asigna el valor 1, lo que implica que $v(P) = 1$.

Luego $v(P) = 1$ para toda fórmula $P \in S$, c.q.d.

Observación 1.3.19 En la demostración del lema anterior sólo se usó el caso particular siguiente de la propiedad (S0):

(S0p) Una variable proposicional y su negación no pueden estar simultáneamente en S.

Luego se tiene que un conjunto S de fórmulas es saturado si y sólo si satisface las propiedades (S0p), (S1) y (S2). En efecto, todo conjunto saturado satisface (S0p), que es un caso particular de (S0). Supongamos ahora que S satisface (S0p), (S1) y (S2). Entonces por la demostración del Lema 1.3.18 resulta que existe una valuación v tal que $v(P) = 1$ para todo $P \in S$, lo que implica que S debe cumplir (S0).

Los lemas anteriores se pueden resumir en el siguiente teorema:

Teorema 1.3.20 *Las siguientes propiedades son equivalentes para todo conjunto finito y no vacío* $S \subseteq$ **Form***:*

(i) *S es satisfacible.*

(ii) *Todo árbol de refutación de S tiene por lo menos una rama abierta.*

(iii) *Existe un árbol de refutación de S completo con una rama abierta.*

Demostración: (i) implica (ii): Resulta del Lema 1.3.17.
(ii) implica (iii): Por el Lema 1.3.16 existe un árbol de refutación de S completo, y por la hipótesis (ii) éste debe tener por lo menos una rama abierta.
(iii) implica (i): Sea $\mathcal{A}$ un árbol de refutación completo de S. Una rama abierta R de $\mathcal{A}$ constituye un conjunto saturado de fórmulas, y por el Lema 1.3.18, R es satisfacible. En particular, la fórmula del nodo inicial de $\mathcal{A}$ es satisfacible, esto es, la conjunción de las fórmulas de S es satisfacible. Luego, por la Observación 1.3.7, resulta que S es satisfacible, c. q. d.

Corolario 1.3.21 *Las siguientes propiedades son equivalentes para todo conjunto finito y no vacío* $S \subseteq$ **Form***:*

(i) *S es insatisfacible.*

(ii) *S tiene un árbol de refutación cerrado.*

(iii) *Todo árbol de refutación completo de S es cerrado.*

Corolario 1.3.22 *Las siguientes propiedades son equivalentes para toda fórmula P:*

(i) *P es una tautología.*

(ii) *$\neg P$ tiene un árbol de refutación cerrado.*

(iii) *Todo árbol de refutación completo de $\neg P$ es cerrado.*

Los dos últimos corolarios nos dan un método alternativo al de las tablas de verdad para determinar si un conjunto finito es o no satisfacible o si una fórmula es o no una tautología. Más aún, en caso de encontrar un árbol completo con una rama abierta, asignando a las variables proposicionales los valores 0 y 1 de acuerdo con lo especificado en la demostración del Lema 1.3.18 podemos definir una valuación que satisface a S. Usamos este procedimiento para encontrar una valuación que satisface a la fórmula dada por (1.12).

De las Observaciones 1.3.3 y 1.3.7 resulta que podemos dar la siguiente caracterización de las consecuencias de un conjunto finito S de fórmulas:

a) Si $S = \emptyset$, entonces $P \in \mathbf{Con}(S)$ si y sólo si la fórmula $\neg P$ tiene un árbol de refutación cerrado.

b) Si $S = \{Q_1, \ldots, Q_n\}$, entonces $P \in \mathbf{Con}(S)$ si y sólo si la fórmula $Q_1 \wedge \ldots \wedge Q_n \wedge \neg P$ tiene un árbol de refutación cerrado.

Estas observaciones sugieren dar la siguiente:

Definición 1.3.23 Una *deducción de una fórmula P a partir de un conjunto de fórmulas S* es un árbol de refutación cerrado de $\neg P$ o de una conjunción de $\neg P$ con fórmulas de S. Una fórmula P se dice *deducible a partir de* $S \subseteq \mathbf{Form}$ si existe una deducción de P a partir de S. El conjunto de las fórmulas deducibles a partir de $S \subseteq \mathbf{Form}$ será denotado por $\mathbf{Ded}(S)$.

Como los árboles de refutación son objetos sintácticos, esto es, construídos a partir de fórmulas siguiendo reglas en las que no intervienen explícitamente los valores de verdad, la noción de deducción es también puramente sintáctica. Corresponde a la idea intuitiva de demostrar una proposición a partir de ciertas hipótesis utilizando reglas de inferencia. En nuestro caso las reglas de inferencia son las de los árboles de refutación, y formalizamos la idea de demostrar "por reducción al absurdo", habitual en matemática.

Las condiciones a) y b) señaladas más arriba muestran que para todo conjunto finito de fórmulas S se tiene que $\mathbf{Con}(S) = \mathbf{Ded}(S)$.

Luego, para los conjuntos finitos de fórmulas hemos reducido la noción *semántica* de consecuencia a la noción *sintáctica* de demostración. Vamos a ver ahora que esta reducción es también posible para conjuntos infinitos.

Teorema 1.3.24 (Teorema de Compacidad) *Sea S un conjunto infinito de fórmulas. Si todo subconjunto finito de S es satisfacible, entonces S es satisfacible.*

Demostración: Sea $S = \{Q_0, Q_1, \ldots Q_n, Q_{n+1}, \ldots\}$ un conjunto infinito de fórmulas, y supongamos que todo subconjunto finito de S es satisfacible. Sea $\mathcal{A}_0$ un árbol de refutación completo de Q_0. Como $\{Q_0\}$ es un subconjunto finito de S, $\mathcal{A}_0$ debe tener por lo menos una rama abierta. Sea $\mathcal{A}_1$ un árbol completo obtenido a partir de agregar Q_1 como nuevo nodo terminal a todas las ramas abiertas de $\mathcal{A}_0$. Notemos que $\mathcal{A}_1$ es un árbol de refutación del subconjunto finito $\{Q_0, Q_1\} \subseteq S$, luego tiene por lo menos una rama abierta. Sea $\mathcal{A}_2$ un árbol completo obtenido a partir de agregar la fórmula Q_2 como nuevo nodo terminal de cada rama abierta de $\mathcal{A}_1$. Entonces $\mathcal{A}_2$ es un árbol de refutación de $\{Q_0, Q_1, Q_2\}$, y por lo tanto tiene ramas abiertas. Podemos proseguir el procedimiento indicado para obtener una sucesión infinita de árboles de refutación completos $\mathcal{A}_0, \mathcal{A}_1, \mathcal{A}_2, \ldots \mathcal{A}_n, \mathcal{A}_{n+1}, \ldots$, de modo que $\mathcal{A}_{n+1}$ se obtiene completando el árbol construído agregando la fórmula Q_{n+1} como nuevo nodo terminal de cada una de las ramas abiertas de $\mathcal{A}_n$. Como $\mathcal{A}_{n+1}$ es un árbol de refutación del conjunto finito $\{Q_0, Q_1, \ldots, Q_n, Q_{n+1}\} \subseteq S$, debe tener ramas abiertas y por lo tanto el procedimiento no se detiene.

Entonces $\mathcal{A} = \bigcup_{n\geq 0} \mathcal{A}_n$ es un árbol binario infinito, y por el Lema de König (ver Apéndice) tiene una rama infinita, que llamaremos R.

Para cada n, $R_n = R \cap \mathcal{A}_n$ es una rama de $\mathcal{A}_n$. Esta rama debe ser abierta, pues de lo contrario no se hubiese prolongado a R_{n+1}. Como $R = \bigcup_{n\geq 0} R_n$, resulta que R no es cerrada. Veamos que R es saturada. Supongamos que una fórmula Q de tipo A estén R. Entonces $Q \in R_n$ para algún n. Como R_n es una rama abierta de $\mathcal{A}_n$, y este árbol es completo, R_n es saturada. Luego ambas conclusiones de Q están en $R_n \subseteq R$. Un argumento similar prueba que si una fórmula de tipo B está en R, entonces una de sus conclusiones también lo está.

Como R es saturado, por el Lema 1.3.18, R es un conjunto satisfacible, y como por la construcción de los árboles $\mathcal{A}_n$, $S \subseteq R$, resulta que también S es satisfacible, c.q.d.

Corolario 1.3.25 *Sea S un conjunto de fórmulas. Una fórmula P es consecuencia de S si y sólo si P es consecuencia de un subconjunto finito de S.*

Demostración: Como cualesquiera que sean los conjuntos de fórmulas S y R se tiene que $S \subseteq R$ implica que $\mathbf{Con}(S) \subseteq \mathbf{Con}(R)$, resulta que toda consecuencia de un subconjunto finito de S es también consecuencia de S. Para probar la recíproca, sea $P \in \mathbf{Con}(S)$. Por el Teorema 1.3.6, $S \cup \{\neg P\}$ es insatisfacible, y por el Teorema de Compacidad, existe un subconjunto finito $F \subseteq S \cup \{\neg P\}$ tal que F es insatisfacible. Entonces $G = F \setminus \{\neg P\}$ es un subconjunto finito de S (eventualmente vacío), y otra vez por el Teorema 1.3.6, se tiene que $P \in \mathbf{Con}(G) \subseteq \mathbf{Con}(S)$, c.q.d.

Observación 1.3.26 En algunos textos se da el nombre de Teorema de Compacidad al Corolario 1.3.25. En realidad, el Teorema 1.3.24 y el Corolario 1.3.25 son equivalentes, en el sentido que uno se deduce del otro. En efecto, ya vimos como el Corolario 1.3.25 se obtiene a partir del Teorema 1.3.24. Veamos ahora como se puede demostrar el Teorema 1.3.24 a partir del Corolario 1.3.25. Para ello, supongamos cierto el enunciado del Corolario 1.3.25 y sean S un conjunto de fórmulas insatisfacible y $P \in S$ (notemos que como S es insatisfacible, no puede ser vacío). Por el Teorema 1.3.6, $\neg P \in \mathbf{Con}(S \setminus \{P\})$, y por el Corolario 1.3.25 existe un subconjunto finito $F \subseteq S \setminus \{P\}$ tal que $\neg P \in \mathbf{Con}(F)$. Luego, otra vez por el Teorema 1.3.6, $F \cup \{\neg\neg P\}$ es insatisfacible, y por lo tanto también lo es $F \cup \{P\}$. Luego hemos encontrado un subconjunto finito de S que es insatisfacible. Por lo tanto, si todo subconjunto finito de S es satisfacible, debe ser S satisfacible.

Finalmente, podemos extender la equivalencia entre consecuencia y deducibilidad a conjuntos arbitrarios de fórmulas:

Corolario 1.3.27 *Para todo conjunto S de fórmulas,*

$$\mathbf{Con}(S) = \mathbf{Ded}(S).$$

Concluiremos este capítulo con un ejemplo que ilustrará las aplicaciones del Teorema de Compacidad y también como se pueden usar las ideas desarrolladas en el estudio del clculo proposicional en la consideración de situaciones concretas.

Comenzaremos por un lema sobre conjuntos parcialmente ordenados finitos, cuya demostración no requiere de los métodos del cálculo proposicional.

Lema 1.3.28 *Toda relación de orden parcial $\leq$ sobre un conjunto finito A se puede extender a una relación de orden total $\preceq$ sobre A. Esto es, existe una relación de orden total $\preceq$ sobre A tal que para todo par de elementos x, y de A, si $x \leq y$ entonces $x \preceq y$.*

Demostración: Por indución en el número n de elementos de A.

Si $n = 1$, no hay nada que probar.

Supongamos que hemos probado el lema para todo conjunto con n elementos, y supongamos que A tiene $n + 1$ elementos. Sea $u \in A$. Entonces $B = A \setminus \{u\}$, con el orden $\leq$ heredado de A, es un conjunto parcialmente ordenado con n elementos, y por la hipótesis inductiva, existe una relación de orden total $\preceq$ sobre B que extiende a $\leq$. Podemos numerar los elementos de B de modo que $x_1 \prec x_2 \prec \ldots \prec x_n$.

Usando esta notación vamos a extender el orden total $\preceq$ a todo A del modo siguiente:

Definimos

$$K = \{i \in \{1, 2, \ldots, n\} \mid x_i \leq u\}$$

Si $K = \emptyset$, podemos agregar u como primer elemento, obteniendo el siguiente orden total para A:

$$u \prec x_1 \prec x_2 \prec \ldots \prec x_n$$

Si $K \neq \emptyset$, sea j el mayor elemento de K. Si $j = n$, agregamos u como último elemento, obteniendo el siguiente orden total sobre A:

$$x_1 \prec x_2 \prec \ldots \prec x_n \prec u$$

Si j es menor que n, colocamos u entre x_j y x_{j+1}, obteniendo:

$$x_1 \prec x_2 \prec \ldots \prec x_j \prec u \prec x_{j+1} \prec \ldots \prec x_n$$

En todos los casos, $\preceq$, extendido de la manera indicada, es un orden total sobre A que extiende a $\leq$, c.q.d.

Como aplicación del Teorema de Compacidad veremos que el lema anterior vale también para ciertos conjuntos infinitos.

Un conjunto A se dice *numerable* cuando existe una biyección del conjunto $\mathbf{N}$ sobre A[3]. Un conjunto A es numerable si y sólo si se lo puede escribir en

[3]La existencia de conjuntos infinitos no numerables (por ejemplo, el conjunto de los números reales) fue un importante descubrimiento de G. Cantor hacia fines del siglo pasado.

forma de una sucesión infinita,

$$A = \{a_0, a_1, \ldots, a_n \ldots\} \text{ con } a_m \neq a_n \text{ si } m \neq n. \tag{1.14}$$

En efecto, si $f: \mathbf{N} \to \mathbf{A}$ es una biyección, para escribir A en la forma (1.14) basta poner $a_n = f(n)$ para todo $n \in \mathbf{N}$. Recíprocamente, si A está en la forma (1.14), la función $f: \mathbf{N} \to \mathbf{A}$ definida por $f(n) = a_n$ es una biyección.

Es fácil verificar que la función $\psi: \mathbf{N} \times \mathbf{N} \to \mathbf{N}$ definida por

$$\psi((m, n)) = 2^m(2n + 1) - 1$$

es una biyección. Luego su inversa $\varphi = \psi^{-1}: \mathbf{N} \to \mathbf{N} \times \mathbf{N}$ es también una biyección, y por lo tanto el producto cartesiano $\mathbf{N} \times \mathbf{N}$ es numerable.

Teorema 1.3.29 *Toda relación de orden parcial $\leq$ sobre un conjunto numerable A se puede extender a una relación de orden total sobre A.*

Demostración: Como A es numerable, puede ser escrito en la forma (1.14). Con cada par de elementos a_m, a_n de A, vamos a asociar la variable proposicional $q_{m,n} = p_{\psi((m,n))}$, donde ψ es la biyección de $\mathbf{N} \times \mathbf{N}$ en $\mathbf{N}$ definida antes de enunciar este teorema. Intuitivamente, vamos a interpretar $q_{m,n}$ como el enunciado "el elemento a_m es menor o igual que el elemento a_n".

Sea S el conjunto formado por las siguientes fórmulas:

(i) $q_{n,n}$, para todo $n \in \mathbf{N}$,

(ii) $\neg(q_{m,n} \wedge q_{n,m})$, para $m \neq n$,

(iii) $(q_{m,n} \wedge q_{n,r}) \to q_{m,r}$, para todo m, n y r en $\mathbf{N}$,

(iv) $q_{m,n} \vee q_{n,m}$, para todo m y n en $\mathbf{N}$,

(v) $q_{m,n}$, para todo m y n tales que $a_m \leq a_n$.

Para probar el teorema, basta probar que el conjunto S es satisfacible. En efecto, sea v una valuación que satisface a S y definamos una relación binaria $\preceq$ sobre A del siguiente modo:

$$a_m \preceq a_n \text{ si y sólo si } v(q_{m,n}) = 1 \tag{1.15}$$

Las fórmulas de tipo (i), (ii) y (iii) grantizan que $\preceq$ es reflexiva, antisimétrica y transitiva, esto es, un orden parcial, y las de tipo (iv) que este orden es

total. Finalmente, las fórmulas de tipo (v) aseguran que $a_m \leq a_n$ implica que $a_m \preceq a_n$.

Probaremos, entonces, que S es satisfacible.

Sea F un subconjunto finito de S, y sea B el subconjunto finito de A formado por los elementos de A cuyo subíndice figura en alguna de las fórmulas de F. Por ejemplo, si $q_{3,5}$ está en F, entonces a_3 y a_5 deben estar en B.

El conjunto finito B está parcialmente ordenado por la relación $\leq$ heredada de A, y por el Lema 1.3.28, se puede extender a un orden total $\preceq$ sobre B.

Definamos la función $f\colon \mathbf{Var} \to \mathbf{B}$ por:

$$f(q_{m,n}) = \begin{cases} 1 & \text{si } q_{m,n} \in S \text{ y } a_m \preceq a_n \\ 0 & \text{en caso contrario} \end{cases}$$

Es fácil verificar que la valuación v_f satisface a F.

Luego todo subconjunto finito de S es satisfacible, y por el Teorema de Compacidad, es S satisfacible, c.q.d.

1.4 Apéndice: árboles y lema de König.

Llamaremos *árbol* a un conjunto parcialmente ordenado $\mathcal{A}$ con primer elemento a_0 y tal que para todo $a \in \mathcal{A}$, el conjunto de los antecesores de a, esto es $\{x \in \mathcal{A} \mid x < a\}$, es finito y totalmente ordenado (con el orden heredado de $\mathcal{A}$).

Para cada $a \in \mathcal{A}$ definimos *el nivel de* a, que denotaremos por $niv(a)$, como el número de antecesores de a. Observemos que $niv(a_0) = 0$, puesto que el primer elemento no tiene antecesores.

Diremos que a es un *antecesor inmediato de* b, y que b es un *sucesor inmediato de* a, en el caso en que $a < b$ y no existe $c \in \mathcal{A}$ tal que $a < c < b$. Observemos que en un árbol todo elemento distinto del primero tiene un único antecesor inmediato, pero que en cambio no hay restricciones sobre el número de sucesores inmediatos.

Notemos que si b es un sucesor inmediato de a se tiene que $niv(b) = niv(a) + 1$, pero en general esta igualdad no implica que b sea un sucesor inmediato de a.

Un elemento $a \in \mathcal{A}$ se dice *terminal* si no tiene sucesores, *simple* si tiene un único sucesor inmediato, y un *punto de ramificación* si tiene más de un sucesor inmediato.

Un árbol se dice *finitamente generado* si todo elemento tiene un número finito de sucesores inmediatos (considerando que los puntos terminales tienen 0 sucesores inmediatos).

Un árbol se dice *finito* si tiene un número finito de elementos.

Se verifica fácilmente que el conjunto $\mathbf{N}$ de los números naturales con su orden natural es un ejemplo de un árbol finitamente generado que no es finito.

Intuitivamente, el número $niv(a)$ mide "la distancia" del elemento a al origen a_0. Por eso resulta natural definir*la altura del árbol* $\mathcal{A}$, que indicaremos $alt(\mathcal{A})$, como el supremo de los niveles de los elementos de $\mathcal{A}$:

$$alt(A) = sup\{niv(a) \mid a \in \mathcal{A}\}$$

Como $niv(a)$ es un número natural para cada $a \in \mathcal{A}$, resulta que $alt(\mathcal{A}) = k < \infty$ si y sólo si existe un $b \in \mathcal{A}$ tal que $niv(b) = k$ y $niv(a) \leq k$ para todo $a \in \mathcal{A}$, y resulta que $alt(\mathcal{A}) = \infty$ si y sólo si para todo número natural n existe $a_n \in \mathcal{A}$ tal que $niv(a_n) = n$.

Es claro que todo árbol finito tiene altura finita. El conjunto cuyos elementos son el conjunto vacío y los subconjuntos unitarios de $\mathbf{N}$ (es decir,

los conjuntos cuyo único elemento es un número natural), ordenado por inclusión, es un ejemplo de un árbol infinito de altura finita (más precisamente, de altura uno).

La demostración del lema siguiente es muy fácil, y la dejamos a cargo del lector:

Lema 1.4.1 *Todo árbol finitamente generado y de altura finita es finito.*

Una *rama de un árbol* $\mathcal{A}$ es un subconjunto totalmente ordenado maximal de $\mathcal{A}$, esto es, una rama de $\mathcal{A}$ es un subconjunto $R \subseteq \mathcal{A}$ que satisface estas dos propiedades:

(R1) R es totalmente ordenado con el orden heredado de $\mathcal{A}$,

(R2) Si S es tal que $R \subset S \subseteq \mathcal{A}$, entonces S no es totalmente ordenado con el orden heredado de $\mathcal{A}$.

Si a es un elemento terminal de $\mathcal{A}$, es claro que

$$Rm(a) = \{x \in \mathcal{A} \,|\, x \leq a\}$$

es una rama de $\mathcal{A}$, que se llama *la rama determinada por* a.

No es difícil demostrar que todas las ramas de un árbol finito $\mathcal{A}$ están determinadas por un elemento terminal de $\mathcal{A}$.

Si $\mathcal{A}$ es un árbol infinito, entonces sus ramas infinitas son los subconjuntos $\{a_o, a_1, \ldots, a_n, \ldots\} \subseteq \mathcal{A}$ tales que $a_n < a_{n+1}$ y $niv(a) = n$ para todo $n \in \mathbf{N}$.

El lema siguiente es un resultado clásico de la teoría de árboles, que usamos para probar el Teorema de Compacidad del Cálculo Proposicional.

Lema 1.4.2 (D. König, 1926) *Todo árbol infinito y finitamente generado tiene al menos una rama infinita.*

Demostración: Sea $\mathcal{A}$ un árbol finitamente generado pero con infinitos elementos. Diremos que un elemento de $\mathcal{A}$ es *bueno* si tiene infinitos sucesores, y que es *malo* en caso contrario. Como $\mathcal{A}$ es infinito, el primer elemento a_0 es bueno. Observemos ahora que todo elemento bueno debe tener al menos un sucesor inmediato bueno. En efecto, como cada elemento tiene sólo un número finito de sucesores inmediatos, si todos ellos fuesen malos el número total de sucesores sería finito, y el elemento no podría ser bueno. Por lo tanto,

como a_0 es bueno, debe tener por lo menos un sucesor inmediato bueno. Elijamos uno, y llamémoslo a_1. Como a_1 es bueno, debe tener por lo menos un sucesor inmediato bueno. Elijamos uno, y llamémoslo a_2. Continuando en esta forma definiremos una sucesión infinita de elementos de $\mathcal{A}$ tal que $a_n < a_{n+1}$ y $niv(a_n) = n$ para todo $n \in \mathbf{N}$, c. q. d.

Observación 1.4.3 El hecho de que $\mathcal{A}$ sea finitamente generado es esencial para la validez del Lema de König, como lo muestra el ejemplo de los subconjuntos unitarios de $\mathbf{N}$ ya mencionado.

Chapter 2

Cálculo de predicados

2.1 Vocabularios

El alfabeto básico de todo lenguaje de primer orden consta de los siguientes tipos de símbolos, que se denominan *símbolos lógicos*:

- Variables: $v_0, v_1, \ldots, v_n, \ldots$. Como en el caso proposicional, v_n es una abreviatura del símbolo v seguido de n barras $|$.
- Conectivos proposicionales: $\vee, \wedge, \rightarrow, \neg$.
- Cuantificadores: $\forall$ (cuantificador universal) y $\exists$ (cuantificador existencial).
- Símbolos auxiliares:(,).

Además de los símbolos lógicos, el alfabeto específico de cada lenguaje de primer orden contiene:

- *Símbolos de constantes.* Indicaremos con $\mathcal{C}$ al conjunto de los símbolos de constante. Puede ser $\mathcal{C} = \emptyset$.
- *Símbolos de funciones n-arias (o de n variables).* Para cada $n \geq 1$, indicaremos con $\mathcal{F}_n$ al conjunto de los símbolos de funciones n-arias, y $\mathcal{F} := \bigcup_{n\geq 1} \mathcal{F}_n$. Pude ser $\mathcal{F} = \emptyset$.
- *Símbolos de predicados n-arios.* Para cada $n \geq 1$, indicaremos con $\mathcal{P}_n$ al conjunto de los símbolos de predicados n-arios, y $\mathcal{P} := \bigcup_{n\geq 1} \mathcal{P}_n$. Se exige que $\mathcal{P} \neq \emptyset$, esto es, *que haya al menos un símbolo de predicado.*

Los símbolos de constante, de funciones y de predicados forman *el vocabulario* de un lenguaje de primer orden.

Como los símbolos lógicos son comunes a todos los lenguajes de primer orden, *un lenguaje de primer orden queda determinado por su vocabulario*, que simbolizaremos $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$.

2.2 Términos y fórmulas

Recordemos una vez más que *el alfabeto de un lenguaje de primer orden está formado por los símbolos lógicos y el vocabulario.*

Definición 2.2.1 Una lista de símbolos del alfabeto A de un lenguaje de primer orden es un *término* si y sólo si se la puede obtener aplicando un número finito de veces las siguientes reglas:

(T1) Las variables son términos.

(T2) Los símbolos de constante son términos.

(T3) Si $t_1, ..., t_n$ son términos y $\mathbf{f}$ es un símbolo de función n-aria, entonces $\mathbf{f}(t_1 \dots t_n)$ es un término.

Como en la definición de fórmulas proposicionales, para precisar el significado de "aplicar un número finito de veces las reglas (T1), (T2) y (T3)" introduciremos la siguiente:

Definición 2.2.2 Una *cadena de formación de términos* es una sucesión finita $t_1, \dots, t_n$ de elementos de A^* que satisface las siguientes condiciones:

(CFT) Para cada i tal que $1 \leq i \leq n$, se tiene que o bien t_i es una variable o un símbolo de constante, o bien existen un símbolo de función k-aria $\mathbf{f}$ y k índices $i_1, \dots, i_k$, todos estrictamente menores que i, tales que $t = \mathbf{f}(t_{i_1} \dots t_{i_k})$.

El número $n \geq 1$ se llama la *longitud* de la cadena de formación.

La Definición 2.2.1 puede expresarse ahora en la siguiente forma:

Una lista t de símbolos del alfabeto de un lenguaje de primer orden es un término si y sólo si existe una cadena de formación de términos $t_1, \dots, t_n$ tal que $t = t_n$.

Llamaremos *grado de complejidad de un término* t, y lo denotaremos por $comp(t)$, al número de símbolos de función que figuran en la expresión t, contados tanta veces como aparezcan.

Observemos que $comp(t) = 0$ si y sólo si t es una variable o una constante, y que si $t = \mathbf{f}(t_1 \dots t_k)$, donde $\mathbf{f}$ es un símbolo de función k-aria y los t_i son términos, entonces

$$comp(t) = comp(t_1) + \cdots + comp(t_k) + 1. \tag{2.1}$$

Con los términos y los símbolos de predicados podemos construir los elementos básicos de nuestro lenguaje, que llamaremos *fórmulas atómicas*:

Definición 2.2.3 Una *fórmula atómica* de un lenguaje de primer orden con vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$ es una lista de símbolos de la forma $\mathbf{P}(t_1 \dots t_k)$, donde $\mathbf{P}$ es un símbolo de predicado k-ario y $t_1, \dots, t_k$ son términos.

Estamos ahora en condiciones de definir las fórmulas, que a su vez nos permitirán definir los enunciados de primer orden, que son el objetivo fundamental de nuestro estudio.

Definición 2.2.4 Una lista de símbolos del alfabeto de un lenguaje de primer orden con vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$ es una *fórmula* si y sólo si se la puede obtener aplicando un número finito de veces las siguientes reglas:

(F1) Las fórmulas atómicas son fórmulas.

(F2) Si φ es una fórmula, también lo es $\neg\varphi$.

(F3) Si φ y ψ son fórmulas, también lo son las expresiones $(\varphi \vee \psi)$, $(\varphi \wedge \psi)$ y $(\varphi \to \psi)$.

(F4) Si φ es una fórmula y x denota una variable, entonces $\forall x\varphi$ y $\exists x\varphi$ son fórmulas.

Dejamos al cuidado del lector precisar la definición anterior, definiendo, por analogía con los casos de fórmulas proposicionales y de términos, la noción de *cadena de formación de fórmulas*.

Llamaremos *grado de complejidad de una fórmula* φ, y lo denotaremos por $comp(\varphi)$, al número de conectivos y cuantificadores que figuran en φ, contados tantas veces como aparezcan.

Observemos que

(2.2) $comp(\varphi) = 0$ si y sólo si φ es una fórmula atómica.

(2.3) $$comp(\neg\varphi) = comp(\varphi) + 1.$$

(2.4) $$comp(\varphi \vee \psi) = comp(\varphi \wedge \psi) = comp(\varphi \to \psi) = comp(\varphi) + com(\psi) + 1.$$

(2.5) $$comp(\forall x\varphi) = comp(\exists x\varphi) = comp(\varphi) + 1.$$

Para poder definir los enunciados de un lenguaje de primer orden, necesitamos introducir los conceptos apariciones libres y ligadas de una variable en una fórmula. Intuitivamente, una variable x aparece *ligada* en una fórmula φ si está afectada por un cuantificador $\forall x$ o $\exists x$. En caso contrario, se dice que aparece *libre* en φ. Por ejemplo, si $\mathbf{P}$ y $\mathbf{Q}$ son símbolos de predicado binario y φ es la fórmula

$$\forall x\mathbf{P}(x, y) \to \exists y\mathbf{Q}(x, y),$$

tanto la primera como la segunda aparición de x en φ son ligadas, mientras que la tercera es libre. En cambio, la primera aparición de y es libre, mientras que la segunda y tercera son ligadas. La definición precisa de apariciones libres y ligadas de una variable en una fórmula se hace por induccin del modo siguiente:

Definición 2.2.5 Si φ es una fórmula atómica, entonces hay un único $k \geq 1$, un único $\mathbf{P} \in \mathcal{P}_k$ y una única k-upla de términos $t_1, \ldots, t_k$ tales que $\varphi = \mathbf{P}(t_1 \ldots t_k)$. Las variables que figuran en φ son las que figuran en alguno de los términos t_i, $i = 1, \ldots, k$. Todas las apariciones de todas las variables que aparecen en φ son libres.

Sea ahora φ una fórmula tal que $comp(\varphi) > 0$ y supongamos que hemos definido las apariciones libres de variables en toda fórmula ψ tal que $comp(\psi) < comp(\varphi)$. Entonces definiremos las apariciones libres de variables en φ de acuerdo a los casos posibles, a saber:

1. $\varphi = \neg\psi$. Las apariciones libres de una variable en φ son las apariciones libres de esta variable en ψ.

2. $\varphi = \psi \vee \eta$, ó $\varphi = \psi \wedge \eta$ ó $\varphi = \psi \to \eta$. Las apariciones libres de una variable en φ son las apariciones libres de esta variable en ψ y en η.

3. $\varphi = \forall x\psi$ o $\varphi = \exists x\psi$. Todas las apariciones de la variable x en φ son ligadas. Para toda variable y distinta de x, las apariciones libres de y en φ son las apariciones libres de y en ψ.

Definición 2.2.6 Un *enunciado* de un lenguaje de primer orden es una fórmula en la que no figuran variables libres.

Si φ es una fórmula en la que figuran libres las variables $x_1, \ldots, x_k$, entonces $\forall x_1 \cdots \forall x_k \varphi$ y $\exists x_1 \cdots \exists x_k \varphi$ son enunciados, que se llaman *la clausura universal de* φ y *la clausura existencial de* φ, respectivamente.

A continuación veremos otro procedimiento para obtener enunciados a partir de fórmulas.

Sean s y t términos de un lenguaje L, y sea x una variable. Con $t(x/s)$ designamos al término que se obtiene sustituyendo todas las apariciones de la variable x en t por s. Más precisamente, $t(x/s)$ se define por inducción en la complejidad de t del modo siguiente:

1. $comp(t) = 0$. Entonces t es una constante o una variable. Si $t \neq x$, entonces $t(x/s) = t$. Si $t = x$, entonces $t(x/s) = s$.

2. Sea $n > 0$ y supongamos que hemos definido $u(x/s)$ para todo término u tal que $comp(u) < n$. Sea $comp(t) = n$. Entonces hay un único $k \geq 1$, un único $\mathbf{f} \in \mathcal{F}_k$ y una única k-upla de términos $t_1, \ldots, t_k$ tales que $t = \mathbf{f}(t_1 \ldots t_k)$. Como $comp(t_i) < comp(t) = n$, por la hipótesis inductiva hemos definido $t_i(x/s)$ para $i = 1, 2, \ldots, k$. Entonces

$$t(x/s) = \mathbf{f}(t_1(x/s) \ldots t_k(x/s)).$$

Vamos ahora a definir la sustitución de una variable por un término en una fórmula.

Supongamos que $\varphi = \forall v_1 \mathbf{P}(v_1 v_2)$ y $s = \mathbf{f}(\mathbf{c})$, donde $\mathbf{P} \in \mathcal{P}_2$, $\mathbf{f} \in \mathcal{F}_1$ y $\mathbf{c} \in \mathcal{C}$.

Si sustituímos las apariciones de v_1 por s en φ, obtenemos la expresión $\forall \mathbf{f}(\mathbf{c}) \mathbf{P}(\mathbf{f}(\mathbf{c}) v_2)$, que no es una fórmula, puesto que el cuantificador universal aparece aplicado a un símbolo de función, mientras que la definición de fórmula permite aplicar los cuantificadores sólo a variables. Esto indica que debemos restringir la sustitución únicamente a variables que figuren libres en una fórmula.

Por otro lado, si φ es como antes, pero hacemos $s = v_1$, al sustituir la variable libre v_2 por s obtenemos $\forall v_1 \mathbf{P}(v_1 v_1)$. Esta expresión es una fórmula, pero intuitivamente vemos que tiene un sentido distinto de la original, pues estamos sustituyendo la variable libre v_2 por la variable v_1, que aparece afectada por el cuantificador universal.

Para evitar estos problemas nos restringiremos a sustituir *variables libres* por *términos sin variables*.

Sean φ una fórmula, x una variable y t un término sin variables de un lenguaje L. Con $\varphi(x/t)$ indicamos la expresión que se obtiene sustituyendo todas las apariciones libres de x en φ por t. Más precisamente, podemos definir $\varphi(x/t)$ por inducción en la complejidad de φ del modo siguiente:

Sea $comp(\varphi) = 0$. Entonces hay un único $k \geq 1$, un único $\mathbf{P} \in \mathcal{P}_k$ y una única k-upla de términos $t_1, \ldots, t_k$ tales que $\varphi = \mathbf{P}(t_1 \ldots t_k)$. Definimos

$$\varphi(x/t) = \mathbf{P}(t_1(x/t) \ldots t_k(x/t)).$$

Sea $n > 0$ y supongamos que hemos definido $\psi(x/t)$ para toda fórmula ψ tal que $comp(\psi) < n$. Sea $comp(\varphi) = n$. Definiremos $\varphi(x/t)$ de acuerdo a los casos posibles, a saber:

1. $\varphi = \neg\psi$. Entonces $\varphi(x/t) = \neg\psi(x/t)$.

2. $\varphi = \psi * \eta$, donde $*$ representa uno de los conectivos $\vee$, $\wedge$ ó $\rightarrow$. Definimos $\varphi(x/t) = \psi(x/t) * \eta(x/t)$.

3. $\varphi = Qy\psi$, donde Q denota uno de los cuantificadores $\forall$ ó $\exists$. Si $x = y$, definimos $\varphi(x/t) = \varphi$. Si $x \neq y$, definimos $\varphi(x/t) = Qy\psi(x/t)$.

2.3 Interpretación de lenguajes de primer orden

En lo que sigue supondremos que hemos fijado un lenguaje de primer orden L, determinado por el vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$.

Una *interpretación* $\mathcal{I}$ del vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$ consiste de los siguientes elementos:

(I1) Un conjunto $U_{\mathcal{I}} \neq \emptyset$, llamado *el universo de la interpretación* $\mathcal{I}$,

(I2) Para cada símbolo de constante $\mathbf{c} \in \mathcal{C}$, un elemento $c_{\mathcal{I}} \in U_{\mathcal{I}}$,

(I3) Para cada símbolo de función $\mathbf{f} \in \mathcal{F}_k$, una función $f_{\mathcal{I}}$ de k variables sobre el universo $U_{\mathcal{I}}$: $f_{\mathcal{I}}\colon U_{\mathcal{I}}^k \to U_{\mathcal{I}}$,

(I3) Para cada símbolo de predicado $\mathbf{P} \in \mathcal{P}_k$, una relación k-aria $P_{\mathcal{I}}$ sobre el universo $U_{\mathcal{I}}$, esto es, un subconjunto $P_{\mathcal{I}}$ del producto cartesiano $U_{\mathcal{I}}^k = \overbrace{U_{\mathcal{I}} \times \cdots \times U_{\mathcal{I}}}^{k \text{ veces}}$.

Luego una interpretación $\mathcal{I}$ del vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$ está determinada por la estructura $\mathcal{E}(\mathcal{I}) := \langle U_{\mathcal{I}}, \{c_{\mathcal{I}}\}_{c \in \mathcal{C}}, \{f_{\mathcal{I}}\}_{f \in \mathcal{F}}, \{P_{\mathcal{I}}\}_{P \in \mathcal{P}} \rangle$, que se denomina *la estructura relacional determinada por* $\mathcal{I}$.

Dada una interpretación $\mathcal{I}$ del vocabulario del lenguaje L, vamos a definir, por inducción en el grado de complejidad, una interpretación $t_{\mathcal{I}} \in U_{\mathcal{I}}$ para los términos t *sin variables* de L del modo siguiente:

1. Sea $comp(t) = 0$. Como en t no figuran variables, debe ser $t = \mathbf{c} \in \mathcal{C}$. Definimos:
$$t_{\mathcal{I}} = c_{\mathcal{I}} \in \mathcal{U}_{\mathcal{I}}.$$

2. Sea $n > 0$ y supongamos que hemos definido $s_{\mathcal{I}}$ para todo término sin variables s de complejidad menor que n. Si t es un término sin variables de complejidad n, entonces hay un único $k \geq 1$, un único $\mathbf{f} \in \mathcal{F}_k$ y una única k-upla de términos $(t_1, \ldots, t_k)$ tales que $t = \mathbf{f}(t_1 \ldots t_k)$. Como $comp(t_i) < comp(t) = n$, por la hipótesis inductiva están definidas las interpretaciones $t_{i_{\mathcal{I}}} \in \mathcal{U}_{\mathcal{I}}$ para $i = 1, 2, \ldots, k$. Entonces definimos $t_{\mathcal{I}}$ del modo siguiente:
$$t_{\mathcal{I}} = f_{\mathcal{I}}(t_{1_{\mathcal{I}}}, t_{2_{\mathcal{I}}} \ldots t_{k_{\mathcal{I}}}) \in \mathcal{U}_{\mathcal{I}}.$$

Diremos que el término sin variables t *designa* o *nombra* al elemento $t_{\mathcal{I}}$ del universo $\mathcal{U}_{\mathcal{I}}$.

La definición anterior nos da un procedimiento para calcular $t_{\mathcal{I}}$: reemplazar cada uno de los símbolos de constante que figuran en t por los elementos del universo $U_{\mathcal{I}}$ que les asigna la interpretación $\mathcal{I}$, e ir aplicando sucesivamente a los elementos de $U_{\mathcal{I}}$ así obtenidos las funciones que $\mathcal{I}$ le

asigna a los símbolos de función que figuran en t, en el orden en que aparecen.

De este procedimiento para calcular $t_{\mathcal{I}}$ resulta inmediatamente que:

Proposición 2.3.1 *Sean $\mathcal{I}$ y $\mathcal{J}$ dos interpretaciones del vocabulario del lenguaje L tales que $U_{\mathcal{I}} = U_{\mathcal{J}}$. Si para un término sin variables t de L se tiene que:*

1. *$c_{\mathcal{I}} = c_{\mathcal{J}}$ para todo símbolo de constante $\mathbf{c}$ que figura en t, y*
2. *$f_{\mathcal{I}} = f_{\mathcal{J}}$ para todo símbolo de función $\mathbf{f}$ que figura en t*

entonces:

$$t_{\mathcal{I}} = t_{\mathcal{J}}.$$

En otras palabras, *la interpretación de un término sin variables t sólo depende de las interpretaciones de los símbolos de constante y de función que figuran en t.* Esto suele expresarse diciendo que la interpretación de los términos es *una propiedad local.*

Para interpretar los enunciados de L, comenzaremos por extender nuetro lenguaje original L a un lenguaje $L(\mathcal{I})$, que se obtiene agregando al vocabulario de L el universo $\mathcal{U}_{\mathcal{I}}$ como un nuevo conjunto de constantes. Esto es, el vocabulario de $L(\mathcal{I})$ es $\langle \mathcal{C} \cup \mathcal{U}_{\mathcal{I}}, \mathcal{F}, \mathcal{P} \rangle$.

La interpretación $\mathcal{I}$ se extiende a una interpretación $\mathcal{I}'$ del nuevo lenguaje $L(\mathcal{I})$ del modo siguiente:

1. $\mathcal{U}_{\mathcal{I}'} = \mathcal{U}_{\mathcal{I}}$,
2. Para todo $\mathbf{c} \in \mathcal{C}$, $\mathbf{c}_{\mathcal{I}'} = \mathbf{c}_{\mathcal{I}}$,
3. Para todo $\mathbf{f} \in \mathcal{F}$, $\mathbf{f}_{\mathcal{I}'} = \mathbf{f}_{\mathcal{I}}$,
4. Para todo $\mathbf{P} \in \mathcal{P}$, $\mathbf{P}_{\mathcal{I}'} = \mathbf{P}_{\mathcal{I}}$, y
5. Para todo $a \in \mathcal{U}_{\mathcal{I}}$, $a_{\mathcal{I}'} = a$.

Esto es, la interpretación $\mathcal{I}'$ tiene el mismo universo que $\mathcal{I}$, y coincide con $\mathcal{I}$ en todos los símbolos del vocabulario de L, mientras que interpreta cada elemento del universo por sí mismo.

El lenguaje extendido $L(\mathcal{I})$ tiene términos sin variables que no son términos de L. Por ejemplo, cada $a \in \mathcal{U}_{\mathcal{I}}$. Por otro lado, todo término sin variables t

de L es también un término sin variables de $L(\mathcal{I})$, y por la Proposición 2.3.1 resulta que

$$t_{\mathcal{I}'} = t_{\mathcal{I}}. \tag{2.6}$$

Análogamente, si bien hay enunciados de $L(\mathcal{I})$ que no son enunciados de L, es claro que todo enunciado de L es también un enunciado de $L(\mathcal{I})$.

Vamos ahora a interpretar los enunciados de $L(\mathcal{I})$ como proposiciones acerca de la estructura relacional determinada por la interpretación $\mathcal{I}$, esto es, como enunciados acerca de $\mathcal{E}(\mathcal{I})$ que pueden ser verdaderos o falsos. Por lo tanto, la interpretación consistirá en asignarle un valor de verdad $V_{\mathcal{I}}(\varphi)$ a cada enunciado φ de $L(\mathcal{I})$. En particular, le estaremos asignando un valor de verdad a cada uno de los enunciados del lenguaje original L.

Como era de esperar, la asignación de valores de verdad la haremos por inducción en la complejidad de los enunciados.

Sea φ un enunciado de $L(\mathcal{I})$ de complejidad 0. Entonces hay un único $k \geq 1$, un único $\mathbf{P} \in \mathcal{P}_k$ y una única k-upla de términos $t_1, \ldots, t_k$ tales que $\varphi = \mathbf{P}(t_1 \ldots t_k)$. Como φ no tiene variables libres, de la Definición 2.2.5 resulta que los t_i deben ser términos sin variables del lenguaje $L(\mathcal{I})$, y por lo tanto designan elementos $t_{\mathcal{I}'}$ del universo $\mathcal{U}_{\mathcal{I}'} = \mathcal{U}_{\mathcal{I}}$, para $i = 1, \ldots, k$. Luego $(t_{1_{\mathcal{I}'}}, \ldots, t_{k_{\mathcal{I}'}}) \in \mathcal{U}_{\mathcal{I}}^k$. Definimos $V_{\mathcal{I}}(\varphi) = 1$ si $(t_{1_{\mathcal{I}'}}, \ldots, t_{k_{\mathcal{I}'}}) \in P_{\mathcal{I}}$ y $V_{\mathcal{I}}(\varphi) = 0$ si $(t_{1_{\mathcal{I}'}}, \ldots, t_{k_{\mathcal{I}'}}) \notin P_{\mathcal{I}}$.

Sea $n > 0$ y supongamos que hemos definido $V_{\mathcal{I}}(\psi)$ para todo enunciado ψ de $L(\mathcal{I})$ tal que $comp(\psi) < n$. Si φ es un enunciado de complejidad n, se puede presentar uno, y sólo uno, de los siguientes casos:

1. Existe un único enunciado ψ de $L(\mathcal{I})$ tal que $\varphi = \neg\psi$.

2. Existe un único par de enunciados ψ, η de $L(\mathcal{I})$ tal que $\varphi = (\psi \vee \eta)$.

3. Existe un único par de enunciados ψ, η de $L(\mathcal{I})$ tal que $\varphi = (\psi \wedge \eta)$.

4. Existe un único par de enunciados ψ, η de $L(\mathcal{I})$ tal que $\varphi = (\psi \rightarrow \eta)$.

5. Existe un único enunciado ψ de $L(\mathcal{I})$ y una única variable x tal que $\varphi = \forall x\psi$.

6. Existe un único enunciado ψ de $L(\mathcal{I})$ y una única variable x tal que $\varphi = \exists x\psi$.

Como en los casos 1, 2, 3 y 4 se tiene que $comp(\psi) < n$ y $comp(\eta) < n$, por la hipótesis inductiva están definidos los valores de verdad $V_{\mathcal{I}}(\psi)$ y $V_{\mathcal{I}}(\eta)$.

En el caso 1, definimos $V_{\mathcal{I}}(\varphi) := 1 - V_{\mathcal{I}}(\psi)$.

En el caso 2, definimos $V_{\mathcal{I}}(\varphi) := \max(V_{\mathcal{I}}(\psi), V_{\mathcal{I}}(\eta))$.

En el caso 3, definimos $V_{\mathcal{I}}(\varphi) := \min(V_{\mathcal{I}}(\psi), V_{\mathcal{I}}(\eta))$.

En el caso 4, definimos $V_{\mathcal{I}}(\varphi) := \max(1 - V_{\mathcal{I}}(\psi), V_{\mathcal{I}}(\eta))$.

Para considerar los casos 5 y 6, observemos primero que como φ es un enunciado, esto es, no tiene variables libres, de la Definición 2.2.5 resulta que la fórmula ψ puede tener a lo sumo a x como variable libre. Por lo tanto, para para cualquier elemento $a \in \mathcal{U}_{\mathcal{I}}$ se tiene que $\psi(x/a)$ es un enunciado de $L(\mathcal{I})$ y $comp(\psi(x/a)) = comp(\psi) < n$. Luego por la hipótesis inductiva está definido $V_{\mathcal{I}}(\psi(x/a))$ para todo $a \in \mathcal{U}_{\mathcal{I}}$.

En el caso 5 definimos

$$V_{\mathcal{I}}(\varphi) := \inf\{V_{\mathcal{I}}(\psi(x/a))\}_{a \in \mathcal{U}_{\mathcal{I}}}. \tag{2.7}$$

Esto es, $V_{\mathcal{I}}(\varphi) = 1$ si y sólo si $V_{\mathcal{I}}(\psi(x/a)) = 1$ para todo elemento $a \in \mathcal{U}_{\mathcal{I}}$.

En el caso 6, definimos

$$V_{\mathcal{I}}(\varphi) := \sup\{V_{\mathcal{I}}(\psi(x/a))\}_{a \in \mathcal{U}_{\mathcal{I}}}. \tag{2.8}$$

Esto es, $V_{\mathcal{I}}(\varphi) = 1$ si y sólo si existe al menos un elemento $a \in \mathcal{U}_{\mathcal{I}}$ tal que $V_{\mathcal{I}}(\psi(x/a)) = 1$.

Esto completa la definición inductiva de $V_{\mathcal{I}}(\varphi)$.

Insistimos que, en particular, está definido el valor de verdad $V_{\mathcal{I}}(\varphi)$ *para todo enunciado φ del lenguaje original L*. Más aún, se ve que la necesidad de considerar el lenguaje extendido $L(\mathcal{I})$ aparece sólo al considerar los casos 5 y 6. En estos dos casos debemos hacer referencia a los elementos del universo de la interpretación, y para ello necesitamos un lenguaje que nos permita mencionarlos.

La siguiente proposición se deduce fácilmente de la Proposición 2.3.1 y de la definición inductiva de los valores de verdad de los enunciados.

Proposición 2.3.2 *Sean $\mathcal{I}$ y $\mathcal{J}$ dos interpretaciones del vocabulario del lenguaje L tales que $U_{\mathcal{I}} = U_{\mathcal{J}}$. Si para un enunciado φ de L se tiene que:*

1. *$c_{\mathcal{I}} = c_{\mathcal{J}}$ para todo símbolo de constante* **c** *que figura en φ,*

2. *$f_{\mathcal{I}} = f_{\mathcal{J}}$ para todo símbolo de función* **f** *que figura en φ, y*

3. $P_{\mathcal{I}} = P_{\mathcal{J}}$ *para todo símbolo de predicado* $\mathbf{P}$ *que figura en* φ*,*

entonces:

$$V_{\mathcal{I}}(\varphi) = V_{\mathcal{J}}(\varphi).$$

Dado un enunciado φ del lenguaje L, indicaremos con $\mathcal{C}_\varphi$, $\mathcal{F}_\varphi$ y $\mathcal{P}_\varphi$ respectivamente a los conjuntos de símbolos de constante, de función y de predicado que figuran en la expresión de φ. Obtenemos así un vocabulario finito $\langle \mathcal{C}_\varphi, \mathcal{F}_\varphi, \mathcal{P}_\varphi \rangle$, que determina un "sublenguaje" de L, que denotaremos por L_φ.

La Proposición 2.3.2 significa que los valores de verdad de φ dependen sólo de las interpretaciones del vocabulario restringido $\langle \mathcal{C}_\varphi, \mathcal{F}_\varphi, \mathcal{P}_\varphi \rangle$. Por eso se dice que *el valor de verdad de un enunciado es una propiedad local.*

Ya hemos observado que una forma de obtener enunciados es sustituir las variables libres de una fórmula φ por términos sin variables del lenguaje L. Cuando consideramos una interpretación $\mathcal{I}$ de L, podemos también sustituir las variables libres de φ por los elementos del universo $U_{\mathcal{I}}$ que designan dichos términos, obteniendo así un enunciado del lenguaje ampliado $L(\mathcal{I})$. El siguiente lema nos dice que no hay ambigüedad, pues $\mathcal{I}$ les asigna a ambos enunciados el mismo valor de verdad.

Lema 2.3.3 *Si* φ *es una fórmula de* L *con una variable libre, que denotaremos por* x *y si* t *es un término sin variables de* L*, entonces* $\varphi(x/t)$ *es un enunciado de* L *y para toda interpretación* $\mathcal{I}$ *se tiene que*

$$V_{\mathcal{I}}(\varphi(x/t)) = V_{\mathcal{I}}(\varphi(x/t)). \tag{2.9}$$

Demostración: A lo largo de esta demostración, I denotará una interpretación del lenguaje L, y t un término sin variables de L.

Sea s un término del lenguaje ampliado $L(\mathcal{I})$ con a lo sumo una variable, que denotaremos por x. Entonces $s(x/t)$ es un término sin variables de $L(\mathcal{I})$, y por lo tanto designa al elemento $s(x/t)_{\mathcal{I}} \in U_{\mathcal{I}}$. Por otro lado, t designa al elemento $t_{\mathcal{I}}$, y la sustitución $s(x/t_{\mathcal{I}})$ también designa un elemento del universo $U_{\mathcal{I}}$. Comenzaremos por verificar, por inducción en la complejidad de s, que:

$$s(x/t)_{\mathcal{I}'} = s(x/t_{\mathcal{I}})_{\mathcal{I}'} \tag{2.10}$$

En efecto, sea $comp(s) = 0$. Si $s \neq x$, entonces $s(x/t) = s = s(x/t_{\mathcal{I}})$, y la igualdad (2.10) resulta trivialmente. Si $s = x$, entonces $s(x/t) = t$ y

$s(x/t_{\mathcal{I}}) = t_{\mathcal{I}}$. Como por la condición 5 en la definición de $\mathcal{I}'$ se tiene que $t_{\mathcal{I}} = (t_{\mathcal{I}})_{\mathcal{I}'}$, la igualdad (2.10) resulta de (2.6).

Sea $n > 0$ y supongamos que (2.10) se cumple para todos los términos de $L(\mathcal{I})$ de complejidad menor que n. Si $comp(s) = n$, entonces existen un único $k \geq 1$, un único $\mathbf{f} \in \mathcal{F}_k$ y una única k-upla de términos de $L(\mathcal{I})$, $(s_1, \ldots, s_k)$, tales que $s = \mathbf{f}(s_1, \ldots, s_k)$. Por la hipótesis inductiva se tiene que $s(x/t)_{\mathcal{I}'} = s(x/t_{\mathcal{I}})_{\mathcal{I}'}$ para $i = 1, \ldots, k$. Luego por la condición 3 en la definición de $\mathcal{I}'$ resulta que

$$s(x/t)_{\mathcal{I}'} = f_{\mathcal{I}}(s_1(x/t)_{\mathcal{I}'}, \ldots, s_k(x/t)_{\mathcal{I}'}) =$$

$$f_{\mathcal{I}'}(s_1(x/t_{\mathcal{I}})_{\mathcal{I}'}, \ldots, s_k(x/t_{\mathcal{I}})_{\mathcal{I}'}) = s(x/t_{\mathcal{I}})_{\mathcal{I}'},$$

lo que completa la demostración de (2.10).

Supongamos ahora que φ es una fórmula del lenguaje ampliado $L(\mathcal{I})$ con una única variable libre x. Probaremos la igualdad (2.9) por inducción en la complejidad de φ.

Supongamos primero que φ es una fórmula atómica, digamos $\varphi = \mathbf{P}(s_1 \ldots s_k)$, con $\mathbf{P} \in \mathcal{P}_r$ y $s_1, \ldots, s_r$ términos de $L(\mathcal{I})$. Como x es la única variable libre de φ, x es la única variable que puede figurar en los términos $s_1, \ldots, s_r$. Luego $\varphi(x/t) = \mathbf{P}(s_1(x/t), \ldots, s_r(x/t))$ es un enunciado, y se tiene que

(2.11) $V_{\mathcal{I}}(\varphi(x/t)) = 1$ si y sólo si $(s_1(x/t)_{\mathcal{I}'}, \ldots, s_r(x/t)_{\mathcal{I}'}) \in P_{\mathcal{I}}$.

Por otra parte, tenemos que:

(2.12) $$V_{\mathcal{I}}(\varphi(x/t_{\mathcal{I}})) = 1 \text{ si y sólo si}$$

$$(s_1(x/t_{\mathcal{I}})_{\mathcal{I}'}, \ldots, s_r(x/t_{\mathcal{I}})_{\mathcal{I}'}) \in P_{\mathcal{I}}.$$

Ahora por la igualdad (2.10) se tiene que

(2.13) $$(s_1(x/t)_{\mathcal{I}'}, \ldots, s_r(x/t)_{\mathcal{I}'}) = (s_1(x/t_{\mathcal{I}})_{\mathcal{I}'}, \ldots, s_r(x/t_{\mathcal{I}})_{\mathcal{I}'}).$$

Como sólo hay dos posibles valores de verdad, de (2.11), (2.12) y (2.13) resulta que $V_{\mathcal{I}}(\varphi(x/t)) = V_{\mathcal{I}}(\varphi(x/t_{\mathcal{I}}))$. Hemos probado así el lema para las fórmulas de complejidad cero.

Sea ahora $comp(\varphi) = n > 0$ y supongamos que hemos probado que $V_{\mathcal{I}}(\psi(x/t)) = V_{\mathcal{I}}(\varphi(x/t_{\mathcal{I}}))$, para toda fórmula ψ de $L(\mathcal{I})$ con x como única variable libre y $comp(\psi) < n$.

Si $\varphi = \neg\psi$ ó $\varphi = \psi * \eta$, con $* = \vee, \wedge$ ó $\rightarrow$, entonces teniendo en cuenta los casos 1, 2, 3 y 4 de la definición de asignación de valores verdad y la hipótesis inductiva, se obtiene inmediatamente que $V_{\mathcal{I}}(\varphi(x/t)) = V_{\mathcal{I}}(\varphi(x/t_{\mathcal{I}}))$.

Supongamos que $\varphi = \forall y\, \psi$. Como x es variable libre de φ, debemos tener que $x \neq y$ y que x figura como variable libre en ψ. Luego $\varphi(x/t) = \forall y\, \psi(x/t)$, y se tiene que

(2.14) $$V_{\mathcal{I}}(\varphi(x/t)) = 1 \text{ si y sólo si}$$
$$V_{\mathcal{I}}(\psi(x/t)(y/a)) = 1 \text{ para todo } a \in U_{\mathcal{I}}.$$

La notación $\psi(x/t)(y/a)$ significa que en la fórmula $\psi(x/t)$ hemos sustituído las apariciones libres de y por a. En otras palabras, que hemos sustituído primero las apariciones libres de x en ψ por t, y luego las apariciones libres de y por a. Como $x \neq y$, y t es un término sin variables de L, es claro que podemos cambiar el orden en que efectuamos las sustituciones (y aún efectuarlas simultáneamente). Por lo tanto tenemos que:

$$\psi(x/t)(y/a) = \psi(y/a)(x/t).$$

De esta igualdad y (2.14)obtenemos que

(2.15) $$V_{\mathcal{I}}(\varphi(x/t)) = 1 \text{ si y sólo si}$$
$$V_{\mathcal{I}}(\psi(y/a)(x/t)) = 1 \text{ para todo } a \in U_{\mathcal{I}}.$$

Con argumentos similares podemos ahora probar que

(2.16) $$V_{\mathcal{I}}(\varphi(x/t_{\mathcal{I}})) = 1 \text{ si y sólo si}$$
$$V_{\mathcal{I}}(\psi(y/a))(x/t_{\mathcal{I}})) = 1 \text{ para todo } a \in U_{\mathcal{I}}.$$

Ahora bien, $\psi(y/a)$ es una fórmula de $L(\mathcal{I})$, con x como única variable libre y $comp(\psi(y/a)) < n$. Por la hipótesis inductiva tenemos entonces que:

(2.17) $$V_{\mathcal{I}}(\psi(y/a)(x/t)) = \psi(y/a)(x/t_{\mathcal{I}}), \text{ para todo } a \in U_{\mathcal{I}}.$$

De (2.15), (2.16) y (2.17) se deduce que

$$V_{\mathcal{I}}(\varphi(x/t)) = V_{\mathcal{I}}(\varphi(x/t_{\mathcal{I}})).$$

Para completar la demostración falta considerar el caso en que $\varphi = \exists y\, \psi$. Pero como éste se trata en forma enteramente análoga al del cuantificador universal, lo dejamos al cuidado del lector.

Finalizaremos esta sección introduciendo los conceptos de satisfabilidad, verdad lógica, equivalencia y consecuencia, en forma análoga a lo hecho al estudiar el cálculo proposicional.

Sea φ un enunciado de un lenguaje de primer orden L. Diremos que una interpretación $\mathcal{I}$ *satisface* a φ, o que $\mathcal{I}$ es un *modelo* de φ, si $V_{\mathcal{I}}(\varphi) = 1$.

Diremos que φ es *satisfacible* si tiene al menos un modelo, y que es *insatisfacible* en caso contrario.

Esto es, φ es satisfacible si hay al menos una interpretación $\mathcal{I}$ de L tal que $V_{\mathcal{I}}(\varphi) = 1$, y φ es insatisfacible si se tiene que $V_{\mathcal{I}}(\varphi) = 0$ cualquiera que sea la interpretación $\mathcal{I}$ de L.

Las definiciones anteriores se extienden a conjuntos de enunciados:

Si $\mathbb{H}$ es un conjunto de enunciados de L, diremos que una interpretación $\mathcal{I}$ es un modelo de $\mathbb{H}$, o que $\mathcal{I}$ satisface a $\mathbb{H}$, si $\mathcal{I}$ satisface a todo enunciado de $\mathbb{H}$.

Un conjunto de enunciados $\mathbb{H}$ es *satisfacible* si tiene al menos un modelo, e *insatisfacible* en caso contrario.

Diremos que un enunciado φ es *universalmente válido* o una *verdad lógica* si $V_{\mathcal{I}}(\varphi) = 1$ para toda interpretación $\mathcal{I}$ de L.

La noción de universalmente verdadero es análoga a la noción de tautología en el cálculo proposicional, y se relaciona de la misma manera con la noción de insatisfabilidad. En efecto, se tiene que *un enunciado es universalmente válido si y sólo si su negación es insatisfacible.*

Diremos que dos enunciados φ y ψ son *lógicamente equivalentes*, y escribiremos $\varphi \equiv \psi$, si tienen exactamente los mismos modelos, es decir si $V_{\mathcal{I}}(\varphi) = V_{\mathcal{I}}(\psi)$ cualquiera que sea la interpretación $\mathcal{I}$ de L.

Se verifica inmediatamente que $\varphi \equiv \psi$ *si y sólo si los enunciados* $\varphi \to \psi$ *y* $\psi \to \varphi$ *son ambos universalmente válidos.*

Finalmente, diremos que un enunciado φ es una *consecuencia* de un conjunto $\mathbb{H}$ de enunciados, y escribiremos $\mathbb{H} \models \varphi$ ó $\varphi \in Con(\mathbb{H})$, si todo modelo de $\mathbb{H}$ es también modelo de φ, esto es, si para toda interpretación $\mathcal{I}$ se tiene que $V_{\mathcal{I}}(\psi)) = 1$ para todo $\psi \in \mathbb{H}$ implica que $V_{\mathcal{I}}(\varphi) = 1$.

La relación entre consecuencia y satisfabilidad es la misma que en el caso proposicional:

Proposición 2.3.4 *Sea $\mathbb{H}$ un conjunto de enunciados de un lenguaje de primer orden L. Para todo enunciado φ de L se tiene que $\mathbb{H} \models \varphi$ si y sólo si el conjunto $\mathbb{H} \cup \{\neg\varphi\}$ es insatisfacible.*

Como en el caso proposicional, de lo anterior resulta que *los enunciados universalmente válidos son las consecuencias del conjunto vacío.*

2.4 Arboles de Refutación.

Nos proponemos ahora extender el método de los árboles de refutación del cálculo proposicional al cálculo de predicados, para obtener criterios sintácticos de insatisfabilidad y consecuencia para enunciados de primer orden.

Continuaremos suponiendo que hemos fijado un lenguaje de primer orden L, determinado por el vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$.

Con $\mathbf{R}_{\neg}$, $\mathbf{R}_{\vee}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$, $\mathbf{R}_{\neg\wedge}$, $\mathbf{R}_{\rightarrow}$ y $\mathbf{R}_{\neg\rightarrow}$ indicaremos a los mismos esquemas vistos al estudiar el cálculo proposicional, con la diferencia que las fórmulas proposicionales P y Q que figuran en las premisas y conclusiones de los mismos deben ser reemplazadas por enunciados φ y ψ del lenguaje L.

Teniendo en cuenta los casos 1, 2, 3 y 4 considerados en la definición inductiva de los valores de verdad $V_{\mathcal{I}}$ correspondientes a una interpretación $\mathcal{I}$ de L, se ve que el Lema 1.3.9 sigue valiendo para enunciados de primer orden. Más precisamente, se tiene que:

Lema 2.4.1 *Para toda interpretación $\mathcal{I}$ del lenguaje de primer orden L se tiene que:*

(i) $\mathcal{I}$ satisface la premisa de una de las reglas $\mathbf{R}_{\neg}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$, o $\mathbf{R}_{\neg\rightarrow}$ si y sólo si $\mathcal{I}$ satisface sus conclusiones.

(ii) $\mathcal{I}$ satisface la premisa de una de las reglas $\mathbf{R}_{\vee}$, $\mathbf{R}_{\wedge}$ o $\mathbf{R}_{\rightarrow}$ si y sólo si $\mathcal{I}$ satisface al menos una de sus conclusiones.

Debemos examinar ahora el comportamiento de los cuantificadores con respecto a la satisfacibilidad.

Lema 2.4.2 *Sea φ una fórmula de L con una única variable libre, que denotaremos por x. Para todo conjunto* satisfacible *$\mathbb{H}$ de enunciados del lenguaje L se tiene que:*

1. *Si $\forall x\, \varphi \in \mathbb{H}$, entonces el conjunto que se obtiene agregando a H todos los enunciados de la forma $\varphi(x/t)$, para todo término sin variables t de L, continúa siendo satisfacible.*

2. *Si $\exists x\varphi \in \mathbb{H}$, entonces el conjunto que se obtiene agregando a $\mathbb{H}$ un enunciado de la forma $\varphi(x/\mathbf{c})$, donde $\mathbf{c}$ denota una constante del vocabulario de L que no figure en ninguno de los enunciados de $\mathbb{H}$, continúa siendo satisfacible.*

3. *Si $\neg\forall x\varphi \in \mathbb{H}$, entonces el conjunto que se obtiene agregando a H un enunciado de la forma $\neg\varphi(x/\mathbf{c})$, donde $\mathbf{c}$ denota una constante del vocabulario de L que no figure en ninguno de los enunciados de $\mathbb{H}$, continúa siendo satisfacible.*

4. *Si $\neg\exists x\varphi \in \mathbb{H}$, entonces el conjunto que se obtiene agregando a H todos los enunciados de la forma $\neg\varphi(x/t)$, para todo término sin variables t de L, continúa siendo satisfacible.*

Demostración: A lo largo de esta demostración supondremos que $\mathcal{I}$ es una interpretación del lenguaje L que satisface a todos los enunciados del conjunto $\mathbb{H}$ (tal interpretación existe, puesto que por hipóteis $\mathbb{H}$ es satisfacible).

Si $\forall x\,\varphi \in \mathbb{H}$, entonces $V_{\mathcal{I}}(\forall x\,\varphi) = \top$, y por el caso 5 de la definición de la asignación $V_{\mathcal{I}}$ de valores de verdad, resulta que $V_{\mathcal{I}}(\varphi(x/a)) = \top$ para todo $a \in U_{\mathcal{I}}$. Luego como para todo término sin variables t de L, $t_{\mathcal{I}}$ designa un elemento de $U_{\mathcal{I}}$, se tiene que $\varphi(x/t_{\mathcal{I}}) = \top$, y por el Lema 2.3.3 resulta entonces que $V_{\mathcal{I}}(\varphi(x/t)) = \top$ para todo término sin variables t de L, lo que prueba el item 1.

Si $\exists x\,\varphi \in \mathbb{H}$, entonces $V_{\mathcal{I}}(\exists x\,\varphi) = \top$, y por el caso 6 de la definición de la asignación $V_{\mathcal{I}}$ de valores de verdad, resulta que $V_{\mathcal{I}}(\varphi(x/a)) = \top$ para algún $a \in U_{\mathcal{I}}$. Supongamos que $\mathbf{c} \in \mathcal{C}$ y que $\mathbf{c}$ no figura en ningún enunciado de $\mathbb{H}$. Consideremos una nueva interpretación $\mathcal{J}$ de L definida del siguiente modo:

$$U_{\mathcal{J}} = U_{\mathcal{I}},$$

$$\mathbf{P}_{\mathcal{J}} = \mathbf{P}_{\mathcal{I}}, \text{ para todo } \mathbf{P} \in \mathcal{P},$$

$$\mathbf{f}_{\mathcal{J}} = \mathbf{f}_{\mathcal{I}}, \text{ para todo } \mathbf{f} \in \mathcal{F},$$

$$\mathbf{d}_{\mathcal{J}} = \mathbf{d}_{\mathcal{I}}, \text{ para todo } \mathbf{d} \in \mathcal{F}, \mathbf{d} \neq \mathbf{c}, \text{ y}$$

$$\mathbf{c}_{\mathcal{J}} = a.$$

Como la única diferencia que puede existir entre las interpretaciones $\mathcal{I}$ y $\mathcal{J}$ es en el elemento del universo que designa el símbolo de constante $\mathbf{c}$, por el Lema 2.3.2 tenemos que $\mathcal{J}$ satisface al conjunto $\mathbb{H}$. Además, por

el Lema 2.3.3, $V_{\mathcal{J}}(\varphi(x/\mathbf{c})) = V_{\mathcal{J}}(\varphi(x/a))$ y, otra vez por el Lema 2.3.3, $V_{\mathcal{J}}(\varphi(x/a)) = V_{\mathcal{I}}(\varphi(x/a)) = \top$, lo que completa la demostración del item 2.

Las demostraciones de los items 3 y 4 son análogas a los de los items 2 y 1, respectivamente, por lo que las dejamos al cuidado del lector.

El lema precedente sugiere introducir las siguientes reglas:

$$\mathbf{R}_{\forall} \quad \frac{\forall x\, \varphi}{\varphi(x/t)} \text{ } \textit{donde } t \textit{ es cualquier término sin variables.}$$

$$\mathbf{R}_{\neg\forall} \quad \frac{\neg\forall x\, \varphi}{\neg\varphi(x/\mathbf{c})} \text{ } \textit{donde } \mathbf{c} \textit{ es una constante con restricciones.}$$

$$\mathbf{R}_{\exists} \quad \frac{\exists x\, \varphi}{\varphi(x/\mathbf{c})} \text{ } \textit{donde } \mathbf{c} \textit{ es una constante con restricciones.}$$

$$\mathbf{R}_{\neg\exists} \quad \frac{\neg\exists x\, \varphi}{\neg\varphi(x/t)} \text{ } \textit{donde } t \textit{ es cualquier término sin variables.}$$

El agregado "con restricciones" que figura en las reglas $\mathbf{R}_{\neg\forall}$ y $\mathbf{R}_{\exists}$ recuerda que el símbolo de constante $\mathbf{c}$ no puede ser elegido arbitrariamente.

En los esquemas anteriores, los símbolos de constante del lenguaje L juegan un papel importante, y se vuelven inaplicables si $\mathcal{C} = \emptyset$. Esto nos lleva a considerar una ampliación del lenguaje L por el agregado de nuevos símbolos de constante, que llamaremos *parámetros*. Más precisamente:

Definición 2.4.3 Si $\langle \mathcal{C}, \mathcal{F}, \mathcal{P} \rangle$ es el vocabulario de L, indicaremos con L^{Par} al lenguaje determinado por el vocabulario $\langle \mathcal{C}', \mathcal{F}, \mathcal{P} \rangle$, con

$$\mathcal{C}' = \mathcal{C} \cup \{a_0, a_1, \ldots, a_n, \ldots\},$$

donde $a_0, \ldots a_n \ \ldots$ son símbolos que no figuran en el vocabulario de L. Estos nuevos símbolos de constante se denominan *parámetros*.

Obviamente, todo enunciado φ de L es también un enunciado de L^{Par}, y del Lema 2.3.2 resulta que φ es satisfacible como enunciado de L si y sólo si lo es como enunciado de L^{Par}. Por lo tanto *para considerar problemas de satisfabilidad podemos siempre operar en el lenguaje L^{Par}, que por construcción tiene infinitos símbolos de constante, y volver luego al lenguaje original L.*

Usaremos este principio para definir los árboles de refutación para enunciados de un lenguaje de primer orden.

Definición 2.4.4 Sea $\mathcal{A}$ un árbol cuyos nodos son enunciados del lenguaje L^{Par}. Un árbol $\mathcal{A}'$ se dice *una extensión inmediata de* $\mathcal{A}$ si $\mathcal{A}'$ se obtiene de una de las siguientes maneras, donde τ representa un nodo terminal de $\mathcal{A}$:

1. Si τ tiene como antecesor un enunciado de la forma de la premisa de una de las reglas $\mathbf{R}_{\neg}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$, o $\mathbf{R}_{\neg\rightarrow}$, se agrega una de las conclusiones de esa regla como sucesor inmediato de τ.

2. Si τ tiene como antecesor un enunciado de la forma de la premisa de una de las reglas $\mathbf{R}_{\vee}$, $\mathbf{R}_{\wedge}$ o $\mathbf{R}_{\rightarrow}$, se agregan las dos conclusiones de esta regla como dos sucesores inmediatos de τ.

3. Si τ tiene como antecesor un enunciado de la forma $\forall x\,\varphi$, se agrega como sucesor inmediato de τ un enunciado de la forma $\varphi(x/t)$, donde t es un término sin variables del lenguaje L^{Par}.

4. Si τ tiene como antecesor un enunciado de la forma $\neg\forall x\,\varphi$, se agrega como sucesor inmediato de τ un enunciado de la forma $\neg\varphi(x/a)$, donde a es un parámetro que no figura en ninguno de los enunciados de $\mathcal{A}$.

5. Si τ tiene como antecesor un enunciado de la forma $\exists x\,\varphi$, se agrega como sucesor inmediato de τ un enunciado de la forma $\varphi(x/a)$, donde a es un parámetro que no figura en ninguno de los enunciados de $\mathcal{A}$.

6. Si τ tiene como antecesor un enunciado de la forma $\neg\exists x\,\varphi$, se agrega como sucesor inmediato de τ un enunciado de la forma $\neg\varphi(x/t)$, donde t es un término sin variables del lenguaje L^{Par}.

En los casos 3, 4, 5 y 6 de la definición anterior, se dice que el árbol $\mathcal{A}$ se ha extendido por aplicación de las reglas $\mathbf{R}_{\forall}$, $\mathbf{R}_{\neg\forall}$, $\mathbf{R}_{\exists}$ y $\mathbf{R}_{\neg\exists}$, respectivamente.

Definición 2.4.5 Un árbol de enunciados del lenguaje L^{Par} es *un árbol de refutación* de un enunciado φ del lenguaje L si y sólo si se lo obtiene haciendo un número finito de extensiones inmediatas a partir del árbol cuyo único nodo es el enunciado φ.Más precisamente, $\mathcal{A}$ es un árbol de refutación de φ si y sólo si existe una sucesión finita $\mathcal{A}_0, \mathcal{A}_1, \ldots \mathcal{A}_n$ de árboles de enunciados de L^{Par} que satisface las siguientes condiciones:

1. $\mathcal{A}_0$ tiene por único nodo al enunciado φ,

2. $\mathcal{A}_i$ es una extensión inmediata de $\mathcal{A}_{i-1}$, para $i = 1, \ldots, n$, y

3. $\mathcal{A} = \mathcal{A}_n$.

Un *árbol de refutación de un conjunto finito* $\mathbb{S}$ de enunciados de L es un árbol de refutación de la conjunción de los enunciados de $\mathbb{S}$.

Los siguientes conceptos son análogos a los vistos al estudiar árboles de refutación para el cálculo proposicional:

Una rama de un árbol de enunciados se dice *cerrada* si contiene a la vez a un enunciado y a su negación. En caso contrario, la rama se dice *abierta*.

Un árbol se dice *cerrado* si todas sus ramas son cerradas.

Teorema 2.4.6 *Sea* $\mathbb{S} = \{\varphi_1, \ldots, \varphi_n\}$ *un conjunto* finito y no vacío *de enunciados de un lenguaje de primer orden* L. *Si* $\mathbb{S}$ *origina un árbol de refutación cerrado, entonces* $\mathbb{S}$ *es insatisfacible.*

Demostración: Los Lemas 2.4.1 y 2.4.2 junto con las Definiciones 2.4.4 y 2.4.5, nos permiten utilizar argumentos similares a los usados en la demostración del Lema 1.3.17 para concluir que si $\mathbb{S}$ fuese satisfacible, entonces todo árbol de refutación de $\mathbb{S}$ debería tener al menos una rama satisfacible, y, por lo tanto, abierta. Por consiguiente, si $\mathbb{S}$ tiene un árbol de refutación cerrado, entonces $\mathbb{S}$ no puede ser satisfacible.

Corolario 2.4.7 *Sea* φ *un enunciado de un lenguaje de primer orden* L. *Si* $\neg\varphi$ *origina un árbol de refutación cerrado, entonces* φ *es universalmente válido.*

Corolario 2.4.8 *Sean* $\varphi_1, \ldots, \varphi_n, \psi$ *enunciados de un lenguaje de primer orden* L. *Si el conjunto* $\{\varphi_1, \ldots, \varphi_n, \neg\psi\}$ *origina un árbol de refutación cerrado, entonces* ψ *es una consecuencia del conjunto* $\{\varphi_1, \ldots, \varphi_n\}$.

El Teorema 2.4.6 y sus corolarios nos dan los criterios sintácticos para satisfabilidad y consecuencia que mencionamos al iniciar esta sección.

2.5 Existencia de modelos

Ahora nos proponemos probar la recíproca del Teorema 2.4.6: si todo árbol de refutación de un enunciado φ tiene una rama abierta, entonces φ es satisfacible.

Imitando lo hecho en el caso del cálculo proposicional, comenzaremos definiendo conjuntos saturados de enunciados.

Definición 2.5.1 Sea L un lenguaje de primer orden con al menos un símbolo de constante. Un conjunto $\mathbb{H}$ de enunciados de L se dice *saturado* o un *conjunto de Hintikka* si y sólo si satisface las siguientes condiciones:

(S0) Un enunciado y su negación no pueden pertenecer simultáneamente a $\mathbb{H}$.

(S1) Si un enunciado de la forma de la premisa de una de las reglas $\mathbf{R}_{\neg}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$, o $\mathbf{R}_{\neg\rightarrow}$ pertenece a $\mathbb{H}$, entonces sus conclusiones también pertenecen a $\mathbb{H}$.

(S2) Si un un enunciado de la forma de la premisa de una de las reglas $\mathbf{R}_{\vee}$, $\mathbf{R}_{\wedge}$ o $\mathbf{R}_{\rightarrow}$ pertenece a $\mathbb{H}$, entonces al menos una de sus conclusiones pertenece a $\mathbb{H}$.

(S3) Si un enunciado de la forma $\forall x\,\varphi$ pertenece a $\mathbb{H}$, entonces para todo término sin variables del lenguaje L, $\varphi(x/t) \in \mathbb{H}$.

(S4) Si un enunciado de la forma $\neg\forall x\,\varphi$ pertenece a $\mathbb{H}$, entonces $\neg\varphi(x/\mathbf{c}) \in \mathbb{H}$, para algún símbolo de constante $\mathbf{c}$ del lenguaje L.

(S5) Si un enunciado de la forma $\exists x\,\varphi$ está en $\mathbb{H}$, entonces $\varphi(x/\mathbf{c}) \in \mathbb{H}$, para algún símbolo de constante $\mathbf{c}$ del lenguaje L.

(S6) Si un enunciado de la forma $\neg\exists x\,\varphi$ pertenece a $\mathbb{H}$, entonces para todo término sin variables del lenguaje L, $\neg\varphi(x/t) \in \mathbb{H}$.

Teorema 2.5.2 *Sea L un lenguaje de primer orden determinado por el vocabulario $\langle \mathcal{C}, \mathcal{F}, \mathcal{P}\rangle$, con $\mathcal{C} \neq \emptyset$. Entonces todo conjunto saturado de enunciados de L es satisfacible.*

Demostración: Sea $\mathbb{H}$ un conjunto saturado de enunciados de L. Vamos a definir una interpretación $\mathcal{I}$ de L del modo siguiente:

- El universo $U_{\mathcal{I}}$ de $\mathcal{I}$ es el conjunto de los términos sin variables de L.
- Para cada $\mathbf{c} \in \mathcal{C}$, $\mathbf{c}_{\mathcal{I}} = \mathbf{c}$.
- Sea $\mathbf{f} \in \mathcal{F}_k$. Como para cada k-upla $(t_1, \ldots, t_k)$ de elementos de $U_{\mathcal{I}}$, $\mathbf{f}(t_1 \ldots t_k) \in U_{\mathcal{I}}$, interpretamos $\mathbf{f}$ por la función $f_{\mathcal{I}}\colon U_{\mathcal{I}}^k \rightarrow U_{\mathcal{I}}$ definida por:

$$f_{\mathcal{I}}(t_1, \ldots, t_k) = \mathbf{f}(t_1 \ldots t_k), \text{ para toda } (t_1, \ldots, t_k) \in U_{\mathcal{I}}^k.$$

- Sea $\mathbf{P} \in \mathcal{P}_k$. Entonces interpretamos $\mathbf{P}$ por la relación k-aria $P_{\mathcal{I}}$ sobre $U_{\mathcal{I}}$ definida por:

$$P_{\mathcal{I}} = \{(t_1, \ldots, t_k) \in U_{\mathcal{I}}^k \mid \mathbf{P}(t_1 \ldots t_k) \in \mathbb{H}\}.$$

Es muy fácil verificar, por inducción en la complejidad, que:

(2.18) Para todo término sin variables t de L, $t_{\mathcal{I}} = t$.

Probaremos ahora, por inducción en la complejidad, que para todo enunciado $\varphi \in \mathbb{H}$, se tiene que $V_{\mathcal{I}}(\varphi) = \top$.

Supongamos primero que φ es un enunciado atómico, digamos $\varphi = \mathbf{P}(t_1 \ldots t_k)$, con $\mathbf{P} \in \mathcal{P}_k$ y $t_1, \ldots, t_k$ términos sin variables de L. Entonces $V_{\mathcal{I}}(\varphi) = \top$ si y sólo si $(t_1, \ldots, t_k) \in P_{\mathcal{I}}$. Pero por la definición de la interpretación $\mathcal{I}$, $(t_1, \ldots, t_k) \in P_{\mathcal{I}}$ si y sólo si $\varphi = \mathbf{P}(t_1 \ldots t_k) \in \mathbb{H}$. Esto prueba que $V_{\mathcal{I}}(\varphi) = \top$ para todo enunciado atómico $\varphi \in \mathbb{H}$.

Sea $n > 0$ y supongamos que hemos probado que $\mathcal{I}$ satisface a todos los enunciados de $\mathbb{H}$ de complejidad menor que n. Sea $\varphi \in \mathbb{H}$, y $comp(\varphi) = n$.

Si $\varphi = (\psi \vee \eta)$, entonces por la propiedad (S2) de los conjuntos saturados se tiene que $\psi \in \mathbb{S}$ ó $\eta \in \mathbb{S}$. Podemos suponer, sin perder generalidad, que $\psi \in \mathbb{S}$. Como $comp(\psi) < comp(\varphi)$, por la hipótesis inductiva tenemos que $V_{\mathcal{I}}(\psi) = \top$, de donde resulta que también $V_{\mathcal{I}}(\varphi) = \top$.

Los casos $\varphi = (\psi \wedge \eta)$ y $\varphi = (\psi \to \eta)$ se tratan de manera análoga.

Si $\varphi = \forall x\, \psi$, entonces por (S3), $\psi(x/t) \in \mathbb{S}$ para todo término sin variables t de L. Luego, la hipótesis inductiva, $V_{\mathcal{I}}(\psi(x/t) = \top$ para todo $t \in U_{\mathcal{I}}$. Esto significa que $V_{\mathcal{I}}(\varphi) = \top$.

Si $\varphi = \exists x\, \psi$, entonces por (S4) hay una constante $\mathbf{c}$ de L tal que $\psi(x/\mathbf{c}) \in \mathbb{S}$. Por la hipótesis inductiva, $V_{\mathcal{I}}(\psi(x/\mathbf{c}) = \top$, y como $\mathbf{c} \in U_{\mathcal{I}}$, resulta que $V_{\mathcal{I}}(\varphi) = \top$.

Queda por considerar el caso en que $\varphi = \neg\psi$. Procedamos por inducción en la complejidad de ψ.

Si $comp(\psi) = 0$, entonces $\psi = \mathbf{P}(t_1 \ldots t_k)$, para algún $k \geq 1$, algún $\mathbf{P} \in \mathcal{P}_k$ y términos sin variables $t_1, \ldots, t_k$. Como $\varphi = \neg\psi \in \mathbb{S}$, por (S0) resulta que $\mathbf{P}(t_1 \ldots t_k) \notin \mathbb{S}$. Luego, por la definición de la relación $P_{\mathcal{I}}$, se tiene que $(t_1, \ldots, t_k) \notin P_{\mathcal{I}}$. Por lo tanto $V_{\mathcal{I}}(\psi) = \bot$ y $V_{\mathcal{I}}(\varphi) = \top$.

Los distintos casos que pueden presentarse cuando $comp(\psi) > 0$, se tratan utilizando las propiedades (S1), (S2), (S4)y (S6), en forma enteramente análoga a los casos analizados anteriormente, por lo que los dejamos a cargo del lector.

El universo de la interpretación $\mathcal{I}$ considerada en la demostración del teorema anterior, esto es, el conjunto formado por los términos sin variables, se denomina *el universo de Herbrand del lenguaje* L.

Del teorema anterior resulta que si un enunciado φ origina un árbol de refutación con una rama saturada, entonces φ es satisfacible, y además podemos construir un modelo de φ a partir de dicha rama.

Luego, por analogía con lo hecho al considerar el cálculo proposicional, diremos que un árbol de refutación de un enunciado de primer orden es *completo* cuando es cerrado o tiene una rama saturada.

Pero ahora se nos presenta una dificultad esencial: hay enunciados de primer orden que son satisfacibles pero que no admiten modelos finitos (es decir, interpretaciones con universos finitos), como lo muestra el siguiente:

Ejemplo 2.5.3 Sea L un lenguaje de primer orden determinado por el vocabulario $\mathcal{C} = \mathcal{F} = \emptyset$ y $\mathcal{P} = \mathcal{P}_2 = \{\mathbf{P}\}$. Sea φ la conjunción de los enunciados

$$\forall v_0\, \neg \mathbf{P}(v_0, v_0),$$

$$\forall v_0 \forall v_1 \forall v_2 ((\mathbf{P}(v_0, v_1) \wedge \mathbf{P}(v_1, v_2)) \rightarrow \mathbf{P}(v_0, v_2)), \text{ y}$$

$$\forall v_0 \exists v_1 \mathbf{P}(v_0, v_1).$$

El enunciado φ es satisfacible: para obtener un modelo, basta interpretar el símbolo de predicado binario $\mathbf{P}$ como la relación de mayor estricto sobre los números naturales. Pero es fácil verificar que ninguna relación binaria sobre un conjunto finito (y no vacío) puede satisfacer simultáneamente las tres condiciones que definen φ.

Luego, teniendo en cuenta el Teorema 2.4.6, podemos concluir que *todo árbol de refutación originado en* φ *tiene ramas abiertas, pero estas ramas no pueden ser saturadas.*

Este ejemplo muestra que, a diferencia de lo visto en el cálculo proposicional, *en general no podremos completar el árbol de refutación de un enunciado satisfacible en un nmero finito de pasos.*

Luego puede ocurrir que tengamos una sucesión *infinita*

$$\mathcal{A}_0, \mathcal{A}_1, \ldots, \mathcal{A}_n, \ldots$$

de árboles con ramas abiertas, de modo que $\mathcal{A}_0$ es el árbol que tiene por único nodo a φ, y cada $\mathcal{A}_{n+1}$ se obtiene extendiendo las ramas abiertas de

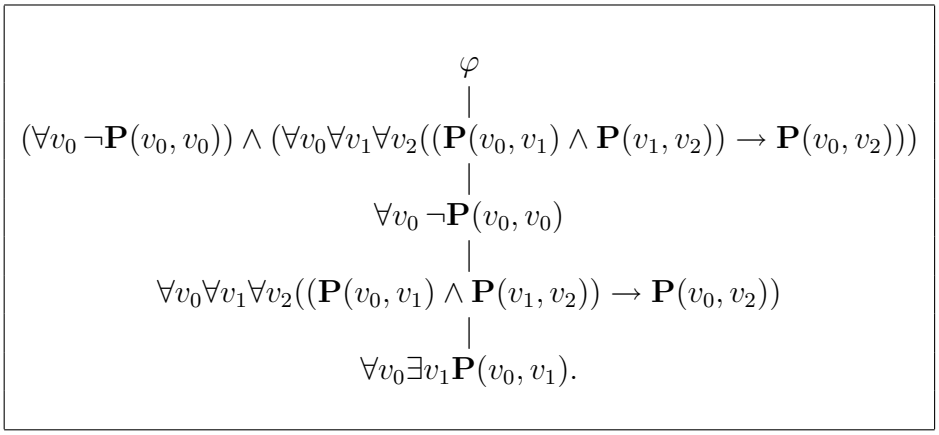

Figure 2.1: $\mathcal{A}_4$

$\mathcal{A}_n$ por aplicación de las reglas de formación de árboles un número finito de veces. Una sucesión con estas propiedades se dirá *una sucesión encajada* de árboles de refutación de φ.

Podemos considerar entonces el árbol infinito $\mathcal{A}_\infty = \bigcup_{n\in\mathbf{N}} \mathcal{A}_n$. Por el Lema de König (Lema 1.4.2), $\mathcal{A}_\infty$ debe tener una rama infinita, digamos $\mathcal{R}$. Pero, a diferencia de lo visto en la demostración del Teorema 1.3.24, *ahora no podemos asegurar que los enunciados que figuran en $\mathcal{R}$ formen un conjunto saturado.*

Ejemplo 2.5.4 Sea φ como en el Ejemplo 2.5.3. Por aplicación de la regla $\mathbf{R}_\wedge$ obtenemos el árbol $\mathcal{A}_4$, indicado en la Figura 2.1.

Este árbol $\mathcal{A}_4$ puede ser continuado indefinidamente aplicando alternadamente las reglas $\mathbf{R}_\forall$ y $\mathbf{R}_\exists$ a su nodo terminal

$$\forall v_0 \exists v_1 \mathbf{P}(v_0, v_1).$$

Obtenemos asíla sucesión infinita de árboles $\mathcal{A}_{4+2k}$, indicados en la Figura 2.2, $k = 1, 2, \ldots$.

El árbol infinito $\mathcal{A}_\infty = \bigcup_{k\geq 1} \mathcal{A}_{4+2k}$ tiene una única rama infinita, que no constituye un conjunto saturado de enunciados: por ejemplo, $\forall v_0\, \neg\mathbf{P}(v_0, v_0)$ figura en la rama, pero no figura $\neg\mathbf{P}(a_0, a_0)$. Luego no se cumple la propiedad (S3).

Como las ramas infinitas no son necesariamente completas, también se puede dar el caso de que un enunciado insatisfacible genere una sucesión

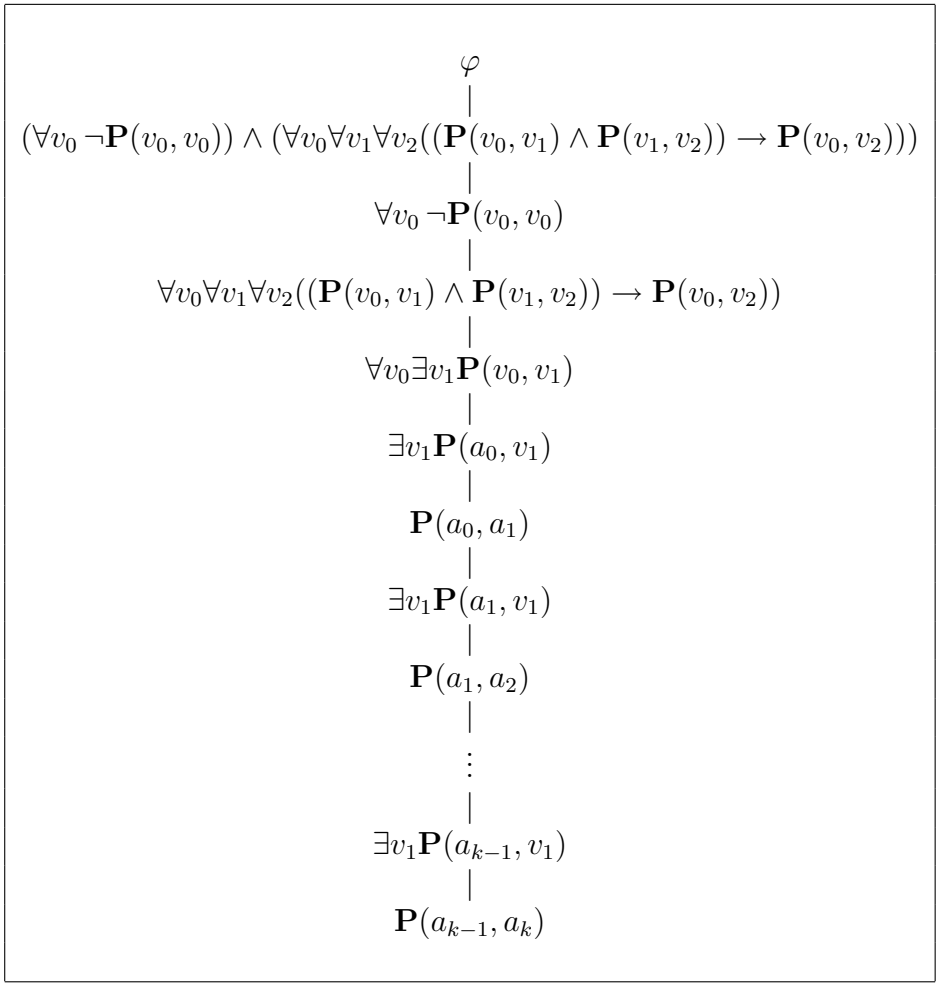

Figure 2.2: $\mathcal{A}_{4+2k}$

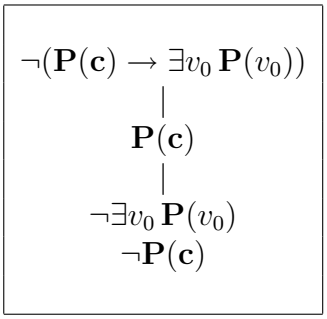

Figure 2.3: Árbol cerrado de $\neg(\mathbf{P}(\mathbf{c}) \to \exists v_0\, \mathbf{P}(v_0))$

encajada de árboles de refutación. En otras palabras, *un enunciado insatisfacible puede originar un árbol que no se cierra en un número finito de pasos.*

Ejemplo 2.5.5 Sea L el lenguaje de primer orden determinado por el vocabulario $\mathcal{C} = \{\mathbf{c}\}$, $\mathcal{F} = \mathcal{F}_1 = \{\mathbf{f}\}$ y $\mathcal{P} = \mathcal{P}_1 = \{\mathbf{P}\}$. El enunciado

$$\psi = \neg(\mathbf{P}(\mathbf{c}) \to \exists v_0\, \mathbf{P}(v_0))$$

es insatisfacible. En efecto, la Figura 2.3 muestra un árbol de refutación cerrado originado en ψ. Por otro lado, la Figura 2.4 muestra el elemento genérico $\mathcal{B}_{2+k}$ de una sucesión encajada de árboles de refutación de ψ.

Los ejemplos anteriores muestran que la existencia de una sucesión encajada de árboles de refutación de un enunciado φ no nos permite concluir nada sobre la satisfabilidad de φ. Del examen de esos ejemplos surge también que esta situación se debe a que las reglas $\mathbf{R}_\forall$ y $\mathbf{R}_{\neg\exists}$ se pueden aplicar de una manera *correcta pero parcial*, produciendo conjuntos infinitos no saturados.

Para subsanar esta dificultad debemos diseñar un procedimiento que nos asegure que la aplicación de las reglas se hace teniendo en cuenta todas las posibilidades, de modo que o bien se obtenga un árbol cerrado (en un número finito de pasos) o bien una sucesión encajada que determine un conjunto saturado. Hay varios procedimientos posibles. A continuación describiremos uno de ellos.

Sea φ un enunciado de un lenguaje de primer orden. Entonces el vocabulario del sublenguaje L_φ (ver página 63) tiene un número finito, digamos $m > 0$, de símbolos. Podemos hacer entonces una lista com $m + 8$ símbolos,

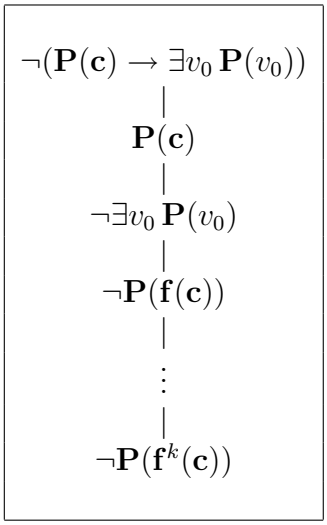

Figure 2.4: $\mathcal{B}_{2+k}$

comenzando por los conectivos, cuantificadores y paréntesis:

$$\vee, \wedge, \rightarrow, \neg, \forall, \exists, (,)$$

y colocando a continuación los símbolos que hubiere en $\mathcal{C}_\varphi$, luego los de $\mathcal{F}_\varphi$ y finalmente los de $\mathcal{P}_\varphi$. El orden en que colocamos los elementos del vocabulario de L_φ es arbitrario, pero supondremos que hemos elegido uno, que llamaremos el *orden alfabético*. De este modo, a cada uno de estos símbolos le corresponde un *número de orden*, entre 9 y m.

Podemos ahora extender nuestra lista, agregando alternativamente variables y parámetros:

$$v_0, a_0, v_1, a_1, \ldots$$

A la variable v_0 le corresponderásí el número de orden $8+m+1$, al parámetro a_0, $8+m+2$, y en general, a v_n le asignaremos el número $8+m+2n+1$, y a a_n el número $8+m+2n+2$.

Podemos ahora asignar un número de orden a cada uno de los términos sin variables del lenguaje L_φ.

Para cada $n > 0$, sea $\mathbb{D}_n$ el conjunto de todos los enunciados del lenguaje L_φ^{Par} que se escriben con a lo sumo n símbolos con número de orden a lo sumo n (cada variable v_k y cada parámetro a_k se cuenta como un único símbolo).

Cada conjunto $\mathbb{D}_n$ es finito, y si $\mathbb{D}_n \neq \emptyset$, entonces puede ser totalmente ordenado por el siguiente criterio: si el número de símbolos de ψ es menor

que el de η, entonces ψ antecede a η. Si tienen el mismo número de símbolos, entonces usamos el orden alfabético.

Ahora podemos listar todos los enunciados de L_φ^{Par}, escribiendo *todos* los enunciados que figuran en $\mathbb{D}_{n+1}$ a continuación de los de $\mathbb{D}_n$. Obtenemos así una sucesión

$$\psi_0, \psi_1, \ldots, \psi_k, \psi_{k+1}, \ldots \tag{2.19}$$

donde ψ_0 es el primer elemento del primer conjunto $\mathbb{D}_n \neq \emptyset$.

Como obviamente $\mathbb{D}_n \subseteq \mathbb{D}_{n+1}$, obtenemos que en esta sucesión, *cada enunciado de L_φ^{Par} aparece repetido infinitas veces*. Esta propiedad es esencial para nuestro propósito.

Procediendo en forma similar, podemos obtener una sucesión

$$t_0, t_1, \ldots, t_k, t_{k+1}, \ldots \tag{2.20}$$

donde figuren *todos los términos sin variables de L_φ^{Par}, repetidos infinitas veces* (aunque esta repetición infinita no tendrá especial interés).

Vamos a describir ahora un procedimiento inductivo para obtener una sucesión

$$\mathcal{E}_0, \mathcal{E}_1, \ldots, \mathcal{E}_n, \ldots$$

de árboles de refutación de φ, que llamaremos *sucesión especial*.

$\mathcal{E}_0$ es el árbol que tiene por único nodo al enunciado φ.

Supongamos ahora que hemos construído el árbol $\mathcal{E}_n$.

Sea ψ_n el n-ésimo enunciado de la sucesión (2.19). Si ψ_n no figura en ninguna rama abierta de $\mathcal{E}_n$, entonces ponemos $\mathcal{E}_{n+1} = \mathcal{E}_n$. En caso contrario, se puede presentar uno, y sólo uno, de los casos siguientes:

1. ψ_n es una fórmula atómica o negación de una fórmula atómica.
2. ψ_n es de la forma de la premisa de unas de las reglas $\mathbf{R}_\neg$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_\wedge$, ó $\mathbf{R}_{\neg\rightarrow}$.
3. ψ_n es de la forma de la premisa de unas de las reglas $\mathbf{R}_\vee$, $\mathbf{R}_{\neg\wedge}$, ó $\mathbf{R}_\rightarrow$.
4. ψ_n es de la forma $\forall x\, \eta$.
5. ψ_n es de la forma $\neg\forall x\, \eta$.
6. ψ_n es de la forma $\exists x\, \eta$.
7. ψ_n es de la forma $\neg\exists x\, \eta$.

En el caso 1, ponemos $\mathcal{E}_{n+1} = \mathcal{E}_n$.

En el caso 2, $\mathcal{E}_{n+1}$ es el árbol que se obtiene agregando a todas las ramas abiertas de $\mathcal{E}_n$ donde figure ψ_n su conclusión (en el caso de $\mathbf{R}_\neg$), o sucesivamente sus dos conclusiones (en los casos $\mathbf{R}_\wedge$ y $\mathbf{R}_{\neg\to}$).

En el caso 3, $\mathcal{E}_{n+1}$ es el árbol que se obtiene agregando a todas las ramas abiertas de $\mathcal{E}_n$ donde figure ψ_n sus conclusiones, como dos nuevos nodos terminales.

En el caso 4, $\mathcal{E}_{n+1}$ es el árbol que se obtiene agregando a todas las ramas abiertas de $\mathcal{E}_n$ donde figure ψ_n sucesivamente los enunciados $\eta(x/t_0), \eta(x/t_1), \ldots, \eta(x/t_n)$, donde t_i es el i-ésimo término de la sucesión (2.20).

En el caso 5, $\mathcal{E}_{n+1}$ es el árbol que se obtiene agregando a todas las ramas abiertas de $\mathcal{E}_n$ donde figure ψ_n el enunciado $\neg\eta(x/a_r)$, donde a_r es el primer parámetro que no figura en $\mathcal{E}_n$.

En el caso 6, $\mathcal{E}_{n+1}$ es el árbol que se obtiene agregando a todas las ramas abiertas de $\mathcal{E}_n$ donde figure ψ_n el enunciado $\eta(x/a_r)$, donde a_r es el primer parámetro que no figura en $\mathcal{E}_n$.

Finalmente, en el caso 7, $\mathcal{E}_{n+1}$ es el árbol que se obtiene agregando a todas las ramas abiertas de $\mathcal{E}_n$ donde figure ψ_n sucesivamente los enunciados $\neg\eta(x/t_0), \neg\eta(x/t_1), \ldots, \neg\eta(x/t_n)$, donde t_i es el i-ésimo término de la sucesión (2.20).

Esto completa la definición de $\mathcal{E}_{n+1}$.

Observemos que si $\mathcal{E}_n$ es cerrado, entonces $\mathcal{E}_m = \mathcal{E}_n$ para todo $m \geq n$. Pero la recíproca no es verdadera, como lo muestra el siguiente:

Ejemplo 2.5.6 Sea L el lenguaje de primer orden determinado por el vocabulario $\mathcal{C} = \{\mathbf{c}\}$, $\mathcal{F} = \emptyset$ y $\mathcal{P} = \mathcal{P}_1 = \{\mathbf{P}\}$, y sea φ el enunciado atómico $\mathbf{P}(\mathbf{c})$. Se verifica fácilmente que $\mathcal{E}_n = \mathcal{E}_0$ para todo árbol especial $\mathcal{E}_n$ originado en φ.

El siguiente lema justifica la construcción de las sucesiones especiales de árboles de refutación.

Lema 2.5.7 *Sea φ un enunciado de un lenguaje de primer orden. Si todos los árboles especiales $\mathcal{E}_n$ originados en φ tienen ramas abiertas, entonces φ es satisfacible.*

Demostración: Supongamos que para todo $n \geq 0$, $\mathcal{E}_n$ tiene al menos una rama abierta. Observemos que por la forma en que están definidos estos árboles, un mismo enunciado puede aparecer repetido en una rama $\mathcal{R}_n$ de

$\mathcal{E}_n$. El *conjunto* formado por los enunciados que figuran en $\mathcal{R}_n$ será denotado por $\mathbb{R}_n$. Sea $m > n$. Diremos que una rama $\mathcal{R}_m$ de $\mathcal{E}_m$ es una *extensión* de la rama $\mathcal{R}_n$ de $\mathcal{E}_n$ si y sólo si $\mathbb{R}_m \supseteq \mathbb{R}_n$.

A continuación probaremos la siguiente afirmación:

(A) *Existe una sucesión $\mathcal{R}_n$ tal que $\mathcal{R}_n$ es una rama abierta de $\mathcal{E}_n$ y $\mathcal{R}_{n+1}$ es una extensión de $\mathcal{R}_n$, para todo $n \geq 0$.*

Diremos que una rama abierta $\mathcal{R}_n$ de $\mathcal{E}_n$ es *buena* si y sólo si para infinitos $m > n$, $\mathcal{E}_m$ contiene ramas abiertas que extienden a $\mathcal{R}_n$.

Cada rama abierta $\mathcal{R}_n$ de $\mathcal{E}_n$ tiene un número finito de extensiones en $\mathcal{E}_{n+1}$, digamos $\mathcal{R}^1_{n+1}, \ldots, \mathcal{R}^k_{n+1}$, con $k \geq 1$. Si $m > n$, entonces toda rama de E_m que sea extensión de $\mathcal{R}_n$ deberá ser también extensión de por lo menos una de las ramas $\mathcal{R}^1_{n+1}, \ldots, \mathcal{R}^k_{n+1}$. Luego, si $\mathcal{R}_n$ es buena, también lo será una de sus extensiones a $\mathcal{E}_{n+1}$.

Como todas las ramas abiertas de cualquier $\mathcal{E}_n$ son extensiones de la única rama $\mathcal{R}_0$ de $\mathcal{E}_0$, resulta que $\mathcal{R}_0$ es buena. Como las ramas abiertas de $\mathcal{E}_1$ extienden a $\mathcal{R}_0$, podemos elegir una que sea buena, y que llamaremos $\mathcal{R}_1$. Entre las ramas abiertas de $\mathcal{E}_2$ que extienden a $\mathcal{R}_1$, elijamos una buena, que llamaremos $\mathcal{R}_2$. En general, habiendo formado la sucesión $\mathcal{R}_0, \mathcal{R}_1, \ldots \mathcal{R}_n$ de ramas abiertas buenas de $\mathcal{E}_0, \mathcal{E}_1, \ldots \mathcal{E}_n$, tales que $\mathbb{R}_0 \subseteq \mathcal{R}_1 \subseteq \ldots \subseteq \mathcal{R}_n$, definimos $\mathcal{R}_{n+1}$ como una rama abierta buena de $\mathcal{E}_{n+1}$ que extienda a $\mathcal{R}_n$. Esto prueba (A).

Veamos ahora que si $\mathcal{R}_n$ es una sucesión de ramas abiertas satisfaciendo las condiciones de (A), entonces $\mathbb{R} = \bigcup_{n \in \mathbf{N}} \mathbb{R}_n$ es un conjunto saturado de enunciados de L^{Par}_{φ}.

Sea $\eta \in \mathbb{R}$. Entonces $\eta \in \mathbb{R}_n$ para algún $n \geq 0$. Por otra parte, existe $m \geq n$ tal que $\eta = \psi_m$, el m-ésimo enunciado de la lista (2.19). Luego si η es de la forma de la premisa de una de las reglas $\mathbf{R}_{\neg}$, $\mathbf{R}_{\neg\vee}$, $\mathbf{R}_{\wedge}$ ó $\mathbf{R}_{\neg\rightarrow}$, entonces todas sus conclusiones están en $\mathbb{R}_{m+1}$. Si η es de la forma de la premisa de alguna de las reglas $\mathbf{R}_{\vee}$, $\mathbf{R}_{\neg\wedge}$ ó $\mathbf{R}_{\neg\rightarrow}$, entonces al menos una de sus conclusiones está en $\mathbb{R}_{m+1}$. Si η es de la forma $\neg\forall x\, \chi$ ó $\exists x\, \chi$, entonces $\neg\chi(x/a)$ ó $\chi(x/a)$ están en $\mathcal{R}_{m+1}$, donde a es algún parámetro (esto es, una constante del lenguaje L^{Par}_{φ}). Finalmente, supongamos que η es de la forma $\forall x\, \chi$ ó $\neg\exists x\, \chi$, y sea t un término sin variables del lenguaje L^{Par}_{φ}. Tendremos que $t = t_p$, donde t_p es el p-ésimo término de la lista (2.20). Si $p \leq m$, entonces $\chi(x/t)$ ó $\neg\chi(x/t)$ está en $\mathbb{R}_{m+1}$. Si $p > m$, existe $q > p$ tal que $\eta = \psi_q$, y resulta que $\chi(x/t)$ ó $\neg\chi(x/t)$ está en $\mathbb{R}_{q+1}$. Vemos así que $\mathbb{R}$ es saturado.

Por el Teorema 2.5.2, resulta entonces que $\mathbb{R}$ es satisfacible, y como $\varphi \in \mathbb{R}$, obtenemos que φ es satisfacible, como querímos demostrar.

Observemos que en el teorema anterior no se descarta la posibilidad de que exista un n tal $\mathcal{E}_m = \mathcal{E}_n$ para todo $m \geq n$. En este caso $\mathbb{R} = \bigcup_{i=0}^{i=n} \mathbb{R}_i$ es un conjunto saturado finito, y por lo tanto φ tiene un modelo finito.

El siguiente teorema es una consecuencia inmediata del Teorema 2.4.6 y del Lema 2.5.7.

Teorema 2.5.8 *Un enunciado φ de un lenguaje de primer orden L es insatisfacible si y sólo si φ origina un árbol de refutación cerrado.*

Corolario 2.5.9 *Sea L un lenguaje de primer orden. Un enunciado φ es consecuencia de los enunciados $\psi_1, \ldots, \psi_k$ si y sólo si el enunciado*

$$\psi_1 \wedge \cdots \wedge \psi_k \wedge \neg\varphi$$

origina un árbol de refutación cerrado.