

Elementos de lógica y calculabilidad

Xavier Caicedo F.

Favor no escribir ni subrayar
los libros y revistas Gracias
Sistema de Bibliotecas
Universidad de los Andes

Departamento de Matemáticas
Universidad de los Andes

Bogotá, julio de 1989. Edición revisada por el autor.
Departamento de Matemáticas
Universidad de los Andes

Diagramación:  una empresa docente®

Impresión: Copilito

Ningún de esta publicación, puede ser reproducida, almacenada o transmitida de manera alguna ni por ningún medio,
sin permiso de una empresa docente y del autor

PA
512.3
6132
1990
E-2

Contenido

Introducción, ix

Primera parte

El cálculo de proposiciones

Capítulo 1		
Sintaxis, 3	1.1 Simbolización	3
	1.2 Fórmulas bien formadas, inducción en fórmulas	8
	1.3 Descomposición única, conectivo principal	12
	1.4 El árbol de una fórmula, definiciones recursivas	16
	1.5 Supresión de paréntesis	20
	1.6 Notación polaca	21
	1.7 Decidibilidad	28

Capítulo 2		
Deducción formal, 35	2.1 Cálculo Proposicional Axiomático	37
	2.2 Deducción con premisas, reglas derivadas	40
	2.3 Teorema de la deducción	44
	2.4 Reducción al absurdo	48
Capítulo 3		
Semántica, 55	3.1 Valuaciones, validez	56
	3.2 Completitud	62
Capítulo 4		
Equivalencia, 71	4.1 Algebra Proposicional	71
	4.2 Conjuntos completos de conectivos	76
	4.3 Formas normales, funciones booleanas	80
Capítulo 5		
Resolución en el cálculo de proposiciones, 89		
Capítulo 6		
Compacidad, Conjuntos infinitos de premisas, 95	6.1 Lema de Konig	96
	6.2 Compacidad	100

Segunda parte

Cálculo de predicados

Capítulo 1

Simbolización, 107	1.1 Predicados, relaciones, cuantificadores	108
	1.2 Igualdad, funciones	116
	1.3 Conceptos primitivos y definidos	121
	1.4 Sintaxis	126

Capítulo 2

Semántica, 135	2.1 Estructuras	135
	2.2 Validez	143
	2.3 Isomorfismo	149

Capítulo 3

Deducción Formal, 155	3.1 Axiomatización del Cálculo de Predicados	155
	3.2 Validez del sistema axiomático	164
	3.3 Especificación existencial	169
	3.4 Equivalencia, Forma Prenexa	173

Capítulo 4

Resolución en el cálculo de predicados, 181	4.1 Forma Normal de Skolem	181
	4.2 Teorema de Herbrand	185

4.3 Método de Resolución	190
4.4 Demostración del Teorema de Herbrand	193

Tercera parte

Calculabilidad

Capítulo 1

Enumerabilidad efectiva, decidibilidad, 199

1.1 Conjuntos enumerables y no enumerables	200
1.2 Conjuntos efectivamente enumerables	206
1.3 Existencia de conjuntos no efectivamente enumerables	212
1.4 Conjuntos decidibles	216

Capítulo 2

Funciones recursivas, 221

2.1 Funciones recursivas primitivas	221
2.2 Funciones recursivas	228
2.3 Conjuntos recursivos	231
2.4 Funciones definidas de relaciones	235
2.5 Conjuntos recursivamente enumerables	239
2.6 Funciones recursivas parciales	244

Capítulo 3

Máquinas de Turing, 247	3.1 Máquinas de Turing	247
	3.2 Ejemplos	256
	3.3 Turing- Calculabilidad de Funciones recursivas	266

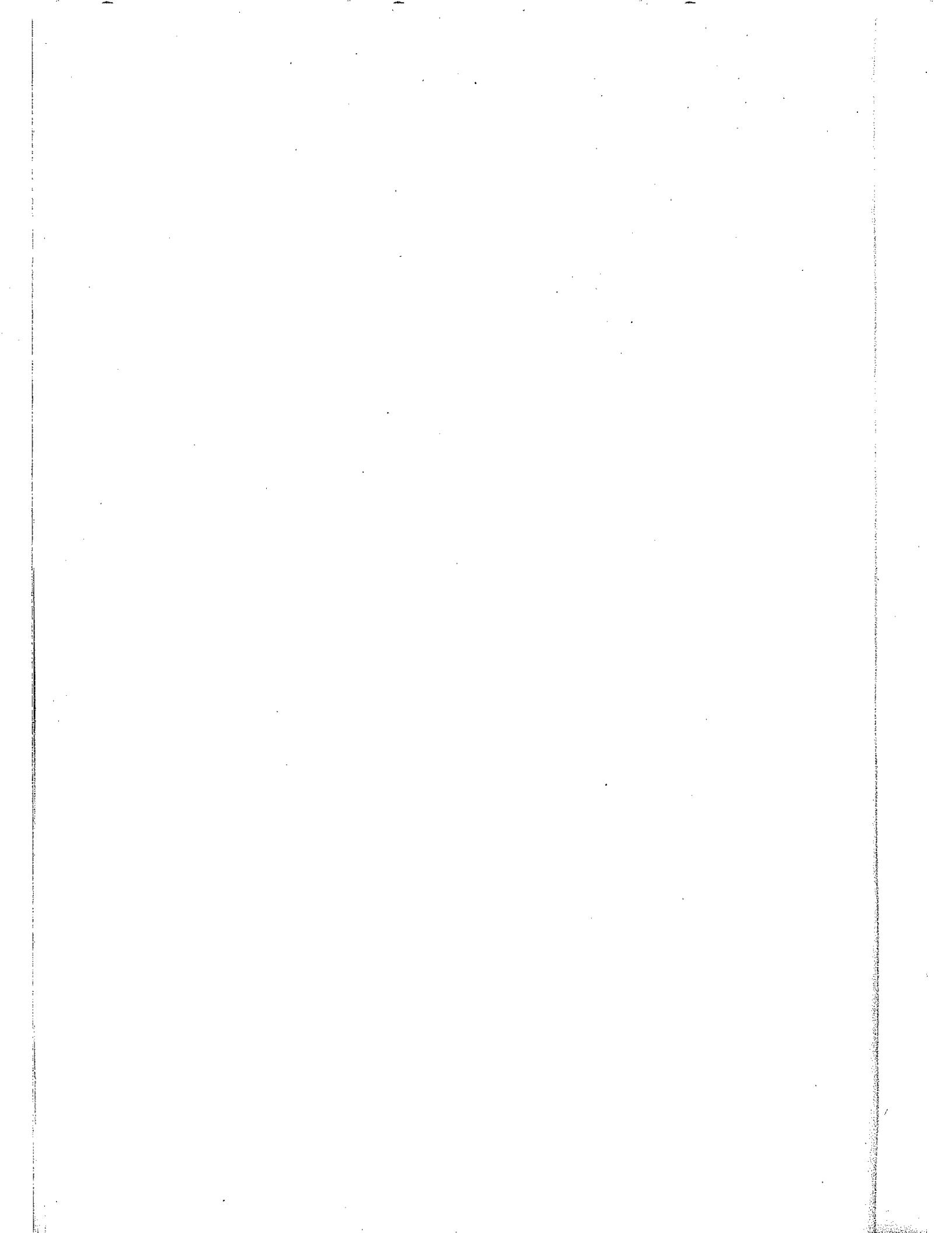
Capítulo 4

El problema de la parada, 279	4.1 Máquinas Universales	279
	4.2 El problema de la parada	288

Capítulo 5

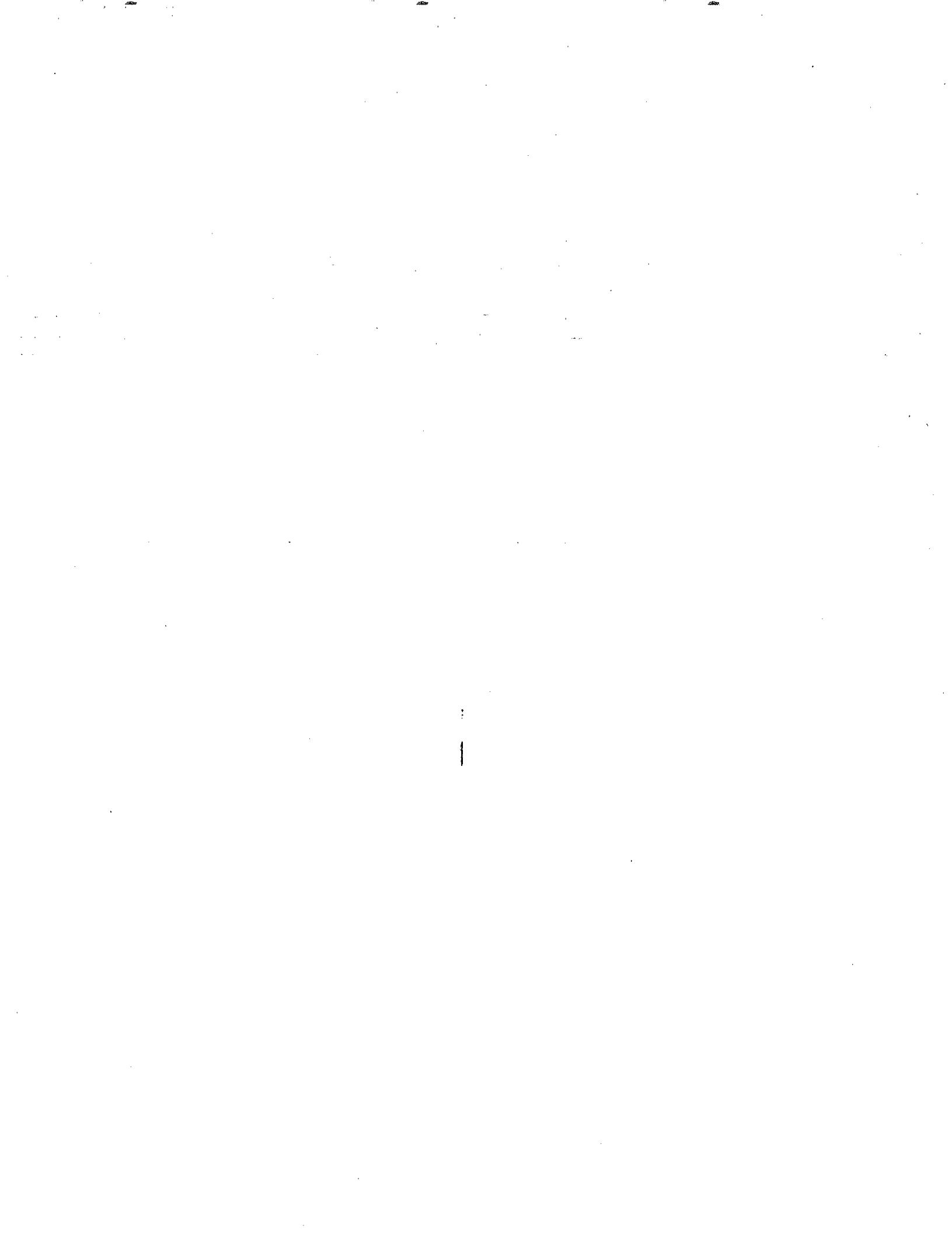
Recursividad de funciones Turing-calculables, 293	5.1 Codificación recursiva de sucesiones finitas	293
	5.2 Recursividad de la función de Ackermann	298
	5.3 Toda función Turing-calculable es recursiva	301
	5.4 Forma normal de Kleene	309

Bibliografía, 315



Explicar el sentido en que se toman las palabras, determinar bien su significado, es ir por el atajo de la verdad, es suprimir obstáculos y disputas interminables, tan prerjudiciales como inútiles.

F.J. de Caldas, "Del influjo del clima sobre los seres organizados"



Introducción

Cuando se habla de lógica moderna se usa referirse a ella como "Lógica Formal", "Lógica Simbólica", "Lógica Matemática". Históricamente tal terminología ha aparecido en el orden indicado. Podemos afirmar que la *Lógica Formal* es por lo menos tan antigua como los escritos de Aristóteles, en donde ya se observa que la validez de los silogismos depende de su forma y no del significado particular de las proposiciones que los componen. En esta etapa los símbolos son meros auxiliares en la clasificación de las diversas formas de argumentos. La *Lógica Simbólica* en cambio tiene su precursor en Leibnitz, uno de los creadores del Cálculo Infinitesimal, quien se interesó por el problema de descubrir una "characteristica universalis", es decir un método para simbolizar proposiciones y argumentos de Matemática y Metafísica y "calcular" con las formas simbólicas en orden a averiguar su verdad o validez. Este deseo se cristaliza, apoyado por los progresos del llamado "método axiomático", en el siglo XIX con los trabajos de George Boole, De Morgan, Frege, Shröeder, Pierce, y Peano. Puede decirse que esta etapa culmina en 1910 - 1913 con la monumental *Principia Mathematica* de Whithead y Russell, en donde una gran porción del raciocinio matemático se reduce a un cálculo simbólico. El desarrollo posterior corresponde a la *Lógica matemática*, cuando los sistemas formales mismos se convierten en objetos de estudio por métodos matemáticos (Metamatemática). Son Hilbert, Löwenheim, Skolem, Gödel, y Tarski, entre otros, los principales propiciadores en la primera mitad del siglo XX de este desarrollo, el cual nos ha dado resultados muy profundos en los Fundamentos de la Matemática y en la Teoría de Calculabilidad Efectiva, resultados que inciden radicalmente en áreas que van desde la pura especu-

lación filosófica hasta las aplicaciones prácticas de la Matemática. Por otra parte la Lógica se ha convertido en un instrumento poderosísimo para el estudio de las Matemáticas mismas, de manera que ha llegado a conformar una de las grandes áreas en que se divide su estudio, junto con las tradicionales como Análisis y Álgebra.

Primera parte

**El Cálculo
de
Proposiciones**



Capítulo 1

Sintaxis

1.1 Simbolización

La idea de simbolizar proposiciones o partes de las proposiciones del lenguaje ordinario con el objeto de hacer explícitas sus conexiones lógicas se remonta por lo menos hasta Aristóteles (384 - 322 A.C.). Expresiones como “todo S es P” fueron usadas por Aristóteles para simbolizar todas las afirmaciones de la forma: “todo hombre es mortal”, “todos los metales brillan”, “todos los gatos son pardos”, etc.. Esto permitió establecer la validez de ciertos argumentos de una forma dada, independientemente de los significados del sujeto S y predicado P de cada proposición involucrada. Este es el caso, por ejemplo, del silogismo : si “Todo S es P” y “Ningún P es Q” entonces “Ningún S es Q”, el cual en notación tradicional se podría expresar:

Todo S es P

Ningún P es Q

Ningún S es Q.

Otro ejemplo más sencillo sería : de “Algún S es P” se sigue “Algún P es S”.

El análisis lógico de las proposiciones a través de la conexión sujeto-predicado está incluido en el Cálculo de Predicados que estudiaremos más adelante (Cap. II). El *Cálculo de Proposiciones* que presentamos en este capítulo estudia por el contrario las conexiones lógicas de las proposiciones desde un punto de vista más general. Analizaremos las proposiciones

solamente en cuanto éstas puedan estar compuestas o ser combinación de proposiciones más sencillas. Este análisis nos llevará a ciertas proposiciones atómicas que no se dejan descomponer más, y que simbolizaremos por letras como p, q, r, etc. Ciertas partículas y expresiones lingüísticas que conectan proposiciones se simbolizarán también por símbolos especiales. Será claro que la validez de muchos argumentos sólo depende de la forma como las proposiciones involucradas se combinan a partir de las atómicas por medio de estos "conectivos". Considere por ejemplo el argumento siguiente:

Juan tiene 20 años o 22 años

Si Juan tiene 22 años,
entonces nació antes que Pedro.
Juan no nació antes que Pedro.

Juan tiene 20 años.

Evidentemente las cuatro oraciones están formadas por combinaciones de las siguientes proposiciones atómicas:

p: Juan tiene 20 años

q: Juan tiene 22 años

r: Juan nació antes que Pedro

en la forma:

1. p ó q
 2. Si q entonces r.
 3. No es el caso que r.
-

p

Ahora bien, el argumento es correcto no importa cual sea el significado de p, q, r. Si llamamos *proposición* a cualquier expresión para la cual tenga sentido decir que es *verdadera* o *falsa* (aunque no tengamos manera de saber cuál es el caso), y suponemos que p, q, r representan cualesquiera proposiciones, tenemos la siguiente demostración informal del argumento simbólico anterior: *si las premisas 1, 2, 3 son verdaderas, entonces p debe ser verdadera ; de lo contrario por la premisa 1, al ser falsa p debería ser*

verdadera q; ahora, por la premisa 2 al ser verdadera q sería verdadera r; pero esto contradiría la premisa 3 que estamos suponiendo cierta.

Es claro que la validez del argumento se basa solamente en el significado de los conectivos: “__ ó __”, “si __ entonces __”, “no es el caso que __”, y no en el de las proposiciones p, q, r.

Muchas expresiones del lenguaje ordinario conectan proposiciones para formar nuevas proposiciones, pero desde cierto punto de vista todas son substituibles por los siguientes conectivos cuya simbolización indicamos:

$p \vee q$	(incluye el caso en que ambos p y q son ciertos)	$p \vee q$
<i>si p entonces q</i>	(de p se sigue q, p implica,...)	$p \supset q$
<i>no es el caso que p</i>	(no es cierto p, es falso p,...)	$\neg p$
<i>p y q</i>		$p \wedge q$
<i>p si y solamente si q</i>		$p \leftrightarrow q$

El anterior argumento quedaría:

$$\begin{array}{c} p \vee q \\ q \supset r \\ \hline \neg r \\ \hline p \end{array}$$

También podría expresarse como una sola proposición compleja:

$$((p \vee q) \wedge (q \supset r) \wedge (\neg r)) \supset p.$$

Esta última oración es *siempre* verdadera, es decir es verdadera cualquiera que sea el significado de p, q y r, como lo hemos demostrado en el párrafo anterior. Podríamos decir que el objeto de la Lógica es hallar estas formas absolutamente verdaderas, estas *verdades lógicas*.

Note la importancia de los paréntesis para evitar ambigüedades. La propo-

sición: "Viene Ana y si viene Luis viene María" debe simbolizarse $p \wedge (q \supset r)$. Si omitimos los paréntesis queda $p \wedge q \supset r$, que podría interpretarse alternativamente como: "Si vienen Ana y Luis entonces viene María", alterándose el significado de la frase. La última proposición debe simbolizarse $(p \wedge q) \supset r$, y puede darse el caso de que esta proposición sea verdadera y la primera falsa.

Ejemplo

La proposición compuesta : "Si x es par entonces x^2 es par. Si x^2 es par entonces no es impar. Además, x es par o impar. Por lo tanto, si x^2 es impar entonces x es impar." se puede simbolizar a partir de las proposiciones atómicas: p : x es par r : x^2 es par q : x es impar s : x^2 es impar como:

$$((p \supset r) \wedge (r \supset \neg s) \wedge (p \vee q)) \supset (s \supset q).$$

A pesar de que la conclusión de esta implicación es verdadera si se entiende que hablamos de números naturales, la forma proposicional completa no es siempre verdadera, se puede dar significados a las letras p , q , r , s bajo los cuales la proposición es falsa. Utilice por ejemplo los significados originales dados arriba, pero cambiando x^2 por $3x$.

Finalmente observamos que cuando hemos hablado aquí de *argumentos*, *validez*, *demostraciones*, etc. nos hemos referido a su significado intuitivo en el raciocinio ordinario, matemático o extramatemático. Uno de los objetivos principales de la lógica es dar un sentido preciso a estos conceptos.

Ejercicios

1. Simbolice de la forma más fina posible, es decir use letras sólo para aquellas proposiciones que no son compuesta de otras (proposiciones atómicas).
 - a. Si Juan viene a la fiesta, Luis vendrá. Si Luis viene María también. Pero María viene solo si Juan viene. Por lo tanto, Luis no vendrá a la fiesta.
 - b. x es primo si y solo si x no es 1 y no tiene divisores propios.

- c. Si ella es alta o rubia entonces debe llamar la atención. Por lo tanto, si el portero estaba en la puerta y no dormía, es imposible que ella haya entrado sin que él la haya visto.
- d. Si 12 divide a n y a m y si cuando x divide tanto a m como n, se tiene que $x \leq 12$, entonces existen a y b tales que $12 = am + bn$.

2. Demuestre que el argumento:

$$\begin{array}{c} p \supset q \\ q \supset \neg r \\ s \vee p \\ \hline r \supset s \end{array}$$

es correcto, independientemente del significado de p, q, r, s.

3. Simbolice:

x es par o impar pero no ambos

x es par si y solo si x^2 es par

si x es impar entonces x^2 es impar

Demuestre que el argumento simbolizado no es correcto independientemente del significado de las proposiciones. ¿Qué premisa se podría añadir para que el argumento resulte correcto, independientemente de su significado?

4. Cada uno de los conectivos siguientes es expresable como $p \supset q$ o como $q \supset p$. Indique cuál es el caso:
- a. p, si q
 - b. p solamente si q
 - c. p con la condición de que q
 - d. p es condición suficiente para q
 - e. p es condición necesaria para q
 - f. p se sigue de q.

- 5.* Simbolice de la manera más fina posible: María no vendrá a la fiesta a menos que Luis venga.
6. El símbolo \vee se reserva para la interpretación *inclusiva* de la disyunción ó, es decir $p \vee q$ incluye como posibilidad el caso en que ambas proposiciones son verdaderas. Suponga que \vee simboliza el ó *exclusivo* ($p \vee q$ significa : sólo p ó sólo q)
- Exprese $p \vee q$ en términos de \neg , \neg e \wedge .
 - Exprese $p \vee q$ en términos de \vee , \neg e \wedge .

1.2 Fórmulas bien formadas, inducción en fórmulas

Definimos en esta sección de una manera precisa el lenguaje formal con que vamos a trabajar. Como hemos visto en la sección anterior la forma y las relaciones estructurales entre las expresiones simbólicas reflejan de alguna manera las propiedades y relaciones de los conceptos que ellas pretenden denotar. Esto justifica el estudio de las expresiones mismas como combinaciones de símbolos.

El *alfabeto* del lenguaje del Cálculo de Proposiciones consiste de los siguientes símbolos:

Conectivos proposicionales	$\neg \vee \wedge \supset \leftrightarrow$
Paréntesis	()
Letras proposicionales	$p q r s t u v w$
	$p_1 p_2 p_3 \dots$
	$q_1 q_2 q_3 \dots$

Los conectivos proposicionales se llaman respectivamente *negación*, *conjunción*, *disyunción*, *implicación* y *equivalencia*. Los paréntesis se distinguen como paréntesis *izquierdo* "(" y *derecho* ")". Si V denota el conjunto

de símbolos de este alfabeto, llamaremos V^* al conjunto de todas las cadenas o sucesiones finitas de símbolos de V , con posibles repeticiones, por ejemplo:

$$\wedge \vee p q \supset ((\leftrightarrow(\neg \neg \wedge)), (p) \supset ((q))), \leftrightarrow \leftrightarrow \leftrightarrow \leftrightarrow, r.$$

Incluiremos por conveniencia en V^* una *cadena vacía* que no contiene símbolos y que denominaremos Λ . Utilizaremos letras griegas $\alpha, \beta, \gamma \dots$ para denotar cadenas arbitrarias de símbolos. Si $\alpha = a_1 \dots a_n \in V^*$ y $\beta = b_1 \dots b_k \in V^*$ entonces $\alpha\beta = a_1 \dots a_n b_1 \dots b_k$ será la yuxtaposición o *concatenación* de α y β . En particular para la cadena vacía tendremos $\Lambda\alpha = \alpha\Lambda = \alpha$.

Entre todas esas cadenas especificaremos por medio de la definición siguiente ciertas expresiones que llamaremos *fórmulas bien formadas* o fbf. La definición es *constructiva* ya que da “instrucciones” para construir fbf.

Definición 1. Una cadena de símbolos es una fbf si y solo si puede obtenerse por medio de las reglas siguientes:

F1. Las letras proposicionales son fbf.

F2. Si α es fbf entonces $\neg(\alpha)$ es fbf.

F3. Si α y β son fbf, entonces $(\alpha) \wedge (\beta)$, $(\alpha) \vee (\beta)$, $(\alpha) \supset (\beta)$ y $(\alpha) \leftrightarrow (\beta)$ son fbf.

Ejemplo

Las expresiones de las líneas siguientes son fbf. Las de la primera línea lo son por (F1). Las demás provienen de fbf en líneas anteriores, de acuerdo con (F2) y (F3),

p	q	r	s
$(p) \wedge (q)$	$\neg (q)$	$(r) \vee (s)$	
		$((\neg(q)) \vee (r))$	$\neg ((r) \vee (s))$
$((p) \wedge (q))$	\leftrightarrow	$((\neg(q)) \vee (r))$	
$((((p) \wedge (q)))$	\leftrightarrow	$((\neg(q)) \vee (r))) \supset (\neg ((r) \vee (s)))$	

En cambio, la expresión $p \wedge q \supset (q)$ no es fbf. Basta observar que (F1) sólo produce letras proposicionales y las reglas (F2) y (F3) sólo producen cadenas que comienzan en “ \neg ” o en “(”. La siguiente cadena tampoco es bien formada, hecho cuya demostración es más difícil:
 $\neg(\neg(\neg(p) \wedge ((r) \vee (s)) \supset (p)))$.

La siguiente propiedad de las fbf's está implícita en la definición. Nos dice que toda fbf se puede “analizar” por medio de fbf's más simples:

F4. Toda fbf es una letra proposicional o tiene una de las formas $\neg(\alpha)$, $(\alpha) \wedge (\beta)$, $(\alpha) \vee (\beta)$, $(\alpha) \supset (\beta)$ ó $(\alpha) \supseteq (\beta)$, en donde α y β son también fbf's.

Llamamos \mathfrak{I} al conjunto de las fbf's. Para cada conectivo proposicional podemos definir una operación en V^* :

$$F \neg : \alpha \vdash \neg(\alpha)$$

$$F \wedge : \alpha, \beta \vdash (\alpha) \wedge (\beta)$$

$$F \vee : \alpha, \beta \vdash (\alpha) \vee (\beta)$$

$$F \supset : \alpha, \beta \vdash (\alpha) \supset (\beta)$$

$$F \Leftrightarrow : \alpha, \beta \vdash (\alpha) \Leftrightarrow (\beta)$$

Un subconjunto \mathcal{J} de V^* es *inductivo* si es cerrado bajo tales operaciones. Por supuesto \mathfrak{I} y V^* mismo son inductivos. El siguiente lema muestra que \mathfrak{I} es el conjunto inductivo más pequeño que contiene las letras proposicionales.

Lema 1. Si \mathcal{J} es un conjunto inductivo que contiene las letras proposicionales entonces $\mathfrak{I} \subseteq \mathcal{J}$

Demostración. Por inducción en n demostramos que si una fbf α de \mathfrak{I} tiene n ocurrencias de conectivos entonces $\alpha \in \mathcal{J}$.

$n = 0$. Entonces α es letra proposicional y $\alpha \in \mathcal{J}$ por hipótesis.

Suponga la hipótesis de inducción cierta para toda $k < n$ y sea α una fbf con k ocurrencias de conectivos. Por (F4) hay 2 casos:

α es $\neg(\alpha')$ ó α es $(\alpha') \square (\alpha'')$ con $\alpha', \alpha'' \in \mathfrak{I}$, donde \square tienen menos que

n ocurrencias de conectivos. Por hipótesis de inducción: $\alpha', \alpha'' \in \mathcal{I}$. Pero \mathcal{I} es inductivo, por lo tanto $\neg(\alpha') \in \mathcal{I}$ y $(\alpha') \square (\alpha'') \in \mathcal{I}$. ■

El lema anterior tiene como corolario el siguiente importante teorema que tendremos ocasión de usar muchas veces.

Principio de inducción en fórmulas. *Sea φ una propiedad que se aplica a ciertas sucesiones de símbolos y tal que:*

- I. *Cada letra proposicional tiene la propiedad,*
- II. (a) *Si α es una fbf que tiene la propiedad entonces $\neg(\alpha)$ la tiene.*
 (b) *Si α y β son fbfs que tienen la propiedad entonces $(\alpha) \square (\beta)$ tiene la propiedad, para cualquier conectivo binario \square .*
Entonces toda fbf tiene la propiedad.

Demostración. Sea $\mathcal{J} = \{\alpha \in \mathfrak{F} \mid \alpha \text{ tiene la propiedad } \varphi\}$, entonces \mathcal{J} contiene las letras proposicionales por (I) y es inductivo por (II). Por tanto $\mathfrak{F} \subseteq \mathcal{J}$. ■

Ejemplo

El número de símbolos en toda fbf es de la forma $3n + 1$. Basta comprobar I y II. Sea $L[\alpha]$ el número de símbolos en α .

- I. Si α es letra proposicional. $L[\alpha] = 1 = 3 \cdot 0 + 1$
- II. (a) Si $L[\alpha] = 3n + 1$ entonces $L[\neg(\alpha)] = 3 + (3n + 1) = 3(n + 1) + 1$.
 (b) Si $L[\alpha] = 3n + 1$ y $L[\beta] = 3k + 1$, entonces
 $L[(\alpha) \square (\beta)] = 5 + 3n + 1 + 3k + 1 = 3(n + k + 2) + 1$.

Ejercicios

1. Demuestre rigurosamente que las cadenas siguientes no son fbfs:
 $(p), \neg, p \wedge q, (pp) \wedge (q), \neg(\neg(\neg(p) \wedge ((r) \vee (s)) \supset (p))),$
 $((\neg((p) \leftrightarrow (q))) \supset (r)) \supset (((\neg(\neg(p)) \wedge (r)) \supset (q))).$
2. Demuestre por inducción en fórmulas:

- a. Si $N[\alpha] = \#$ negaciones en α y $B[\alpha] = \#$ de conectivos binarios, entonces el número de paréntesis de α es $2N[\alpha] + 4B[\alpha]$.
 - b. Toda ocurrencia de ")" en una fbf va precedida de ")" o de una letra proposicional.
 - c. Toda fbf que no es letra proposicional termina en ")".
 - d. En toda fbf el número de paréntesis izquierdos es igual al de derechos.
3. Demuestre que $\mathfrak{I} = \cap \{\mathcal{J} \mid \mathcal{J}$ es inductivo y contiene las letras proposicionales}.
4. Un conjunto \mathcal{K} de cadenas de símbolos es *proporcional* si todo elemento de \mathcal{K} es una letra proposicional o tiene una de las formas $\neg(\alpha)$, $(\alpha)\square(\beta)$, donde α y β pertenecen a \mathcal{K} y \square es un conectivo.
- a. Demuestre por inducción en el número de conectivos de los elementos de \mathcal{K} que $\mathcal{K} \subseteq \mathfrak{I}$.
 - b. Demuestre que $\mathfrak{I} = \cup \{\mathcal{K} \mid \mathcal{K}$ es proposicional}.
 - c.* Demuestre que si $\mathcal{K} = \emptyset$ entonces \mathcal{K} debe contener letras proposicionales. (Ayuda: considere un elemento de \mathcal{K} de longitud mínima.)
 - d.* Demuestre que si \mathcal{J} es un conjunto no vacío inductivo y proposicional entonces $\mathcal{J} = \mathfrak{I}'$, donde \mathfrak{I}' es el conjunto de fbf's que se obtienen utilizando solamente las letras proposicionales de \mathcal{J} .

1.3 Descomposición única, conectivo principal

Demostraremos en esta sección que el lenguaje formal introducido es *inambiguo*, es decir no es posible que una fbf α tenga dos descomposiciones distintas, v.gr.:

$$\begin{aligned}\alpha &= (\dots \beta \dots) \wedge (\dots \gamma \dots) \\ \alpha &= (\dots \beta' \dots) \vee (\dots \gamma' \dots)\end{aligned}$$

en fbfs más sencillas β, γ o β', γ' .

Sea α una cadena de símbolos arbitrarios; la cadena β es un *segmento inicial* de α si existe otra cadena γ tal que $\alpha = \beta\gamma$. Denotaremos esta relación por $\beta \leq \alpha$. La cadena vacía, Λ , es un segmento inicial de α ($\alpha = \Lambda\alpha$), α mismo es un segmento inicial de α ($\alpha = \alpha\Lambda$). Si $\lambda \leq \alpha$ y $\lambda \neq \alpha$, λ es un segmento inicial *propio* de α , y esto se denota $\lambda < \alpha$. Dada una cadena de símbolos λ , sea $I(\lambda)$ el número de paréntesis izquierdos que ocurren en λ y $D(\lambda)$ el número de paréntesis derechos. Note que si λ es fbf entonces $I(\lambda) = D(\lambda)$ (Sec. 1.2, Ejerc. 2(d)).

Lema 2. *Sea α una fbf y λ un segmento inicial propio de α , entonces:*

- (1) λ no es fbf.
- (2) $I(\lambda) \geq D(\lambda)$.

Demostración. Demostramos ambas afirmaciones simultáneamente por inducción en la fórmula α .

I. $\alpha = p$ letra proposicional. El único segmento propio es Λ , y las afirmaciones son triviales.

II. (a) Suponemos el lema para todos los segmentos propios de α y lo mostramos para $\neg(\alpha)$. Hay varios casos para un segmento inicial de $\neg(\alpha)$:

Caso 1: $\lambda = \Lambda$, $\lambda = \neg$. Trivial.

Caso 2: $\lambda = \neg(\gamma)$, donde $\gamma \leq \alpha$. Si $\gamma < \alpha$ tenemos por hipótesis de inducción (2) que $I[\gamma] \geq D[\gamma]$. Si $\gamma = \alpha$, también $I[\gamma] = D[\alpha]$. En cualquier caso, $I[\neg(\gamma)] \geq D[\neg(\gamma)]$ el segmento no puede ser fbf. Esto demuestra (2) y (1).

(b) Suponga (1) y (2) cierto para todo segmento inicial propio de α y β . Lo demostraremos para $(\alpha) \square (\beta)$.

Caso 1: $\gamma = \Lambda$. Trivial.

Caso 2: $\lambda = (\gamma)$, con $\gamma \leq \alpha$. Similar al caso 2 de (a).

Caso 3: $\lambda = (\alpha)$. (2) vale trivialmente. Para mostrar (1) suponga que (α) es fbf, entonces $(\alpha) = (\beta) \square (\mu)$ donde β y μ son fbfs, por (F4). Esto implica que $\beta < \alpha$ y β es fbf, contradiciendo la hipótesis de inducción (1) para α .

Caso 4: $\lambda = (\alpha) \square$. Trivial (Sec. 1.2, Ejerc. 2(c)).

Caso 5: $\lambda = (\alpha) \square (\gamma)$ con $\gamma \leq \beta$. Como $I[\gamma] \geq D[\gamma]$ por hipótesis de inducción (2) aplicado a β , y además $D[\alpha] = I[\alpha]$ tenemos que

$I[\lambda] = 2 + I[\alpha] + I[\gamma] \geq 2 + D[\alpha] + D[\gamma] = 1 + D[\lambda] > D[\lambda]$, lo cual implica (2) y (1) para λ . ■

Teorema de la Descomposición única. *Toda fbf es una letra proposicional, es de la forma $\neg(\alpha')$ con α' fbf, o se puede expresar de una manera única en la forma $\alpha = (\alpha') \square (\alpha'')$ con α', α'' fbf y \square un conectivo binario.*

Demostración. Por (F4) α tiene una de las tres formas indicadas. Suponga que existen $\alpha', \alpha'', \beta', \beta'' \in \mathfrak{F}$ y conectivos binarios * y # tales que $\alpha = (\alpha') * (\alpha'') = (\beta') \# (\beta'')$. Mostraremos que $\alpha' = \beta'$, $\alpha'' = \beta''$ y $* = \#$. Suponga $\alpha' \neq \beta'$, entonces $\alpha \prec \beta'$ o $\beta \prec \alpha'$. En cualquier caso se contradice el lema anterior pues un segmento inicial propio de una fbf resulta fbf.

Por lo tanto, $\alpha' = \beta'$. Esto fuerza los segmentos $*(\alpha'')$ y $\#(\beta'')$ a ser iguales. Por lo tanto: $* = \#$, $\alpha' = \beta''$ y así $\alpha'' = \beta''$. ■

Note que si nuestra definición de fbf no incluyera paréntesis el teorema sería falso, pues $\alpha = p \wedge q \vee r$ tendría dos descomposiciones con $\alpha' \neq \beta'$, $\alpha'' \neq \beta''$ y $* \neq \#$.

La primera ocurrencia de la negación en una fbf de la forma $\neg(\alpha)$, o la ocurrencia del conectivo \square en la descomposición única: $\alpha = (\alpha') \square (\alpha'')$ se llama el *conectivo principal*, lo señalamos con una flecha en el ejemplo siguiente:

$$((p) \wedge ((\neg(r)) \vee (s))) \wedge \uparrow ((\neg(p)) \supset (\neg(\neg(r)))).$$

En fórmulas muy complicadas puede ser difícil encontrar a simple vista el conectivo principal. Daremos un algoritmo que podría programarse fácilmente en una computadora para hallarlo.

Algoritmo. *Si la fórmula comienza por \neg ese símbolo es el conectivo principal, de lo contrario la fórmula debe comenzar por un paréntesis izquierdo. Recorra la fórmula de izquierda a derecha. Asigne al primer paréntesis izquierdo el número 1. Asigne a cada paréntesis izquierdo el valor asignado al paréntesis anterior más 1. A cada derecho, el valor del paréntesis anterior menos 1. El conectivo principal es el símbolo inmediatamente después del primer paréntesis al que se asigne 0.*

Ejemplo :

$$((p) \wedge ((\neg(r)) \vee (s))) \wedge ((\neg(p)) \supset (\neg(\neg(\neg(r)))))$$

12 1 23 4 32 3 2 10

Tenemos el algoritmo. ¿Qué nos garantiza que funciona?

Detrás de cada algoritmo no trivial hay un teorema que garantiza que al aplicarlo se obtiene realmente lo que se pretende.

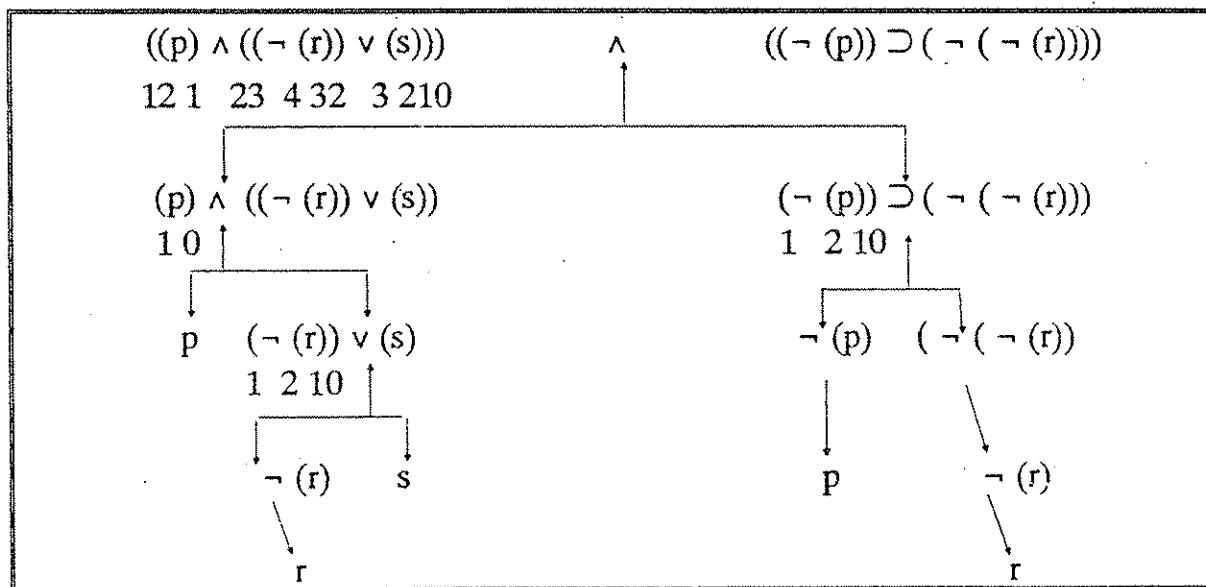
Demostración de la validez del algoritmo. Note que si λ es un segmento inicial de una fbf α , el número asignado al último paréntesis de λ según las instrucciones del algoritmo es $I[\lambda] - D[\lambda]$. Por el lema 2 este número es ≥ 0 . De manera que el número asignado al i -ésimo paréntesis de α es $n_i \geq 0$. En particular, como los paréntesis izquierdos y derechos están balanceados en α , el número asignado al último paréntesis de α es $I[\alpha] - D[\alpha] = 0$. Considere ahora $(\alpha) \square (\beta)$. Al aplicar el algoritmo a esta nueva fórmula, se comienza con un paréntesis izquierdo adicional a los que pueda haber en α al cual se le asigna 1, de manera que al i -ésimo paréntesis de α se asigna $n_i + 1 > 0$. En particular, al último de (α) se asigna $0 + 1 = 0$. Por lo tanto, al paréntesis ")" que le sigue se asigna $1 - 1 = 0$, siendo éste el primer cero. ■

Ejercicios

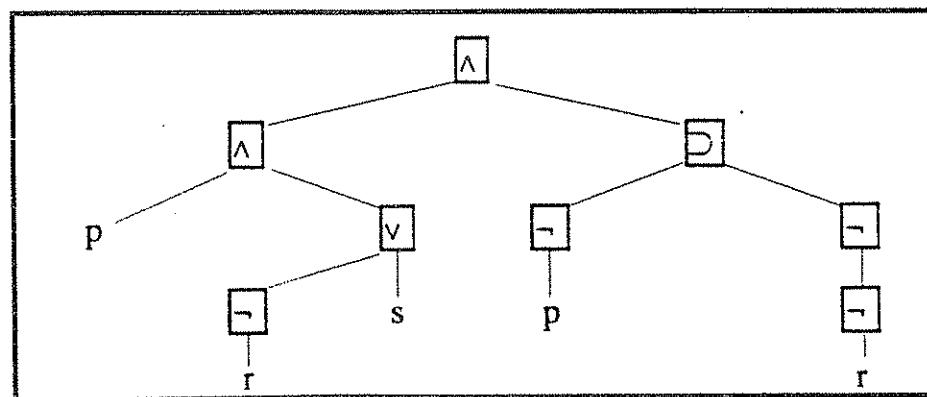
1. Demuestre que la relación \leq es un orden parcial en V^* , es decir es reflexiva, transitiva y antisimétrica.
2. Demuestre que en todo segmento inicial λ de una fbf $D[\lambda] \geq \#$ de conectivos binarios en λ .
3. Sea α una fbf y λ una cadena arbitraria de símbolos, demuestre que si $(\alpha)\lambda$ es una fbf entonces λ tiene la forma $\square(\beta)$ donde \square es un conectivo binario y β es una fbf.
4. Defina segmento final de una cadena de forma apropiada y demuestre que si λ es segmento final de una fbf, $I[\lambda] \leq D[\lambda]$.

1.4 El árbol de una fórmula, definiciones recursivas

Una vez que sabemos cómo hallar el conectivo principal, tenemos un método algorítmico para descomponer una fbf en todas sus *subfórmulas*, es decir aquellas fbf que entran en su formación, como lo indica el ejemplo siguiente:

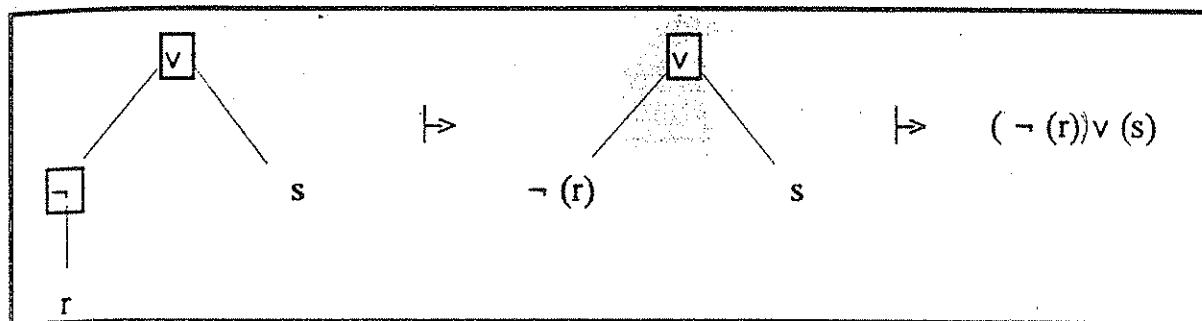


La descomposición solo se detiene en las letras proposicionales. Si conectamos cada subfórmula con la fórmula de la cual se ha desprendido obtenemos un árbol cuyos nodos son fórmulas. Si colocamos en cada nodo solamente el conectivo principal de cada subfórmula tenemos el *árbol de la fórmula*:



cuyos nodos terminales son las letras proposicionales.

Es posible reconstruir la fórmula a partir de su árbol; basta comenzar en los nodos terminales (letras proposicionales) y aplicar la operación sintáctica indicada por cada nodo. por ejemplo:



El árbol de una fórmula es más fundamental que la fórmula misma pues indica la (única) forma como ésta es combinación de operaciones aplicadas a las letras proposicionales, y es independiente del tipo de notación que se use, como veremos en la sección 1.6.

Considere ahora la siguiente definición de una función $C : \mathfrak{F} \rightarrow \mathbb{N}$,

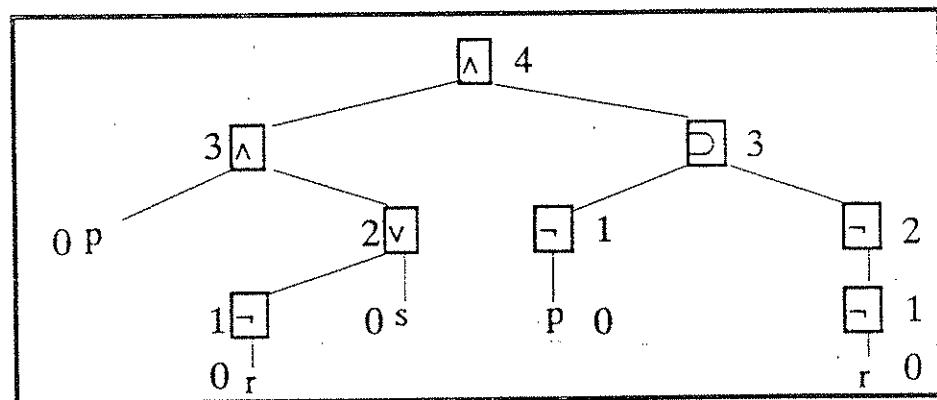
1. $C[p] = 0$ si p es letra proposicional
2. $C[\neg(\alpha)] = C[\alpha] + 1$
3. $C[(\alpha) \square (\beta)] = C[\alpha] + C[\beta] + 1$

La definición anterior se llama *recursiva* porque permite calcular el valor en una fórmula en términos de los valores en las subfórmulas que la componen. Como la longitud de las subfórmulas va disminuyendo, el cálculo debe terminar en un número finito de pasos. La función C del ejemplo simplemente cuenta el número de conectivos que ocurren en la fórmula. Sin embargo, hay otras funciones recursivamente definidas que no tienen una interpretación tan sencilla. Considere

1. $f[p] = 0$ si p es letra proposicional
2. $f[\neg(\alpha)] = f[\alpha] + 1$
3. $f[(\alpha) \square (\beta)] = \text{Max}(f[\alpha], f[\beta]) + 1$

Dada una fbf α podemos calcular $f[\alpha]$ de la manera siguiente.

Determinamos si es letra proposicional, comienza en " \neg " o comienza en "(" . En el primer caso $f[\alpha] = 0$, en el segundo caso $\alpha = \neg(\alpha')$ y debemos calcular $f[\alpha']$. En el tercer caso debemos hallar el conectivo principal, escribir $\alpha = (\alpha') \square (\alpha'')$ y calcular $f[\alpha'], f[\alpha'']$. Para hallar $f[\alpha']$, o $f[\alpha'']$ y $f[\alpha'']$ debemos repetir el análisis con cada uno de ellos. Debemos construir pues el árbol de la fórmula! Si α es la fórmula del ejemplo que consideramos al comienzo de esta sección, $f[\alpha]$ se calcula construyendo el árbol y luego calculando de las proposicionales hacia arriba según la definición de la función:



En cada nodo se da el valor de f en la subfórmula correspondiente a ese nodo.

Note que es el teorema de descomposición única el que garantiza que una función definida por recurrencia está bien definida: Si este no valiera y tuviéramos $\alpha = (\alpha') * (\beta') = (\alpha'') \# (\beta'')$ con $\alpha' \neq \beta'$, $\alpha'' \neq \beta''$, entonces al aplicar (3) del ejemplo anterior tendríamos:

$$f[\alpha] = \text{Max}(f[\alpha'], f[\beta']) + 1$$

$$f[\alpha] = \text{Max}(f[\alpha''], f[\beta'']) + 1$$

lo cual podría dar dos valores diferentes.

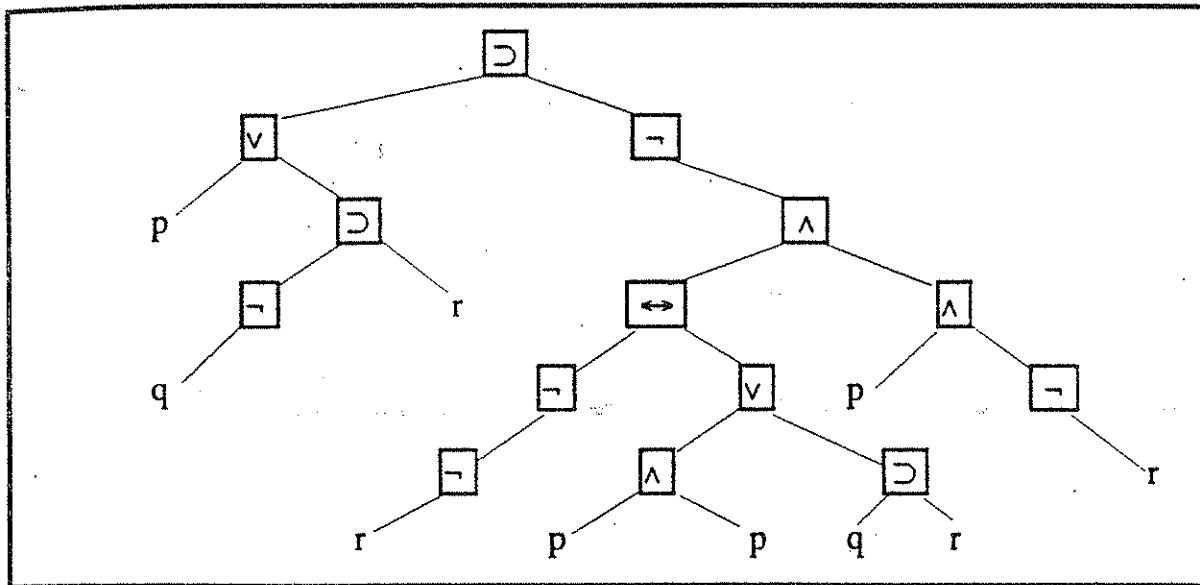
Se puede demostrar, por inducción en fórmulas, que la función f definida anteriormente da la longitud máxima (en número de arcos) de las ramas del árbol de la fórmula. Por ejemplo, $f[(p \wedge q) \vee (r \supset s)] = 2$ mientras que $f[p \wedge (q \vee (r \supset s))] = 3$ (Ejerc.3).

Ejercicios

1. Halle el árbol de la fórmulas:

$$(((p) \vee ((q) \supset (r))) \wedge (\neg((\neg(\neg(\neg(r)))) \Leftrightarrow ((q) \wedge (q)))) \Leftrightarrow (((q) \wedge (q)) \wedge (q))) \vee ((\neg(p)) \vee (q))$$

2. Halle la fórmula cuyo árbol es:



3. Demuestre que para la función f definida en esta sección, $f[\alpha] =$ longitud máxima de las ramas del árbol de α , en donde la longitud se mide por el número de arcos (o el número de nodos menos 1).
4. Dé definiciones recursivas de las funciones siguientes:
- $L[\alpha] =$ Longitud de α .
 - $P[\alpha] =$ # de ocurrencias de las letras proposicionales en α .
 - $g[\alpha] =$ # de ocurrencias del conectivo \wedge en α .
 - $h[\alpha] = \{\square \mid \square \text{ es un conectivo que ocurre en } \alpha\}$.
 - $S[\alpha] = \{\beta \mid \beta \text{ es una subfórmula de } \alpha\}$.
5. Demuestre por inducción en fórmulas que si $p[\alpha]$ es el número de ocurrencias de letras proposicionales en α , $c[\alpha]$ es el número de ocurrencias de conectivos y $f[\alpha]$ es la función del problema (3), entonces

$$\log_2(p[\alpha]) \leq f[\alpha] \leq c[\alpha]$$

Dé ejemplos en los cuales se dé el máximo y el mínimo posible para $f[\alpha]$.

6. Demuestre que si $s[\alpha]$ es el número de subfórmulas de α entonces $s[\alpha] = p[\alpha] + c[\alpha]$.
7. Demuestre por inducción en fórmulas:

$$f[\alpha] = \text{Max}\{I[\lambda] - D[\lambda] \mid \lambda \leq \alpha\}.$$

- 8.* Sean α y β fbf tales que $\alpha = \lambda \beta \gamma$, donde λ y γ son cadenas de símbolos.
Demuestre que β debe ser subfórmula de α , es decir β debe aparecer como un nodo en el árbol de α .

1.5 Supresión de paréntesis

Los paréntesis tienen el propósito de evitar ambigüedad cuando los fbf se interpretan intuitivamente como proposiciones del lenguaje ordinario o matemático. Sin embargo, no todos los paréntesis son necesarios para tal propósito. Aunque la fbf

$$(((\neg(p)) \wedge (q)) \vee ((\neg((r) \supset (s))) \supset (t))) \supset (\neg(\neg(r)))$$

es perfectamente legible para una “máquina” por ejemplo, es innecesariamente complicada y aún ilegible para nosotros. Damos en seguida dos reglas básicas para suprimir algunos paréntesis, sin que se pierda la estructura de la expresión:

- S1. Si p es letra proposicional suprima los paréntesis que encierran a p .
es decir:

$$(p) \rightarrow p$$

S2. Suprime los paréntesis que encierran una negación:

$$(\neg(\alpha)) \rightarrow \neg(\alpha)$$

Si aplicamos las reglas, nuestro ejemplo se convierte sucesivamente en:

$$(((\neg p) \wedge q) \vee ((\neg(r \supset s)) \supset t)) \supset (\neg(\neg r)) \quad (S1)$$

$$((\neg p \wedge q) \vee (\neg(r \supset s) \supset t)) \supset \neg\neg r \quad (S2)$$

Lo importante es que las reglas son reversibles y de la forma abreviada se puede reconstruir la fórmula con todos sus paréntesis. Note que cualquier supresión adicional produciría ambigüedad en el sentido de que habría varias maneras de reconstruir la fórmula original. Por ejemplo, si en la subfórmula

$$\neg(r \supset s) \supset t$$

suprimimos el par de paréntesis queda:

$$\neg r \supset s \supset t$$

Al reponer paréntesis podríamos hacerlo de varias formas:

$$(\neg(r)) \supset ((s) \supset (t))$$

$$((\neg(r)) \supset (s)) \supset (t)$$

ninguna de las cuales corresponde a la original.

1.6 Notación polaca

Podemos preguntarnos si los paréntesis son realmente necesarios para evitar ambigüedad en las fórmulas. Sorprendentemente, la respuesta es que si se cambia la notación los paréntesis no son necesarios. El lógico polaco Jan

Lucasiewicz introdujo la llamada *notación polaca* que no usa paréntesis. La idea es usar notación *prefija* (el conectivo delante de las fórmulas) en lugar de *infija* (el conectivo entre las fórmulas). Por ejemplo, $\wedge\alpha\beta$ en lugar de $(\alpha)\wedge(\beta)$. En lugar de los conectivos originales, la forma polaca usa las siguientes letras: N (por \neg), K (por \wedge), A (por \vee), C (por \supset) y E (por \leftrightarrow). Las reglas siguientes definen el conjunto de fórmulas bien formadas en notación polaca, fbf's:

- P1. Las letras proposicionales son fbf's.
- P2. Si α es fbf's entonces $N\alpha$ es fbf's.
- P3. Si α y β son fbf's entonces $K\alpha\beta$, $A\alpha\beta$, $C\alpha\beta$ y $E\alpha\beta$ son fbf's.

Naturalmente existe un principio de inducción para fbf's. La correspondencia entre la formación de fbfs y fbf's está indicada por la tabla:

Infija	Polaca
$\neg(\alpha)$	$N\alpha$
$(\alpha) \wedge (\beta)$	$K\alpha\beta$
$(\alpha) \vee (\beta)$	$A\alpha\beta$
$(\alpha) \supset (\beta)$	$C\alpha\beta$
$(\alpha) \leftrightarrow (\beta)$	$E\alpha\beta$

Ejemplos

$$(a) (\neg(p)) \wedge (q) \rightarrow KNpq$$

$$(b) ((\neg((p) \vee (r))) \leftrightarrow (s)) \supset ((\neg(p)) \wedge (q)) \rightarrow CENAprsKNpq$$

$$(c) (p \wedge q) \supset r \rightarrow CKpqr$$

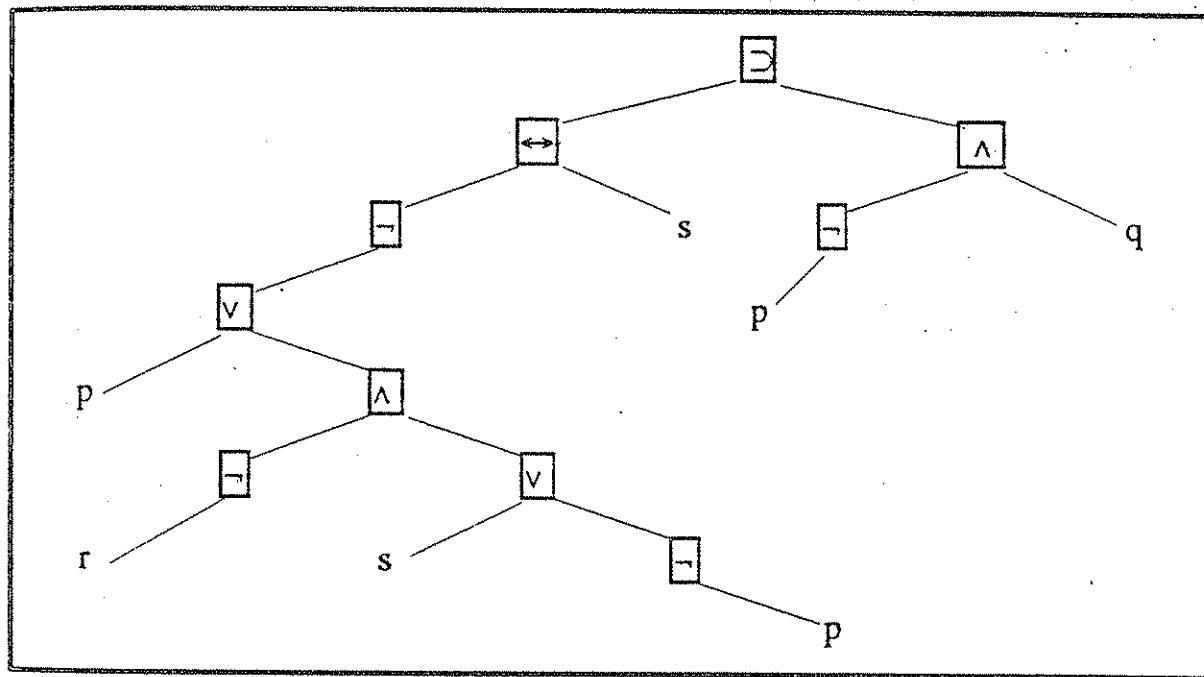
$$(d) p \wedge (q \supset r) \rightarrow KpCqr$$

Los dos últimos ejemplos muestran cómo se evita la ambigüedad. Daremos en seguida algoritmos para pasar de una forma a la otra.

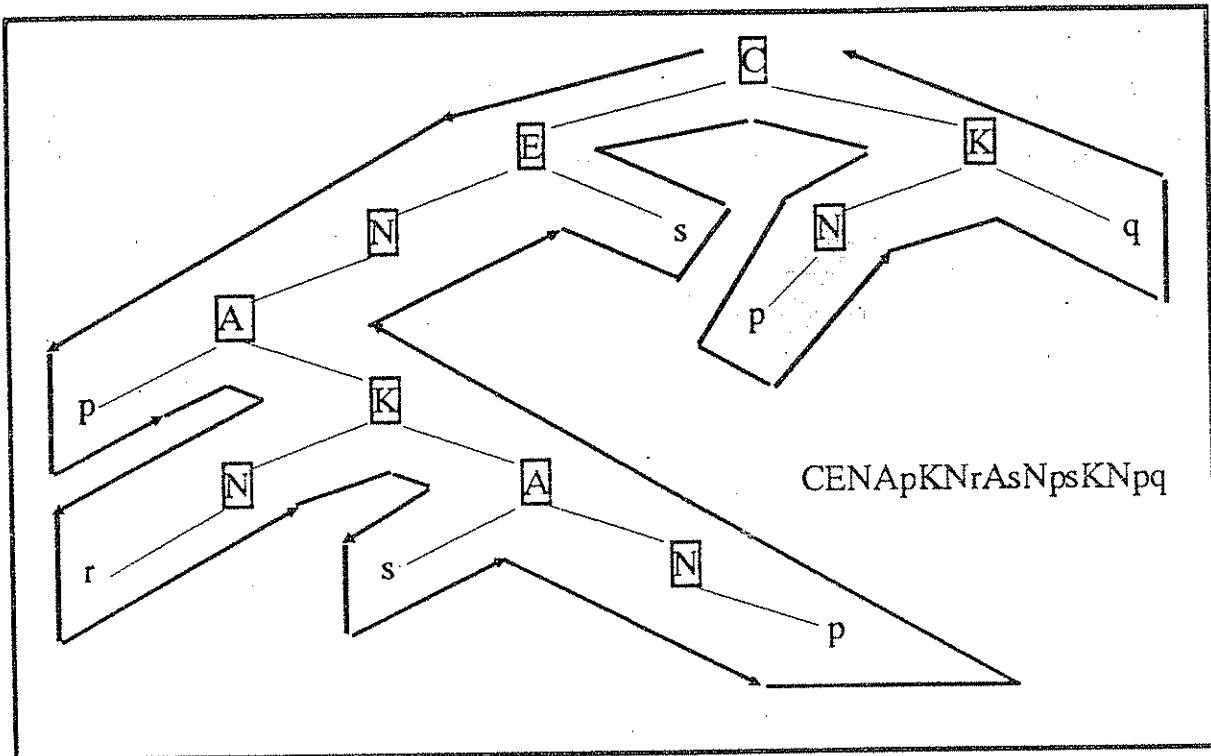
I. De infija a polaca

Primer paso. Halle el árbol de la fórmula:

$$\begin{array}{c}
 (((\neg((p)\vee((\neg(r))\wedge((s)\vee(\neg(p))))))\leftrightarrow(s))\supset((\neg(p))\wedge(q))) \\
 . \quad (\neg((p)\vee((\neg(r))\wedge((s)\vee(\neg(p))))))\leftrightarrow(s) \quad (\neg(p))\wedge(q) \\
 \neg((p)\vee((\neg(r))\wedge((s)\vee(\neg(p)))))) \quad s \quad \neg(p) \quad q \\
 (p)\vee((\neg(r))\wedge((s)\vee(\neg(p)))) \quad p \\
 p \quad (\neg(r))\wedge((s)\vee(\neg(p))) \\
 \neg(r) \quad (s)\vee(\neg(p)) \\
 r \quad s \quad \neg(p) \\
 p
 \end{array}$$



Segundo paso. Traduzca cada conectivo a su notación polaca y comenzando en la raíz del árbol (nodo superior) recorra el exterior de éste en el sentido contrario a las manecillas del reloj, anotando cada nodo la primera vez que pase por él. El recorrido termina al volver a la raíz.



Se podría describir este proceso como una caminata por el interior del árbol. El caminante parte de la raíz, camina de frente. Al salir de cada nodo escoge siempre el primer camino a su derecha. Al llegar a una letra proposicional da media vuelta. Anota cada nodo la primera vez que lo pasa, solamente. *

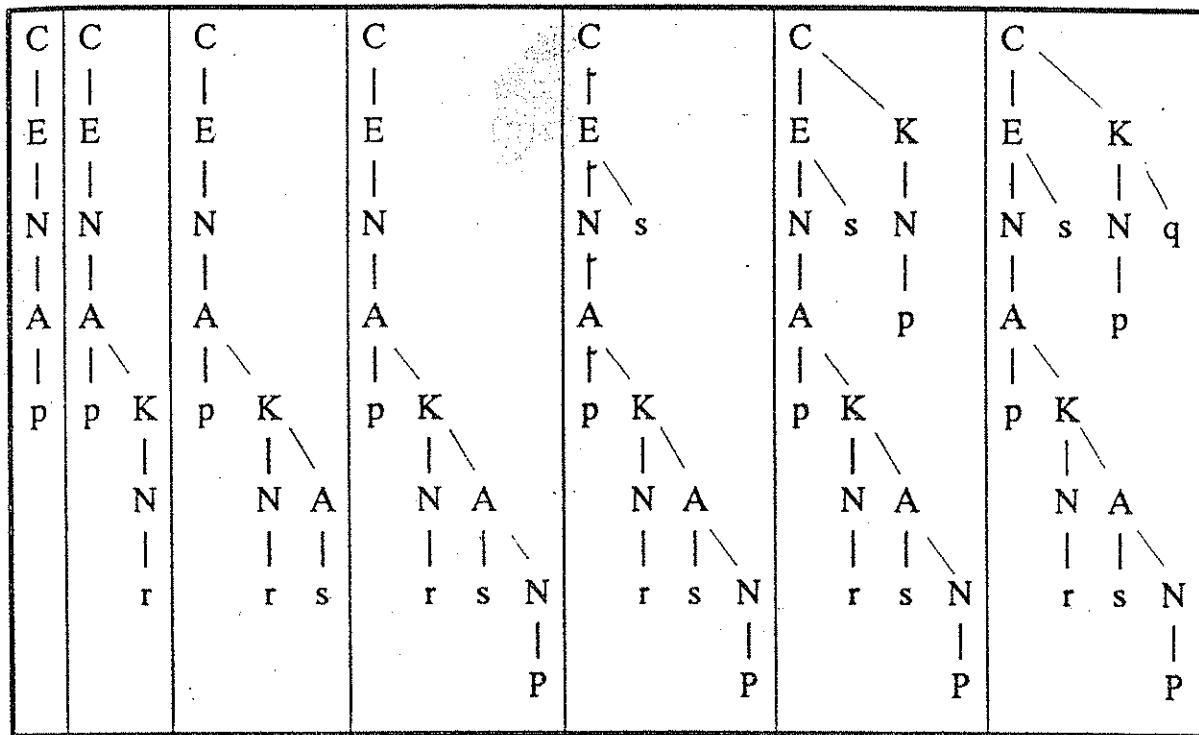
II. De Polaca a Infija

Primer paso. Dada una fórmula en forma polaca se construye su árbol en la forma siguiente. Leyendo de izquierda a derecha, busque la primera letra proposicional, el segmento inicial así determinado se toma como la rama más a la izquierda del árbol. Leyendo el resto de la fórmula, busque la siguiente letra proposicional, el segundo segmento así obtenido constituye otra rama que se conecta a la derecha del nodo binario más bajo del cual salga un solo arco en la parte ya construida del árbol. Se repite este proceso hasta agotar la fórmula.

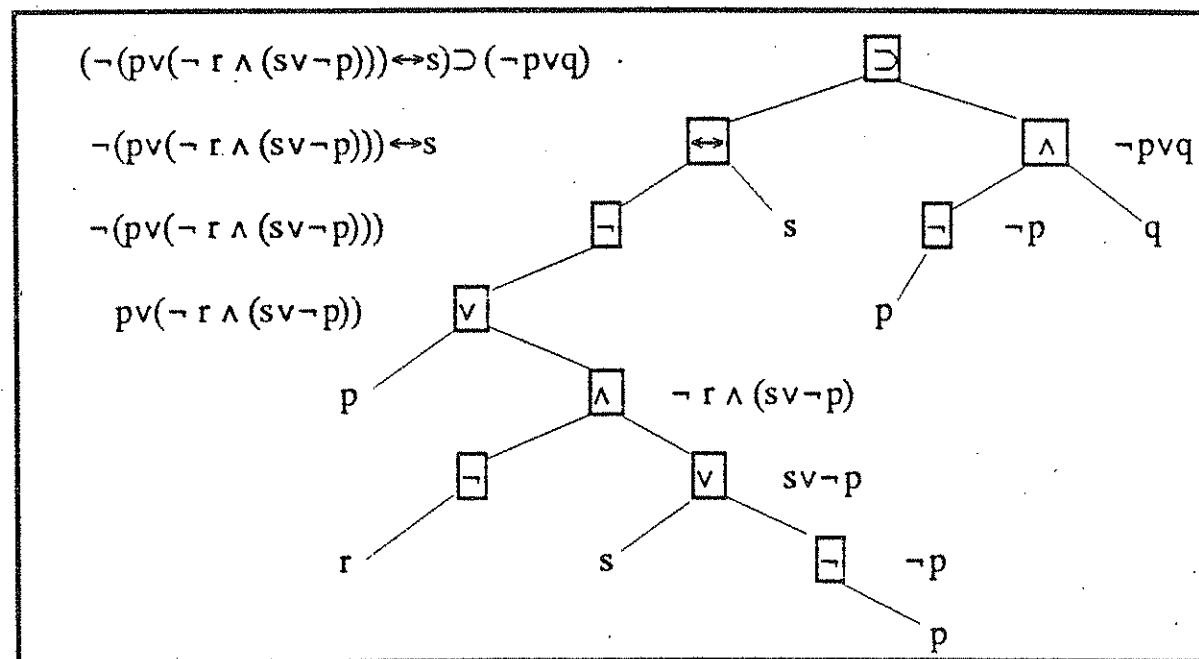
* Note que como lo sugiere el dibujo, se puede interpretar el procedimiento como una caminata por el exterior del árbol.

Ejemplo:

CENApKNrAsNpsKNpq

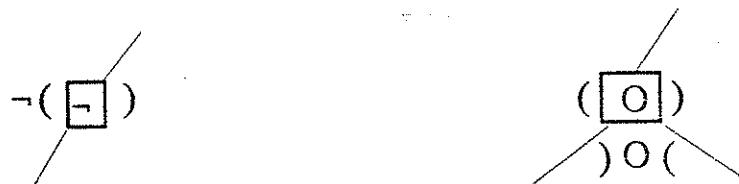


Segundo paso. Traduzca los conectivos y use el árbol para hallar la forma infija. Esto se puede hacer trabajando de las letras proposicionales hacia arriba.



Note que hemos obtenido la fórmula original del primer ejemplo.

El paso del árbol a la forma infija de la fórmula se puede realizar en un solo recorrido del exterior del árbol si asociamos a cada nodo correspondiente a un conectivo los símbolos siguientes:



El exterior del árbol se recorre en el sentido contrario a las manecillas del reloj anotando los símbolos exteriores asociados a cada nodo y las letras proposicionales que se vayan encontrando. Al retornar a la raíz se tiene la fórmula infija.

Ejercicios

1. Halle la forma polaca de las fórmulas siguientes:
 - a. $(\neg p \supset r) \wedge (q \leftrightarrow p)$
 - b. $(p \wedge (\neg r \vee s)) \wedge (p \supset \neg \neg r)$
 - c. $(\neg p \wedge (r \supset q)) \vee (p \wedge (\neg q \wedge r))$
 - d. $((p) \vee ((q) \supset (r))) \wedge (\neg ((\neg (\neg (\neg (r)))) \leftrightarrow (((q) \wedge (q)) \wedge (q)))) \vee ((\neg (p)) \vee (q))$
2. Halle el árbol y la forma infija de las fórmulas siguientes (con el mínimo de paréntesis):
 - a. (a) KNAApNNq!CN!KAKErpqst
 - b. (b) AKApNNqCqrNENNpAKKqqNpr
 - c. (c) AKApKKqpCNqppCqCKNNqpECNpKNqrq
 - d. (d) CEpCNqrNAENN!AKpCqrKpNrr
3. ¿Qué forma toma el principio de inducción para fbf^*s .?

4.* La función $\alpha \mapsto \alpha^*$ que transforma fórmulas a su forma polaca puede definirse por recurrencia:

- i) $\alpha^* = \alpha$
- ii) $[\neg(\alpha)]^* = N\alpha^*$
- iii) $[(\alpha) \wedge (\beta)]^* = K\alpha^*\beta^*$
 $[(\alpha) \vee (\beta)]^* = A\alpha^*\beta^*$
 $[(\alpha) \supset (\beta)]^* = C\alpha^*\beta^*$
 $[(\alpha) \leftrightarrow (\beta)]^* = E\alpha^*\beta^*$

Démuestre que si \wp es el conjunto de las fbf*s entonces $*$ es una función inyectiva y sobreycactiva de \mathfrak{F} en \wp . Demuestre que los algoritmos dados en esta sección para traducir de infija a polaca y viceversa realmente calculan las funciones $F[\alpha] = \alpha^*$ y su inversa.

5.* Diseñe el diagrama de flujo de programas de computador que traduzcan de notación infija a polaca y que traduzcan de polaca a infija.

6. Defina un nuevo tipo de fórmulas bien formadas, fbf#s:

- i) Toda letra proposicional es fbf#.
- ii) Si α es fbf# entonces $\neg\alpha$ es fbf#.
- iii) Si α y β son fbf#s entonces $\alpha \square \beta$ es fbf#, donde $\square \in \{\wedge, \vee, \supset, \leftrightarrow\}$.

Determine si las fórmulas siguientes son fbf#s y póngalas en forma "standard".

- a. $r \wedge p) \vee r \supset q))$
- b. $r \wedge p \vee r)) \supset q)$
- c. $r \wedge p) \vee r) \supset q)$
- d. $r \wedge p, o r) \supset q))$
- e. $r \wedge p \vee r \supset q))))$
- f. $\neg q \wedge \neg \neg p) \wedge r))))$
- g. $p \wedge \neg q)) \wedge \neg r) \wedge s)$

7. Dé una definición recursiva que transforme una fbf en su correspondiente fbf#.

8** Demuestre que el teorema de descomposición única vale para fbf#s.

1.7 Decidibilidad

¿Dada una cadena de símbolos, es posible decidir de una manera mecánica si es una fbf o no? Para nosotros es fácil hacerlo cuando se trata de cadenas cortas en notación infija, pero no es claro cómo dar un conjunto de instrucciones que permita hacerlo en todos los casos (o en la forma polaca) y que se puedan programar en una máquina. Además, ¿qué es lo que nos permite hacer la decisión correcta en algunos casos?

El siguiente es un algoritmo de decisión para la forma infija:

- (1) Substituya todas las ocurrencias de letras proposicionales por 0.
- (2) En la expresión resultante substituya todas las ocurrencias de $\neg(0)$ por 0.
- (3) En la expresión resultante substituya todas las ocurrencias de $(0) \square (0)$ por 0.
- (4) Vuelva al paso (2).

El proceso se detiene cuando (2) y (3) no son aplicables. Si el resultado final es 0 la expresión es fbf, de lo contrario no lo es.

Ejemplos

- (a) $((p) \wedge ((\neg(r)) \supset (s))) \vee ((p) \supset (\neg(\neg(r))))$
 $((0) \wedge ((\neg(0)) \supset (0))) \vee ((0) \supset (\neg(\neg(0))))$
 $((0) \wedge ((0) \supset (0))) \vee ((0) \supset (\neg(0)))$

$$\begin{array}{c}
 (((0) \wedge (0)) \vee ((0) \supset (0))) \\
 ((0) \wedge (0)) \vee ((0) \supset (0)) \\
 0 \qquad \qquad \qquad \text{es fbf}
 \end{array}$$

$$(b) \neg(\neg(\neg(p) \wedge ((r) \vee (s)) \wedge (p)))$$

$$\begin{array}{l}
 \neg(\neg(\neg(0) \wedge ((0) \vee (0)) \wedge (0))) \\
 \neg(\neg(0 \wedge (0) \wedge (0))) \\
 \neg(\neg(0 \wedge 0))
 \end{array}$$

No se puede reducir más, no es fbf.

Dejamos como ejercicio el probar la validez de este algoritmo de una manera rigurosa (pues intuitivamente es claro que debe funcionar). Es claro que si leemos símbolo por símbolo al ejecutar el algoritmo, debemos recorrer la expresión varias veces. Cabe preguntar si existe un algoritmo de decisión para las fórmulas bien formadas que se pueda ejecutar recorriendo la expresión una sola vez de izquierda a derecha (o de derecha a izquierda). Veremos que en notación polaca la respuesta es afirmativa.

Algoritmo en notación polaca. En la forma polaca el algoritmo de decisión toma una forma muy sencilla. Primeramente se da un valor a cada símbolo del alfabeto:

- (1) $v[p] = 1$, si p es letra proposicional
- (2) $v[N] = 0$
- (3) $v[K] = v[A] = v[C] = v[E] = -1$.

Algoritmo. Dada una expresión $s_n s_{n-1} \dots s_1$ con símbolos en el alfabeto de la notación polaca, se asigna un rango R_i a cada símbolo s_i de la manera siguiente. Se recorre la expresión de derecha a izquierda y comenzando con el valor de s_1 se va sumando el valor de cada símbolo, más precisamente:

$$R_1 = v[s_1]$$

$$R_{k+1} = R_k + v[s_{k+1}]$$

Ejemplos:

KCNprEqp	KpNCApq	KANp
1 2 3 32 121	01 0 0 121	-1 0 11

Esto es suficiente para determinar si la expresión es bien formada por el siguiente resultado.

Teorema 2. $s_n s_{n-1} \dots s_1$ es fbf* si y solo si (1) $R_i \geq 1$ para todo i , y (2) $R_n = 1$.

Demostración: Tiene dos partes.

(A) Si $\alpha = s_n s_{n-1} \dots s_1$ es fbf* entonces valen (1) y (2). Esto se demuestra por inducción en fórmulas en forma polaca (vea ejercicio 3 de 1.6)

I. α es letra proposicional, entonces $R_n = R_i = R_1 = v[\alpha] = 1$.

II. (a) Supóngalo cierto para α y considere $N\alpha = s_n s_{n-1} \dots s_1$.

Como $\alpha = s_{n-1} \dots s_1$ es fbf*, entonces por hipótesis de inducción $R_i \geq 1$ para $i \leq n-1$ y $R_{n-1} = 1$. Como $s_n = N$, entonces

$$R_n = R_{n-1} + v[s_n] = 1 + 0 = 1.$$

(b) Supóngalo para α y β y considere $\Box\alpha\beta = s_n s_{n-1} \dots s_k s_{k-1} \dots s_1$ donde $\alpha = s_{n-1} \dots s_k$, $\beta = s_{k-1} \dots s_1$ y $\Box = s_n$ es un conectivo binario.

Por hipótesis β es fbf* y $R_{k-1} = 1$, $R_i \geq 1$ para $1 \leq i \leq k-1$. Por hipótesis α es fbf*, además para $k \leq i \leq n-1$ R_i será uno más que el valor de R_i que se habría obtenido al aplicar el algoritmo a la cadena α sola, pues se comienza con un exceso de $R_k = 1$. Por hipótesis de inducción aplicada a α , $R_i \geq 1$ para $k \geq i \geq n-1$, y $R_{n-1} = 1$. Por lo tanto $R_i \geq 2$ y $R_{n-1} = 2$.

Finalmente, $R_n = R_{n-1} - 1 = 1$. ■

(B) Si valen (1) y (2) para $s_n s_{n-1} \dots s_1$, entonces es fbf*. Esto se demuestra por inducción en la longitud n de la cadena.

I. $n=1$. Como $v[s_1] = R_1 = R_n = 1$, s_1 debe ser letra proposicional.

II. Suponga que toda sucesión con menos que n símbolos que satisface (1) y (2) es fbf*, y sea $s_n s_{n-1} \dots s_1$ una sucesión que satisface (1) y (2).

Caso 1: s_n es letra proposicional. Entonces $n=1$, de lo contrario $1 = R_n = 1 + R_{n-1}$ y $R_{n-1} = 0$, contradiciendo (1). Por tanto la expresión se reduce a una letra proposicional.

Caso 2: $s_n = N$. Entonces $n \geq 2$, de lo contrario $R_1 = 0$. Además, $R_{n-1} = R_n - v[s_n] = R_n - 1$. Esto prueba que $s_{n-1} \dots s_1$ cumple (1) y (2) y por hipótesis de inducción es fbf*. Entonces $N s_{n-1} \dots s_1$ es fbf*.

Caso 3: $s_n = \square$. Entonces $R_{n-1} = 2$. Halle el primer símbolo de izquierda a derecha diferente de s_n y con rango $R_k = 1$:

$$\begin{array}{ccccccc} \square & s_{n-1} & \dots & s_{k+1} & s_k & \dots & s_1 \\ 1 & 2 & & & 1 & 1 & \nearrow \\ & & \swarrow & | & & & \\ & & & & & & \geq 2 \end{array}$$

Debe existir, pues de las posibilidades para R_1 que son 1, 0, -1, sólo la primera no viola (1). Por inducción, $s_k \dots s_1$ es fbf*. Además, para $k+1 \leq i \leq n-1$, $R_i = R'_i + 1$ donde R'_i sería el rango asignado a los símbolos de $s_{n-1} \dots s_{k+1}$ comenzando con $R'_{k+1} = v[s_{k+1}]$, es decir a la sucesión aislada. Por tanto $R'_i \leq 1$ para $k+1 \leq i \leq n-1$ y $R'_{n-1} = 1$. Por hipótesis de inducción esta sucesión es fbf*. Por definición la sucesión total es fbf*. ■

Debe notarse que este algoritmo vale para cualquier lenguaje de expresiones construidas iterando en forma prefija ciertos operadores de un número determinado de variables.

Ejemplo

Sean +, x dos operadores binarios y * un operador ternario y sean a, b, c, etc. variables. El lenguaje se forma por las reglas:

(1) a, b, c, ... son expresiones del lenguaje,

- (2) si α , β y γ son expresiones del lenguaje entonces $+\alpha\beta$, $x\alpha\beta$ y $\star\alpha\beta\gamma$ son expresiones del lenguaje.

Para determinar si una cadena pertenece al lenguaje se usa el mismo algoritmo que para la forma polaca pero dando valores

$$v[+] = v[x] = -1 \quad v[*] = -2$$

En general, si O es un operador n -ádico se le debe asignar el valor $v[O] = -(n-1)$.

Por ejemplo, $+dx+ab*+aaa+ab$ está en el lenguaje, pero $*a+a+aaa$ no.

1212321 3432121

013232321

Ejercicios

1. Demuestre la validez del primer algoritmo de decisión para fbf's dado en esta sección.
2. Use el Teorema 1 para demostrar que si α es una fbf* y λ es una sucesión no vacía de símbolos entonces no $\alpha\lambda$ es fbf*. (Ayuda: Si $\alpha\lambda$ fuera fbf* entonces R_n en $\alpha\lambda$ sería ≥ 2).
3. Por el problema anterior un segmento inicial de una fbf* no es fbf*. Use esto para demostrar que el teorema de descomposición única vale para fbf*'s. Es decir, si $\square\alpha\beta = *'\alpha'\beta'$ con \square , $*$, conectivos polacos y α , $*'$, α' , β' fbf*'s entonces $\square = *$, $\alpha = \alpha'$ y $\beta = \beta'$.
4. Describa un algoritmo que leyendo una sola vez de *izquierda a derecha* una expresión en el alfabeto de la forma polaca decida si es o no fbf*.
5. ¿Existe un algoritmo que leyendo una sola vez de izquierda a derecha una expresión en el alfabeto del Cálculo de Proposiciones (forma infija) decida si ésta es fbf o no?

6. Si P es un conjunto de letras proposicionales sea $\mathfrak{F}(P)$ el conjunto de fbf's construidas utilizando solamente las letras de P .
* ¿Son finitos, infinitos enumerables o no enumerables los conjuntos siguientes?
a. $\mathfrak{F}(p)$ donde p es una letra proposicional.
b. $\mathfrak{F}(p_1, \dots, p_n)$ donde p_1, p_2, p_3, \dots es una lista infinita de letras proposicionales.
7. Demuestre que el conjunto $\{\alpha \in \mathfrak{F}(p_1, \dots, p_n) \mid \alpha \text{ tiene longitud } \leq k\}$ tiene cardinal $\leq (n+7)^k$.
8. Describa un algoritmo para generar todas las fbf's en $\mathfrak{F}(p, q, r)$. (Como se trata de un conjunto infinito, se pide un procedimiento para generar fórmulas consecutivamente, tal que cada fbf aparezca eventualmente en la lista generada, si se espera suficientemente).



Capítulo 2

Deducción formal

Uno de los temas de la lógica es el explicar lo que es una demostración o deducción. Continuamente hacemos raciocinios o utilizamos argumentos cuya validez puede parecer a veces inmediata, pero que exige en ocasiones dar al interlocutor una demostración, es decir una justificación consistente en transformarlo y descomponerlo en “pasos” más sencillos que resulten evidentes. En las ciencias, y especialmente en la Matemáticas, estos argumentos pueden ser muy complejos y no resultan en manera alguna evidentes, así que su demostración debe sistematizarse. Ya en los famosos Elementos de Euclides (330-320 A.C.) se presentan los teoremas de la Geometría y la Aritmética como deducidos sistemáticamente de axiomas y postulados iniciales por medio de cadenas de pasos puramente lógicos. La utilización explícita de este método axiomático-deductivo es en realidad anterior a Euclides, como se desprende de algunos pasajes de Platón (427-347 A.C.).

“Tu no ignoras, presumo, que aquellos que se ocupan de la geometría, aritmética y de otras ciencias del mismo género, una vez que han puesto por hipótesis el impar y el par, las figuras, tres especies de ángulos y cosas análogas... partiendo de esas hipótesis y siguiendo una cadena ininterrumpida logran, de consecuencia en consecuencia, la demostración que se habrían propuesto encontrar”.

Hoy día uno de los tratados más famosos de la Matemática, la serie de Bourbaki, comienza con la afirmación: “Después de los griegos, quién habla de Matemáticas habla de demostración”.

¿Cuáles son esos “pasos lógicos”, esas reglas, que nos permiten pasar de proposiciones supuestas verdaderas a otra proposición que debemos entonces aceptar como verdadera? Si examinamos de nuevo el argumento dado como ejemplo en Sec 1.1:

$$\begin{array}{c} p \vee q \\ q \supset r \\ \neg r \\ \hline p \end{array}$$

vemos que la demostración del mismo que allá dimos consistió en primer lugar en utilizar el famoso método de *prueba por contradicción*:

Si de α, β, \dots y $\neg \mu$ se sigue una contradicción, entonces de α, β, \dots se sigue μ ; y la utilización luego de dos “reglas de deducción” adicionales. En efecto, supusimos $\neg p$ además de las otras premisas y con la aplicación consecutiva de las dos reglas siguientes llegamos a la contradicción entre r y la premisa $\neg r$:

$$\begin{array}{ccc} \vdots & \begin{array}{c} p \vee q \text{ (Modus} \\ \neg p \text{ tollendo} \\ \hline q \end{array} & \begin{array}{c} q \supset r \text{ (Modus} \\ q \text{ ponendo} \\ \hline r \end{array} \end{array}$$

Estos y muchos otros métodos y reglas de deducción fueron reconocidos desde la antigüedad, incluyendo a los silogismos aristotélicos, lo mismo que ciertos “principios” de validez universal, por ejemplo:

$$\begin{array}{ccc} \begin{array}{c} p \supset q \text{ (Modus} \\ \neg q \text{ tollendo} \\ \hline \neg p \end{array} & & \begin{array}{c} p \supset r \text{ (Prueba por casos)} \\ q \supset r \\ \hline (p \vee q) \supset r \end{array} \end{array}$$

2.1 Cálculo Proposicional Axiomático

- $\neg(\alpha \wedge \neg\alpha)$ principio de no contradicción
- $\alpha \vee \neg\alpha$ principio del tercio excluso
- $a = a$ principio de identidad.

La lista de reglas y métodos, lo mismo que los nombres con que fueron bautizados sería interminable.

Hoy sabemos que las “leyes lógicas” se pueden separar en aquellas que sólo dependen de la estructura puramente proposicional de las oraciones, es decir la que solo hace referencia a los conectivos, y aquellas dependientes de la estructura más fina asociada con el análisis sujeto-predicado y cuantificacional. También sabemos que todas ellas pueden reducirse en un sentido muy preciso a unos pocos axiomas y reglas.

En este capítulo comenzamos el estudio de la lógica propiamente dicha desarrollando un sistema deductivo para el lenguaje de Cálculo de Proposiciones. Consistirá de 9 principios fundamentales (axiomas proposicionales) y de una sola regla de deducción o inferencia (Modus Ponens). Desarrollaremos y estudiaremos el sistema lo suficiente para convencernos de que todas las formas corrientes de demostración que involucran sólo conectivos proposicionales son reducibles a los axiomas y reglas de nuestro sistema.

2.1 Cálculo Proposicional Axiomático

Por simplicidad, consideraremos que el lenguaje contiene solamente los símbolos de conectivo \neg , \wedge , \vee , \supset . y supondremos que \Leftrightarrow es una abreviación:

$\alpha \Leftrightarrow \beta$ abrevia $(\alpha \supset \beta) \wedge (\beta \supset \alpha)$.

Las variables α , β , γ , ... denotarán en adelante fórmulas bien formadas. Suprimiremos paréntesis cuando ello no produzca ambigüedad (véase 1.5).

Definición 1. Cualquier fbf que tenga una de las formas siguientes es un *axioma proposicional*:

- A1. $\alpha \supset (\beta \supset \alpha)$
- A2. $(\alpha \supset (\beta \supset \gamma)) \supset ((\alpha \supset \beta) \supset (\alpha \supset \gamma))$
- A3. $(\neg \alpha \supset \neg \beta) \supset (\beta \supset \alpha)$
- A4. $\alpha \wedge \beta \supset \alpha$
- A5. $\alpha \wedge \beta \supset \beta$
- A6. $(\alpha \supset \beta) \supset ((\alpha \supset \gamma) \supset (\alpha \supset \beta \wedge \gamma))$
- A7. $\alpha \supset (\alpha \vee \beta)$
- A8. $\beta \supset (\alpha \vee \beta)$
- A9. $(\alpha \supset \gamma) \supset ((\beta \supset \gamma) \supset (\alpha \vee \beta \supset \gamma))$

Las expresiones anteriores son *esquemas axiomáticos* pues α , β , γ pueden substituirse por fbf's cualesquiera; por ejemplo, la fórmula

$$(\neg \neg p_1 \supset \neg(p_1 \vee \neg p_2)) \supset ((p_1 \vee \neg p_2) \supset \neg p_1),$$

es un axioma (un *caso* de A3). Estos esquemas representan “leyes lógicas” fundamentales, de cuya validez intuitiva debe tratar de convencerse el lector. Sin embargo, el aceptar esas leyes como “verdaderas” no es necesario para el desarrollo que sigue.

Definición 2. Una sucesión de fbf's $\alpha_1 \alpha_2 \dots \alpha_k$ se llama una *deducción formal* si cada α_i ($1 \leq i \leq k$) es un axioma proposicional o resulta de fórmulas anteriores de la sucesión por la aplicación de la *regla de deducción* siguiente, *Modus Ponens*, (MP): de α y $\alpha \supset \beta$ se puede pasar a β .

Definición 3. Una fbf es un *teorema formal* si aparece como la última fórmula de una deducción formal; este hecho lo expresamos escribiendo $\vdash \alpha$.

Teorema 1*. $\vdash \neg \alpha \supset \alpha$ (cualquiera que sea α).

La siguiente es una deducción formal de la fórmula $\alpha \supset \alpha$. Numeramos cada fórmula introducida e indicamos a la derecha la justificación de su introducción.

1. $(\alpha \supset ((\alpha \supset \alpha) \supset \alpha)) \supset ((\alpha \supset (\alpha \supset \alpha)) \supset (\alpha \supset \alpha))$ A2
2. $\alpha \supset ((\alpha \supset \alpha) \supset \alpha)$ A1
3. $(\alpha \supset (\alpha \supset \alpha)) \supset (\alpha \supset \alpha)$ MP 1, 2
4. $\alpha \supset (\alpha \supset \alpha)$ A1
5. $\alpha \supset \alpha$ MP 3, 4

(Ponemos un asterisco (*) a los resultados que sólo dependen de A1, A2, A3)

Ejercicios

1. Halle deducciones para los siguientes teoremas :

- a. $(\neg \neg \beta \supset \neg \alpha) \supset (\alpha \supset \neg \beta)$
- b. $\alpha \supset (\beta \supset (\alpha \supset \beta))$
- c. $\alpha \supset (\alpha \wedge \alpha)$ (Use Teorema 1)
- d. $(\alpha \wedge \alpha) \supset \alpha$
- e. $(\alpha \wedge \beta) \supset (\beta \wedge \alpha)$
- f. $\alpha \leftrightarrow \alpha$ (Use Teorema 1)
- g. $\alpha \supset (\alpha \vee \alpha)$
- h. $(\alpha \vee \alpha) \supset \alpha$
- i. $(\alpha \vee \beta) \supset (\beta \vee \alpha)$
- j. $((\alpha \wedge \beta) \vee (\beta \wedge \alpha)) \supset \alpha$ (Use A4, A9)

2. ¿Son verdaderas o falsas las afirmaciones siguientes?

- a. Toda fórmula de una deducción formal es una teorema formal.
- b. Puesto que todos los axiomas del sistema deductivo establecido arriba son fórmulas implicativas, es decir tienen la forma $\alpha \supset \gamma$, entonces en este sistema sólo se van a poder deducir fórmulas implicativas; en particular no será posible deducir la fórmula $\alpha \vee \neg \gamma$.

3. Determine si las siguientes sucesiones de fbf's son deducciones formales o no:

a. $\neg p \supset (\neg q \supset \neg p)$
 $(\neg q \supset \neg p) \supset (p \supset q)$
 $((\neg q \supset \neg p) \supset (p \supset q)) \supset (\neg p \supset ((\neg q \supset \neg p) \supset (p \supset q)))$
 $(\neg p \supset ((\neg q \supset \neg p) \supset (p \supset q))) \supset ((\neg p \supset (\neg q \supset \neg p)) \supset (\neg p \supset (p \supset q)))$
 $\neg p \supset ((\neg q \supset \neg p) \supset (p \supset q))$
 $(\neg p \supset (\neg q \supset \neg p)) \supset (\neg p \supset (p \supset q))$
 $\neg p \supset (p \supset q)$

b. $(r \supset s) \supset ((s \supset r) \supset (r \supset s))$
 $((s \supset r) \supset (r \supset s)) \supset (((s \supset r) \supset r) \supset ((s \supset r) \supset s))$
 $(r \supset s) \supset (((s \supset r) \supset r) \supset ((s \supset r) \supset s)).$

4. En cada caso describa un algoritmo para decidir si una cadena de V^* pertenece o no al conjunto indicado:

- a. El conjunto de los axiomas proposicionales.
- b. El conjunto de las deducciones formales. (Note que una sucesión de fbf's se puede considerar como una cadena en V^* .)
- c.* El conjunto de los teoremas formales.

2.2 Deducción con premisas, reglas derivadas

Dada una colección de fórmulas $\sigma_1, \sigma_2, \dots, \sigma_n$, que llamamos *premisas*, una *deducción formal con las premisas* $\sigma_1, \sigma_2, \dots, \sigma_n$ es una sucesión de fórmulas $\sigma_1, \sigma_2, \dots, \sigma_k$, tal que cada σ_i se ha introducido por una de las razones siguientes (a) es una axiomática propia, (b) es una premisa, (c) resulta de fórmulas anteriores por Modus Ponens.

La última fórmula de una deducción con premisas $\sigma_1, \dots, \sigma_n$ se llama una *consecuencia formal* de $\sigma_1, \dots, \sigma_n$. Usamos la notación $\sigma_1, \dots, \sigma_n \vdash \sigma$ para

indicar que σ es una tal consecuencia formal. En particular tenemos que $\vdash \sigma$ si y solo si σ es una consecuencia formal del conjunto vacío de premisas.

Teorema 2*. $\alpha \supset \beta, \beta \supset \gamma \vdash \alpha \supset \gamma$

Lo siguiente es una deducción de $\alpha \supset \gamma$ con premisas $\alpha \supset \beta, \beta \supset \gamma$:

1. $\beta \supset \gamma$	P (Premisa)
2. $(\beta \supset \gamma) \supset (\alpha \supset (\beta \supset \gamma))$	A1
3. $\alpha \supset (\beta \supset \gamma)$	MP 1, 2
4. $(\alpha \supset (\beta \supset \gamma)) \supset ((\alpha \supset \beta) \supset (\alpha \supset \gamma))$	A2
5. $(\alpha \supset \beta) \supset (\alpha \supset \gamma)$	MP 3, 4
6. $\alpha \supset \beta$	P
7. $\alpha \supset \gamma$	MP 5, 6

Podemos abreviar las deducciones introduciendo teoremas ya demostrados:

Teorema 3. $\alpha, \beta \vdash \alpha \wedge \beta$

1. $(\alpha \supset \alpha) \supset ((\alpha \supset \beta) \supset (\alpha \supset \alpha \wedge \beta))$	A6
2. $\alpha \supset \alpha$	Teorema 1
3. $(\alpha \supset \beta) \supset (\alpha \supset \alpha \wedge \beta)$	MP 1, 2
4. $\beta \supset (\alpha \supset \beta)$	A 1
5. β	P
6. $\alpha \supset \beta$	MP 4, 5
7. $\alpha \supset (\alpha \wedge \beta)$	MP 3, 6
8. α	P
9. $\alpha \wedge \beta$	MP 7, 8

Note que la anterior no es una deducción formal en el sentido estricto, pues la línea 2 no es premisa ni axioma, ni resulta por MP. Sin embargo, se convierte en una deducción de $\alpha \supset \alpha$. Puesto que tal deducción ya se tiene, sería superfluo repetirla. En general, suponga que $\alpha_1, \dots, \alpha_r, \beta$ son esquemas de fórmulas y que se ha demostrado que $\alpha_1, \dots, \alpha_r \vdash \beta$; entonces podemos abreviar las deducciones utilizando, cuando ello convenga, la nueva regla de deducción que permite pasar directamente de $\alpha_1, \dots, \alpha_r$ a β ; esta será una *regla derivada*. Cualquier deducción en donde se utilicen reglas derivadas se podrá transformar en una deducción formal en el sentido estricto. Nos proporcionan reglas derivadas por ejemplo teorema 2, teorema 3 ya demos-

trados, teorema 1 puede considerarse como una regla sin premisas. A cada uno de los axiomas corresponde una o más reglas derivadas. por ejemplo, a A 9 corresponde : $\alpha \supset \gamma, \beta \supset \gamma \vdash \alpha \vee \beta \supset \gamma$.

1. $\alpha \supset \gamma$	P
2. $\beta \supset \gamma$	P
3. $(\alpha \supset \gamma) \supset ((\beta \supset \gamma) \supset (\alpha \vee \beta \supset \gamma))$	A 9
4. $(\beta \supset \gamma) \supset (\alpha \vee \beta \supset \gamma)$	MP 1, 3
5. $\alpha \vee \beta \supset \gamma$	MP 2, 4

Tenemos pues, de los axiomas (ejercicio) :

- Re1.* $\alpha \vdash \beta \supset \alpha$
- Re2.* $\alpha \supset (\beta \supset \gamma), \alpha \supset \beta \vdash \alpha \supset \gamma$
- Re3.* $\neg \alpha \supset \neg \beta \vdash \beta \supset \alpha$
- Re4. $\alpha \wedge \beta \vdash \alpha$
- Re5. $\alpha \wedge \beta \vdash \beta$
- Re6. $\alpha \supset \beta, \alpha \supset \gamma \vdash \alpha \supset \beta \wedge \gamma$
- Re7. $\alpha \vdash \alpha \vee \beta$
- Re8. $\beta \vdash \alpha \vee \beta$
- Re9. $\alpha \supset \gamma, \beta \supset \gamma \vdash \alpha \vee \beta \supset \gamma$

Los siguientes teoremas constituyen importantes reglas derivadas :

Teorema 4*. $\neg \alpha \vdash \alpha \supset \beta$

- 1. $\neg \alpha$ P
- 2. $\neg \beta \supset \neg \alpha$ Re 1, 1
- 3. $\alpha \supset \beta$ Re 3, 2

Teorema 5*. $\neg \neg \alpha \vdash \alpha$ (*Ley de la doble negación I*)

- 1. $\neg \neg \alpha$ P
- 2. $\neg \alpha \supset \neg \neg \neg \alpha$ Teorema 4, 1
- 3. $\neg \neg \alpha \supset \alpha$ Re3, 2
- 4. α MP 1, 3

Teorema 6. $\alpha \vee \beta$. (*Modus Tollendo Ponens*)

1. $\alpha \vee \beta$	P
2. $\neg \alpha$	P
3. $\alpha \supset \beta$	Teorema 4, 2
4. $\beta \supset \beta$	Teorema 1
5. $\alpha \vee \beta$	Re 9, 3, 4
6. β	MP 1, 5

Teorema 7*. $\alpha, \neg \alpha \vdash \beta$

1. $\neg \alpha$	P
2. $\alpha \supset \beta$	Teorema 4, 1
3. α	P
4. β	MP 2, 3

La última regla expresa el hecho de que de una contradicción se puede deducir “cualquier cosa”.

Ejercicios

1. Demuestre $\vdash \neg \alpha \vee \beta \Leftrightarrow \beta \wedge \alpha; \alpha \vee \beta, \neg \beta \vdash \alpha$.
2. Dé deducciones de no más de 5 líneas para Re1. a Re 9.
3. Dé deducciones de teorema 4 y teorema 5 usando sólo los axiomas y M.P.
4. ¿Por qué no es posible deducir $\alpha \supset \alpha$ usando sólo las reglas Re1. a Re9. y M.P.?

2.3 Teorema de la deducción

El siguiente teorema simplifica la tarea de construir deducciones formales. Nótese que no es un teorema *dentro* del sistema deductivo, sino un teorema *sobre* el sistema, que describe una de sus propiedades. Por eso lo llamamos *metateorema*.

Cuando se quiere deducir de un conjunto de premisas una consecuencia de la forma $\alpha \supset \beta$, el método natural consiste en “suponer α ” y tratar de deducir β , rara vez se intenta deducir $\alpha \supset \beta$ directamente de las premisas. El metateorema muestra que el método está totalmente justificado desde el punto de vista formal, y es además reducible a los 9 axiomas proposicionales y la regla MP.

Metateorema (Teorema de la Deducción). *Sea Γ un conjunto de premisas, posiblemente vacío. Si $\Gamma, \alpha \vdash \beta$ entonces $\Gamma \vdash \alpha \supset \beta$.*

Demostración. Suponga $\Gamma = \{\sigma_1, \dots, \sigma_r\}$ y sea β_1, \dots, β_n una deducción de β con premisas $\sigma_1, \dots, \sigma_r, \alpha$. Por inducción en k , $1 \leq k \leq n$, mostraremos que $\sigma_1, \dots, \sigma_r \vdash \alpha \supset \beta_k$ para cada k . Como $\beta_n = \beta$, tendremos el resultado haciendo $k = n$.

I. $k = 1$, hay dos posibilidades para β_1 .

Caso 1: β_1 es un axioma proposicional o pertenece a Γ . Entonces tenemos :

- | | |
|---|------------------------------|
| 1. β_1 | Axioma o Premisa en Γ |
| 2. $\beta_1 \supset (\alpha \supset \beta_1)$ | A 1 |
| 3. $\alpha \supset \beta_1$ | MP |

y por lo tanto $\Gamma \vdash \alpha \supset \beta_1$.

Caso 2: β_1 es α . Entonces $\vdash \alpha \supset \beta_1$ por teorema 1, y se tiene trivialmente $\Gamma \vdash \alpha \supset \beta_1$.

II. Supongamos cierto para $1 \leq k$ que $\Gamma \vdash \alpha \supset \beta_1$. Si β_k es un axioma proposicional o pertenece a $\Gamma \cup \{\alpha\}$ usamos el mismo argumento de (I) para concluir que $\Gamma \vdash \alpha \supset \beta_k$. La otra posibilidad

es que β_k resulte por MP de fórmulas β_s, β_t con $s, t \leq k$, entonces debemos tener $\beta_t = (\beta_s \supset \beta_k) \circ \beta_s = (\beta_t \supset \beta_k)$.

Suponga el primer caso (el otro es similar). Por hipótesis de inducción:

$$\Gamma \vdash \alpha \supset \beta_s \quad (1)$$

$$\Gamma \vdash \alpha \supset (\beta_s \supset \beta_k) \quad (2)$$

Si añadimos a las deducciones de (1) y (2) las líneas

$$(\alpha \supset (\beta_s \supset \beta_k)) \supset ((\alpha \supset \beta_s) \supset (\alpha \supset \beta_k)) \quad A2$$

$$(\alpha \supset \beta_s) \supset (\alpha \supset \beta_k) \quad MP, (2)$$

$$\alpha \supset \beta_k \quad MP, (1)$$

tenemos que $\Gamma \vdash \alpha \supset \beta_k$. ■

La demostración del Teorema de la Deducción es *constructiva*, en el sentido de que nos indica como transformar una deducción para $\Gamma, \alpha \vdash \beta$ en una deducción para $\Gamma \vdash \alpha \supset \beta$. Por ejemplo la deducción :

1. α	P
2. $\alpha \supset \beta$	P
3. β	MP 1, 2

que muestra $\alpha, \alpha \supset \beta \vdash \beta$ se puede transformar en una de $\alpha \vdash (\alpha \supset \beta) \supset \beta$ de la forma siguiente :

1 α	P
1' $\alpha \supset ((\alpha \supset \beta) \supset \alpha)$	A1
1'' $(\alpha \supset \beta) \supset \alpha$	MP 1, 1'
2 $(\alpha \supset \beta) \supset (\alpha \supset \beta)$	Teorema 1
3 $((\alpha \supset \beta) \supset (\alpha \supset \beta)) \supset [((\alpha \supset \beta) \supset \alpha) \supset ((\alpha \supset \beta) \supset \beta)]$	A2
3' $((\alpha \supset \beta) \supset \alpha) \supset ((\alpha \supset \beta) \supset \beta)$	MP 2, 3
3'' $(\alpha \supset \beta) \supset \beta$	MP 1'', 3'

El teorema de la deducción puede considerarse como un nuevo tipo de regla indirecta (TD). Para demostrar teoremas de la forma $\Gamma \vdash \alpha \supset \beta$, la estrategia

consiste en adjuntar α a las premisas, tratar de deducir β y en caso de lograrlo, invocar TD. Su poder reside en que al adjuntar la nueva premisa α se aumentan las posibilidades de aplicar reglas y avanzar en la deducción.

Ejemplo

El Teo 2 puede demostrarse trivialmente de esta forma, pues aplicando MP dos veces : $\alpha \supset \beta$, $\beta \supset \gamma$, $\alpha \vdash \gamma$, e invocando TD, tenemos : $\alpha \supset \beta$, $\beta \supset \gamma$ $\vdash \alpha \supset \gamma$. Si continuamos utilizando TD, podemos mostrar dos nuevos teoremas cuya deducción directa sería muy difícil:

- (a) $\alpha \supset \beta \vdash (\beta \supset \gamma) \supset (\alpha \supset \beta)$
- (b) $\vdash (\alpha \supset \beta) \supset ((\beta \supset \gamma) \supset (\alpha \supset \beta))$.

Podemos utilizar TD para continuar desarrollando nuestra lista de teoremas y reglas derivadas. Por ejemplo la ley de doble negación inversa a la del Teo 5. Primero necesitamos :

Teorema 8*. $\vdash \neg \neg \alpha \supset \alpha$, por TD, de Teorema 5.

Teorema 9*. $\alpha \vdash \neg \neg \alpha$ (*Ley de doble negación 2*)

- | | |
|--|-------|
| 1. α | P |
| 2. $\neg \neg \neg \alpha \supset \neg \alpha$ | Teo 8 |
| 3. $\alpha \supset \neg \neg \alpha$ | Re 3 |
| 4. $\neg \neg \alpha$ | MP |

Ejercicios

1. Demuestre $\vdash (\alpha \supset \beta) \supset ((\beta \supset \gamma) \supset (\alpha \supset \gamma))$ sin usar TD.
2. Demuestre:
 - a. $\alpha \wedge \beta \supset \gamma \vdash \alpha \supset (\beta \supset \gamma)$

- b. $\alpha \supset (\beta \supset \gamma) \vdash \alpha \vee \beta \supset \gamma$
c. $\vdash (\alpha \wedge \beta \supset \gamma) \Leftrightarrow (\alpha \supset (\beta \supset \gamma))$
d. $\vdash \neg \alpha \supset (\alpha \supset \beta)$
3. Demuestre :
- $\vdash \alpha \Leftrightarrow \neg \neg \alpha$
 - $p \Leftrightarrow q, s \supset t, (p \supset q) \supset (s \vee \neg t) \vdash ((q \supset p) \supset t) \Leftrightarrow s$
4. Es cierto que $\sigma_1, \dots, \sigma_n \vdash \alpha \supset \beta$ si y solo si $\sigma_1, \dots, \sigma_n, \alpha \vdash \beta$?
5. Demuestre : $\neg \alpha \vee \beta \vdash \neg \alpha \supset \beta$.
6. Demuestre : $\alpha \Leftrightarrow (\beta \Leftrightarrow \gamma) \vdash (\alpha \Leftrightarrow \beta) \supset \gamma$.
7. Demuestre : $\alpha \Leftrightarrow (\beta \Leftrightarrow \gamma) \vdash (\alpha \Leftrightarrow \beta) \supset \gamma$
8. Sea $\alpha_1 \vee \alpha_2 \vee \dots \vee \alpha_n = \alpha_1 \vee (\alpha_2 \vee (\dots (\alpha_{n-1} \vee \alpha_n) \dots))$.
Demuestre que si $\Gamma \vdash \alpha_1 \vee \dots \vee \alpha_n$ y $\Gamma, \alpha_i \vdash \beta$ para cada i entonces $\Gamma \vdash \beta$ (Use inducción en n).
9. Defina una familia $\alpha_1, \alpha_2, \dots$ de fórmulas de la manera siguiente :
 $\alpha_1 = p \supset p$
 $\alpha_{n+1} = ((\alpha_n) \supset p) \supset p$
Demuestre que para toda n se tiene $\vdash \alpha_n$ (Use inducción en n , MP, y TD).
10. Muestre que TD vale en cualquier sistema que contenga A1, A2 y cuya única regla de deducción sea MP.

2.4 Reducción al absurdo

Cuando se quiere demostrar que de Γ se sigue α en donde Γ es una colección de premisas (posiblemente vacía), muchas veces se dé una *prueba por contradicción*; es decir se supone $\neg \alpha$ como premisa adicional y se trata de llegar a una contradicción, este fue el método utilizado en nuestro ejemplo introductorio. El siguiente teorema muestra que el método vale en nuestro sistema formal y es por lo tanto una consecuencia de los axiomas proposicionales y MP.

Metateorema (Reducción al absurdo). Si $\Gamma, \neg \alpha \vdash \beta$ y $\Gamma, \neg \alpha \vdash \neg \beta$ entonces $\Gamma \vdash \alpha$.

Demostración. Suponga $\Gamma, \neg \alpha \vdash \beta, \neg \beta$; Por el Teo 7, $\beta, \neg \beta \vdash \neg (\beta \supset \beta)$. Entonces $\Gamma, \neg \alpha \vdash \neg (\beta \supset \beta)$, y por TD: $\Gamma \vdash \neg \alpha \supset \neg (\beta \supset \beta)$. De ahí se obtiene sucesivamente de Γ :

1. $\neg \alpha \supset \neg (\beta \supset \beta)$
2. $(\beta \supset \beta) \supset \alpha$ Re3
3. $\beta \supset \beta$ Teo 1
4. α MP 2, 3

Corolario. Si $\Gamma, \alpha \vdash \beta, \neg \beta$ entonces $\Gamma \vdash \neg \alpha$.

Demostración. Si $\Gamma, \alpha \vdash \beta, \neg \beta$ entonces $\Gamma, \neg \neg \alpha \vdash \Gamma, \alpha \vdash \beta, \neg \beta$ por Teo 5, y aplicando el Metateorema 2 se tiene $\Gamma \vdash \neg \alpha$.

El Metateorema y su corolario pueden tomarse como una regla deductiva indirecta, análoga a TD, la cual llamaremos RA. Los siguientes teoremas muestran su utilidad.

Teorema 10*. $\alpha \supset \beta, \neg \beta \vdash \neg \alpha$ (*Modus Tollendo Tollens*)

Demostración. $\alpha \supset \beta, \neg \beta, \alpha \vdash \neg \beta$ (Trivialmente)

$\alpha \supset \beta, \neg \beta, \alpha \vdash \beta$ (Por MP)

El resultado sigue de RA.

Teorema 11*. $\alpha \supset \beta, \neg \alpha \supset \beta \vdash \beta$

Demostración. Basta mostrar que $\alpha \supset \beta, \neg \alpha \supset \beta, \neg \beta \vdash \beta$ y aplicar RA:

- | | |
|--------------------------------|------------|
| 1. $\alpha \supset \beta$ | P |
| 2. $\neg \beta$ | P |
| 3. $\neg \alpha$ | Teorema 10 |
| 4. $\neg \alpha \supset \beta$ | P |
| 5. β | MP 3, 4 |

Teorema 12. $\vdash \alpha \vee \neg \alpha$ (*Ley del tercio excluso*)

- | | |
|--|------------|
| 1. $\alpha \supset \alpha \vee \neg \alpha$ | Ax 7 |
| 2. $\neg \alpha \supset \alpha \vee \neg \alpha$ | Ax 8 |
| 3. $\alpha \vee \neg \alpha$ | Teorema 11 |

Hemos podido, pues, deducir la ley del tercero excluido dentro del sistema formal, a pesar de que ésta parecía remota de los axiomas originales. Según lo que hemos mostrado, debe haber una prueba directa de $\alpha \vee \neg \alpha$ que sólo utilice los axiomas y Modus Ponens.

Terminamos con algunos teoremas (reglas derivadas) que necesitaremos más adelante.

Teorema 13*. $\alpha, \neg \beta \vdash \neg (\alpha \supset \beta)$

Demostración. Trivialmente $\alpha, \neg \beta, \alpha \supset \beta \vdash \beta, \neg \beta$. El resultado sigue por RA.

Teorema 14. $\neg \alpha, \neg \beta \vdash \neg (\alpha \vee \beta)$

Demostración. Por Teo 6 (MTP): $\neg \alpha, \alpha \vee \beta \vdash \beta$.

Entonces $\neg \alpha, \neg \beta, \alpha \vee \beta \vdash \beta, \neg \beta$ y el resultado sigue por RA.

Teorema 15. (a) $\neg \alpha \vdash \neg (\alpha \wedge \beta)$

(b) $\neg \beta \vdash \neg (\alpha \wedge \beta)$

Demostración. Sigue por RA de: $\neg \alpha, \alpha \wedge \beta \vdash \alpha, \neg \alpha$ y $\neg \beta, \alpha \wedge \beta \vdash \beta, \neg \beta$ (Reglas 4 y 5)

Ejercicios

1. Un conjunto Γ de fbf's se dice *inconsistente* si existe α tal que $\Gamma \vdash \alpha$ y $\Gamma \vdash \neg \alpha$, de lo contrario se llama *consistente*. Muestre:
 - Γ es inconsistente si y solamente si $\Gamma \vdash \beta$ para toda fbf β .
 - $\Gamma \not\vdash \alpha$ si y solamente si $\Gamma \cup \{ \neg \alpha \}$ es consistente.
2. $\vdash \neg(\alpha \wedge \neg \alpha)$.
3. $\vdash \neg(\alpha \vee \beta) \Leftrightarrow \neg \alpha \wedge \neg \beta$ (Leyes de De Morgan)
 $\vdash \neg(\alpha \wedge \beta) \Leftrightarrow \neg \alpha \vee \neg \beta$.
4. $\vdash \alpha \vee \beta \Leftrightarrow \neg \alpha \supset \beta$
 $\vdash \alpha \wedge \beta \Leftrightarrow \neg(\alpha \supset \neg \beta)$.
5. Demuestre $\Gamma \vdash \alpha \vee \beta$ si y solo si $\Gamma, \neg \alpha \vdash \beta$.
6. Demuestre:
 - $\neg p \vdash \neg(p \Leftrightarrow q) \Leftrightarrow q$
 - $(p \wedge q) \supset (\neg p \vee (s \supset t)), p, \neg t \vdash \neg q \supset \neg s$
 - $p \supset (\neg q \wedge r), q \vee s, (p \supset u) \supset \neg s, r \supset p \wedge t \vdash \neg q \supset t$
 - $p \supset (q \supset r), r \wedge s \supset t, \neg u \supset (s \wedge \neg t) \vdash p \supset (q \supset u)$
 - $(p \supset q) \wedge (r \supset s), s \wedge q \supset t, \neg t \vdash \neg p \wedge \neg r$
 - $p \supset (q \supset r), s \vee p, q \vdash s \vee r$
 - $(p \wedge \neg q) \supset \neg r, \neg r \supset q \vdash p \supset q$
 - $(p \wedge q) \vee (r \wedge s), p \supset \neg p \vdash r$
 - $p \supset r \vee s, s \supset \neg q \vdash p \supset (q \supset p)$.
7. Demuestre:
 - $\alpha \wedge (\beta \vee \gamma) \vdash (\alpha \wedge \beta) \vee (\gamma \wedge \gamma)$

- b. $(\alpha \wedge \beta) \vee (\alpha \wedge \gamma) \vdash \alpha \wedge (\beta \vee \gamma)$
- c. $\vdash \alpha \wedge (\beta \vee \gamma) \Leftrightarrow (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$
- d. $\vdash \alpha \vee (\beta \wedge \gamma) \Leftrightarrow (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$
- e. $\alpha \vee \beta, \neg \alpha \vee \beta \vdash \beta$
- f. $\alpha \vee \beta \vdash (\alpha \supset \beta) \supset \beta$
- g. $(\alpha \supset \beta) \supset \beta \vdash \alpha \vee \beta$
- h. $\alpha \supset (\beta \supset \gamma) \vdash \beta \supset (\alpha \supset \gamma)$

8. Simbolice los siguientes argumentos y demuestre que la conclusión se deduce formalmente de las premisas.
- a. Si la clase trabaja y el profesor lo nota, no les hará examen. Pero el profesor no lo notará, a menos que haga un examen. Por lo tanto, si la clase trabaja el profesor lo notará.
 - b. Si el banco no le dá el préstamo a Pérez, éste tendrá que obtener crédito en otra parte. Pero si el banco no le presta, no obtendrá crédito en ninguna parte, y si vende el negocio tendrá que irse a sembrar papas. Por lo tanto, si el banco no le presta, Pérez tendrá que irse a sembrar papas.
 - c. Si A está bien entrenado puede hacer los 100 m en 10 ss. Si A no está bien entrenado y no puede hacer los 100 m en 10 ss, entonces B estará en mejores condiciones. Si B está en mejores condiciones y A no hace los 10 m en 10 ss entonces B ganará y por lo tanto A no ganará. En conclusión : Si A gana, hará los 100 m en 10 ss.
 - d. Juan tiene 20 ó 22 años. Si Juan tiene 22 años entonces nació antes que Pedro. Juan no nació antes que Pedro. Luego Juan tiene 20 años.
 - e. X es par o impar. Si X es par entonces X^2 es par. Si X^2 es par entonces no es impar. Por lo tanto, si X^2 es impar entonces X es impar.
 - f. Si Juan viene a la fiesta, Luis no vendrá a menos que María venga, pero, María viene solo si Juan no viene. Por lo tanto, Luis no vendrá a la fiesta si Juan viene.
- 9.* Muestre que RA vale en cualquier sistema que contenga A1, A2, A3 y cuya única regla de deducción sea MP.

10. Demuestre que si reemplaza A3 por el esquema :

$$(\neg \alpha \supset \neg \beta) \supset ((\neg \alpha \supset \beta) \supset \alpha)$$

el sistema resultante es equivalente al sistema original (A1-A9 y MP), en el sentido de que en ambos sistemas se deducen exactamente los mismos teoremas (y las mismas consecuencias formales). Use el Ejercicio 11.

11*. Si α es una fórmula sea α' la fórmula que resulta de eliminar \vee y \wedge en α por medio de las sustituciones :

$$\alpha \vee \beta \rightarrowtail \neg \alpha \supset \beta; \alpha \wedge \beta \rightarrowtail \neg (\alpha \supset \neg \beta).$$

Demuestre que usando sólo A1, A2 y A3 se pueden deducir A4 a A9. Concluya que si $\sigma_1, \dots, \sigma_n \vdash \alpha$ entonces α puede deducirse de $\sigma_1, \dots, \sigma_n$ usando solamente A1, A2, A3.

12. Demuestre que los sistemas siguientes, que sólo consisten de reglas, son equivalentes al sistema de axiomas A1 a A9 y MP.

a. MP, TD y las reglas :

$$\frac{\neg \alpha \supset \neg \beta}{\beta \supset \alpha} \quad \frac{\alpha, \beta}{\alpha \wedge \beta} \quad \frac{\alpha \wedge \beta}{\alpha} \quad \frac{\alpha \wedge \beta}{\beta} \quad \frac{\alpha}{\alpha \vee \beta} \quad \frac{\beta}{\alpha \vee \beta} \quad \frac{\alpha \supset \gamma, \beta \supset \gamma}{\alpha \vee \beta \supset \gamma}$$

b. MP, TD y las reglas :

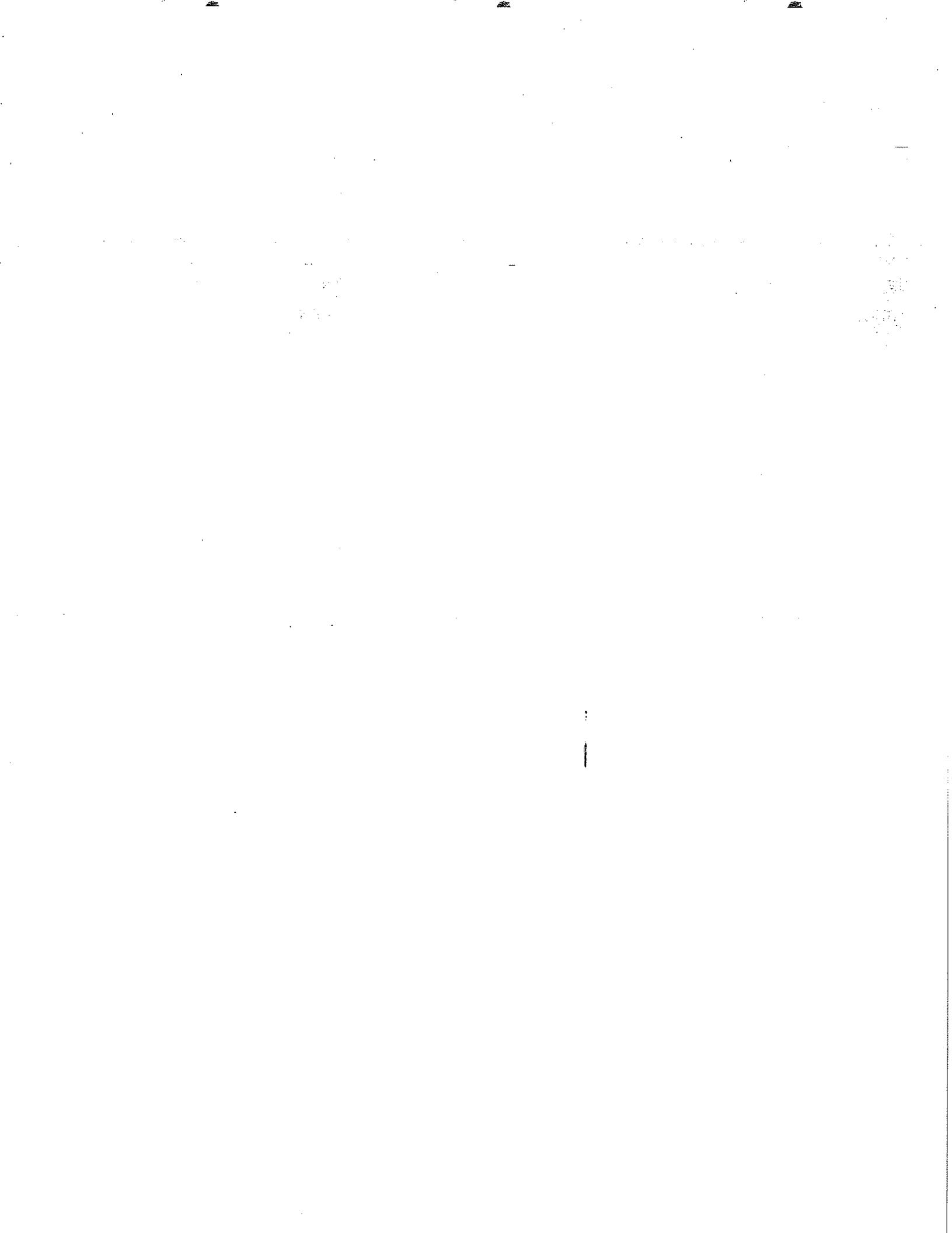
$$\frac{\alpha, \beta}{\alpha \wedge \beta} \quad \frac{\alpha \wedge \beta}{\alpha} \quad \frac{\alpha \wedge \beta}{\beta} \quad \frac{\alpha}{\alpha \vee \beta} \quad \frac{\beta}{\alpha \vee \beta} \quad \frac{\alpha \supset \gamma, \beta \supset \gamma}{\alpha \vee \beta \supset \gamma}$$

c. MP, TD y las reglas :

$$\begin{array}{cccccc} \frac{\alpha, \beta}{\alpha \wedge \beta} & \frac{\alpha \wedge \beta}{\alpha} & \frac{\alpha \wedge \beta}{\beta} & \frac{\alpha}{\alpha \vee \beta} & \frac{\beta}{\alpha \vee \beta} & \frac{\alpha \vee \beta, \neg \alpha}{\beta} \\ \\ \frac{\alpha \supset \beta, \neg \beta}{\neg \alpha} & \frac{\alpha}{\neg \neg \alpha} & \frac{\neg \neg \alpha}{\alpha} & & & \end{array}$$

13. Demuestre :

- a. $\alpha \leftrightarrow \beta \vdash \neg \alpha \leftrightarrow \neg \beta$
- b. $\alpha \leftrightarrow \beta \vdash (\alpha \square \gamma) \leftrightarrow (\beta \square \gamma)$ para todo conectivo binario γ .
- c. Si γ' es el resultado de substituir una subfórmula α de γ por β , demuestre que $\alpha \leftrightarrow \beta \vdash \gamma \leftrightarrow \gamma'$.
(Use inducción en la fórmula γ).



Capítulo 3

Semántica

El sistema deductivo que hemos desarrollado es puramente sintáctico. El ser un teorema o una consecuencia formal de ciertas premisas es una propiedad que depende solamente de la forma y las relaciones estructurales entre fórmulas. Por otra parte, el análisis de los posibles significados y consiguiente veracidad o falsedad que las fórmulas puedan tener, aunque no juega ningún papel en la deducción formal, parece ser su justificación intuitiva. Comenzaremos ahora el estudio del Cálculo de Proposiciones desde este punto de vista, que corresponde a la *Semántica*. Intuitivamente, la validez de un argumento o de un principio lógico consiste en el hecho de siempre es verdadero, no importa que significado se dé a sus componentes proposicionales. En primer lugar debemos precisar, pues, qué queremos decir con qué una fbf sea “verdadera” o “falsa”. Dada una fórmula bien formada α su significado depende del que le demos a sus letras proposicionales p_1, \dots, p_n . Para averiguar, por ejemplo, si es “siempre verdadera” deberíamos entonces examinar todos los posibles significados de p_1, \dots, p_n y en cada caso averiguar el de α y su consiguiente verdad o falsedad, una labor imposible por muchas razones. Entre otras: los posibles significados son infinitos en número, y una vez especificado un significado para α puede éste ser tan complejo, ambiguo, paradójico, o de impracticable verificación que no podamos determinar su verdad o falsedad.

Afortunadamente, no es necesario hacer el anterior examen. El que α sea verdadera o falsa depende solamente de que p_1, \dots, p_n sean cada una verdadera o falsa. Basta pues examinar las posibles situaciones de las letras proposi-

cionales con respecto a verdad y falsedad para conocer el comportamiento de α con respecto a verdad y falsedad en cualquier situación posible, y estas combinaciones son finitas.

Las consideraciones anteriores nos llevan a las siguientes definiciones.

3.1 Valuaciones, validez

Si P es un conjunto de letras proposicionales, $\mathfrak{I}(P)$ es el conjunto de fbf's cuyas letras proposicionales están en P . Sea $\{V, F\}$ un conjunto de dos elementos distintos que llamamos *valores de verdad*.

Definición 1. Una *valuación* de P es una función $v: P \rightarrow \{V, F\}$.

Definición 2. Cada valuación v de P puede extenderse a todo $\mathfrak{I}(P)$ de la manera siguiente, donde llamamos $\bar{v}: \mathfrak{I}(P) \rightarrow \{V, F\}$ a la función extendida:

$$1. \bar{v}[\alpha] = v[\alpha] \text{ si } \alpha \in P$$

$$2. \bar{v}[\neg\alpha] = \begin{cases} V & \text{si } \bar{v}[\alpha] = F \\ F & \text{si } \bar{v}[\alpha] = V \end{cases}$$

$$3. \bar{v}[\alpha \wedge \beta] = \begin{cases} V & \text{si } \bar{v}[\alpha] = \bar{v}[\beta] = V \\ F & \text{de lo contrario} \end{cases}$$

$$4. \bar{v}[\alpha \vee \beta] = \begin{cases} F & \text{si } \bar{v}[\alpha] = \bar{v}[\beta] = F \\ V & \text{de lo contrario} \end{cases}$$

$$5. \bar{v}[\alpha \supset \beta] = \begin{cases} F & \text{si } \bar{v}[\alpha] = V \text{ y } \bar{v}[\beta] = F \\ V & \text{de lo contrario} \end{cases}$$

La anterior es una definición recursiva en el sentido de que permite calcular $\bar{v}[\alpha]$ en términos de los valores de \bar{v} en las subfórmulas inmediatas de α .

Por el teorema de la Descomposición Unica, las partes recursivas (3), (4) y (5) de la definición no son ambiguas. Por el principio de inducción en fórmulas, \bar{v} está definida en todo $\mathfrak{I}(P)$. Se acostumbra presentar las partes (2) a (5) de la definición en forma de *tablas de verdad*:

α	$\neg\alpha$
V	F
F	V
(2)	

α	β	$\alpha \wedge \beta$	$\alpha \vee \beta$	$\alpha \supset \beta$
V	V	V	V	V
V	F	F	V	F
F	V	F	V	V
F	F	F	V	V
		(3)	(4)	(5)

Si interpretamos V como “verdadero” y F como “falso”, el significado asignado a \neg y \wedge corresponde al uso ordinario de la negación y la conjunción de proposiciones. La tabla de \vee corresponde al uso del “ó” como disyunción no-exclusiva (α ó β , ó ambos). Sobre la tabla de \supset ha habido controversias que se remontan a varios siglos. El problema radica en que, en el uso ordinario del lenguaje, hay circunstancias en las cuales no es claro cuál valor de verdad asignarle a $\alpha \supset \beta$ cuando α es falso. Además, aún en el caso en que α y β sean ambas verdaderas se argumenta que $\alpha \supset \beta$ no debe ser verdadera, a no ser que exista una conexión de causa-efecto entre α y β . Ha habido intentos de formalizar estos puntos de vista por medio de cálculos proposicionales de más de dos valores de verdad, o con varios tipos de implicaciones. El hecho es que si queremos trabajar en un sistema de *dos* valores donde el valor de $\alpha \supset \beta$ esté siempre definido, la única interpretación razonable para \supset es la tabla (5).

Si P es finito, $P = \{p_1, \dots, p_n\}$, el conjunto de todas las valuaciones de P se puede representar por una tabla. Basta identificar cada $v: P \rightarrow \{V, F\}$ con la n -tupla $(v(p_1), \dots, v(p_n))$. Es claro entonces que el número de valuaciones de P es 2^n . Si $\alpha \in \mathfrak{I}(P)$, todos los posibles valores $\bar{v}[\alpha]$ pueden calcularse con la ayuda de esta tabla, calculando consecutivamente los valores de las subfórmulas de α para cada n -tupla. Por ejemplo, si $P = \{p_1, p_2, p_3\}$ y $\alpha = p_1 \supset (p_2 \vee \neg p_1)$, tenemos

p_1	p_2	p_3	p_1	\neg	$(p_2 \vee \neg p_1)$
V	V	V		V	V F
V	V	F		V	V F
V	F	V		F	F F
V	F	F		F	F F
F	V	V		V	V V
F	V	F		V	V V
F	F	V		V	V V
F	F	F		V	V V

Los valores de cada subfórmula aparecen debajo de su conectivo principal. En el ejemplo, es claro que $\bar{v}[\alpha]$ es el mismo, no importa cuál sea V (p_3). Esto es consecuencia del siguiente resultado.

Lema 1. Sea $\alpha \in \mathfrak{F}(P)$ y sean p_1, \dots, p_n las letras proposicionales que ocurren en α . Si v y w son valuaciones que coinciden en $\{p_1, \dots, p_n\}$ entonces $\bar{v}[\alpha] = \bar{w}[\alpha]$.

Demostración. Por inducción en fórmulas. Si α es letra proposicional, es obvio pues $v[\alpha] = \bar{v}[\alpha] = w[\alpha] = \bar{w}[\alpha]$. Suponga que vale para α y β y sean P' , P'' los conjuntos de letras proposicionales de α y β , respectivamente.

- (a) Si v y w coinciden en las letras de $\neg\alpha$, coinciden en P' . Por hipótesis de inducción, $\bar{v}[\alpha] = \bar{w}[\alpha]$ y por tanto $\bar{v}[\neg\alpha] = \bar{w}[\neg\alpha]$.
- (b) Si v y w coinciden en las letras de $\alpha \square \beta$ entonces coinciden en $P' \cup P''$ y a fortiori en P' y P'' . Por hipótesis de inducción, $\bar{v}[\alpha] = \bar{w}[\alpha]$ y $\bar{v}[\beta] = \bar{w}[\beta]$. Por lo tanto $\bar{v}[\alpha \square \beta] = \bar{w}[\alpha \square \beta]$. ■

Corolario. Sea $\alpha \in \mathfrak{F}(P)$, $P \subseteq P'$, v una valuación de P y w una valuación de P' que extiende a v , entonces $\bar{v}[\alpha] = \bar{w}[\alpha]$.

En adelante, si escribimos $\bar{v}[\alpha]$ asumimos que v está definida en un conjunto P que contiene las letras proposicionales de α . Por el corolario anterior, las definiciones siguientes son independientes de P .

Definición 3. α es *tautología*, $\models \alpha$, si $\bar{v}[\alpha] = V$ para toda valuación (definida en sus letras proposicionales).

Definición 4. α es una *consecuencia válida* de $\sigma_1, \dots, \sigma_n$ si para toda valuación v tal que $\bar{v}[\sigma_1] = \dots = \bar{v}[\sigma_n] = V$, se tiene $\bar{v}[\alpha] = V$. Este hecho se denota por $\sigma_1, \dots, \sigma_n \models \alpha$.

Lema 2. $\sigma_1, \dots, \sigma_n \models \alpha$ si y sólo si $\models \sigma_1 \supset (\sigma_2 \supset \dots \supset (\sigma_n \supset \alpha))$

Demuestração. Suponga $\sigma_1, \dots, \sigma_n \models \alpha$ y sea v una valuación cualquiera; si $\bar{v}[\sigma_1 \supset (\sigma_2 \supset \dots \supset \alpha)] = F$ deberíamos tener $\bar{v}[\sigma_1] = V$ y $\bar{v}[\sigma_2 \supset \dots \supset (\sigma_n \supset \alpha)] = F$. Continuando de este manera, obtenemos que $\bar{v}[\sigma_1] = \dots = \bar{v}[\sigma_n] = V$ y $\bar{v}[\alpha] = F$, lo cual contradice la hipótesis. Por tanto $\bar{v}[\sigma_1 \supset (\sigma_2 \supset \dots \supset (\sigma_n \supset \alpha))] = V$. Conversamente, si $\models \sigma_1 \supset (\sigma_2 \supset \dots \supset \alpha)$ y $\bar{v}[\sigma_1] = \dots = \bar{v}[\sigma_n] = V$, debemos tener $\bar{v}[\alpha] = V$, de lo contrario tendríamos sucesivamente $\bar{v}[\sigma_n \supset \alpha] = F$, $\bar{v}[\sigma_{n-1} \supset (\sigma_n \supset \alpha)] = F, \dots, \bar{v}[\sigma_1 \supset (\sigma_2 \supset \dots \supset (\sigma_n \supset \alpha))] = F$. ■

Lema 3. Cada uno de los axiomas proposicionales es una tautología.

Demuestração. Lo hacemos para A2 y A9, el resto queda como ejercicio. En cada caso basta convencerse de que las tablas siguientes, donde las rayas significan valores arbitrarios, incluyen todas las posibilidades para $\bar{v}[\alpha]$, $\bar{v}[\beta]$, $\bar{v}[\gamma]$.

α	β	γ	$(\alpha \supset (\beta \supset \gamma)) \supset ((\alpha \supset \beta) \supset (\alpha \supset \gamma))$
-	-	V	
V	V	F	V F V F F
V	F	F	
F	-	F	

α	β	γ	$(\alpha \supset \gamma) \supset ((\beta \supset \gamma) \supset (\alpha \vee \beta \supset \gamma))$
-	-	V	V F F
V	-	F	F V
F	-	-	

Teorema de validez (Soundness). Si $\vdash \alpha$ entonces $\models \alpha$.

Demostración. Todo axioma lógico es tautología por el lema 3. Además MP preserva la propiedad de ser tautología porque si $\models \alpha$ y $\models \alpha \supset \beta$, para cualquier v debemos tener $\bar{v}[\alpha] = \bar{v}[\alpha \supset \beta] = V$, lo cual hace imposible $\bar{v}[\beta] = F$. El teorema sigue por inducción en la longitud de las deducciones. ■

Corolario. Si $\sigma_1, \dots, \sigma_n \vdash \alpha$ entonces $\sigma_1, \dots, \sigma_n \models \alpha$.

Demostración. Si $\sigma_1, \dots, \sigma_n \vdash \alpha$ entonces $\vdash \sigma_1 \supset (\sigma_2 \supset (\dots \supset (\sigma_n \supset \alpha) \dots))$ por TD aplicado n veces. Por tanto $\models \sigma_1 \supset (\dots \supset (\sigma_n \supset \alpha))$ y por el lema 2: $\sigma_1, \dots, \sigma_n \models \alpha$. ■

Ejercicios

1. ¿Cuántas valuaciones tiene un conjunto infinito enumerable de letras proposicionales?
2. Demuestre que Ax 1 y Ax 3 - Ax 8 son tautologías.
3. Demuestre que el sistema deductivo es *consistente*, es decir no existe fórmula α tal que $\vdash \alpha$ y $\vdash \neg \alpha$.
4. Sabiendo que $\alpha \leftrightarrow \beta$ es una abreviación, ¿cuál debe ser el valor de $\bar{v}[\alpha \leftrightarrow \beta]$ en función de los valores $\bar{v}[\alpha]$ y $\bar{v}[\beta]$?
5. Demuestre que de las fórmulas $(r \vee q) \supset p$, $(\neg r \wedge p) \supset q$, $(p \wedge q) \supset r$ no se puede deducir $(p \supset q) \supset r$.
6. ¿Son ciertas las afirmaciones siguientes?
 - a. Para todo α se tiene $\vdash \alpha$ ó $\vdash \neg \alpha$.
 - b. Si $\vdash \alpha \vee \beta$ entonces $\vdash \alpha$ ó $\vdash \beta$.
 - c. $\vdash \alpha \wedge \beta$ si y solo si $\vdash \alpha$ y $\vdash \beta$.
 - d. Si $\bar{v}(\alpha) = F$ siempre que $\bar{v}(\beta) = F$, entonces $\beta \vdash \alpha$.

e. Si $\bar{v}(\alpha) = F$ siempre que $\bar{v}(\beta) = F$, entonces $\alpha \models \beta$.

7. Bajo qué valuaciones de p, q, r, s, t, u , es verdadera la fórmula siguiente?

$$\neg(t \supset (\neg p \supset r)) \wedge (((r \vee s) \leftrightarrow ((u \supset q) \supset t)) \wedge (q \supset s)).$$

8. ¿Cuáles son tautologías?

- a. $p \supset ((\neg(r \wedge s) \vee (t \leftrightarrow (v \supset \neg w \vee q))) \vee (r \vee w)) \supset p$
- b. $(p \supset q) \supset p$
- c. $(p \supset q) \supset (\neg p \supset \neg q)$
- d. $\neg(\neg(\neg(\neg p \vee p) \vee p) \vee p) \vee p$
- e. $(p \supset q) \vee (q \supset p)$
- f. $(p \supset (q \vee r)) \supset ((p \supset q) \vee (p \supset r))$
- g. $((p \supset q) \vee (\neg p \wedge \neg q)) \vee \neg(p \leftrightarrow q)$
- h. $((p \supset q) \wedge p) \leftrightarrow q$
- i. $((\neg q \supset \neg r) \vee p) \supset \neg(p \wedge q \supset r)$

9. Si α y β son fórmulas, p una letra proposicional entonces $\alpha(p / \beta)$ es el resultado de substituir todas las ocurrencias de p en α por β . Demuestre que si $\models \alpha$ entonces $\models \alpha(p / \beta)$.

10. a. Demuestre que todas las fórmulas de la forma

$$\begin{aligned} & p \supset p \\ & ((p \supset p) \supset p) \supset p \\ & (((p \supset p) \supset p) \supset p) \supset p \\ & \text{etc.} \end{aligned}$$

con un número impar de tales conectivos, son tautologías.

b. Demuestre que las fórmulas

$$\begin{aligned} & p \\ & (p \supset p) \supset p \\ & (((p \supset p) \supset p) \supset p) \supset p \\ & \text{etc.} \end{aligned}$$

con un número par de tales conectivos, no son tautologías.

11. α es contingente si existen valuaciones v , w tales que $\bar{v}[\alpha] = V$ y $\bar{w}[\alpha] = F$. Demuestre por inducción en fórmulas que si cada letra proposicional de α ocurre sólo una vez en α , entonces α es contingente y por lo tanto no puede ser tautología.
- 12.* Demuestre que una fórmula que contiene solamente el conectivo \leftrightarrow es una tautología, si solo si cada letra proposicional ocurre un número par de veces.
13. Simbolice los siguientes argumentos y demuestre que *no* son válidos, es decir, la conclusión no es una consecuencia válida de las premisas.
- Si salgo al patio y está lloviendo entonces me mojo. No está lloviendo. Por lo tanto, si salgo al patio no me mojo.
 - x es par o impar. Si x es par entonces x^2 es par. Si x^2 es par entonces no es impar. Por lo tanto si x^2 es par entonces x es par.
 - Si puedo comprar un carro nuevo o arreglar el viejo, iré a Venezuela y pasaré por Bucaramanga. Si paso por Bucaramanga visitaré a mis amigos. Pero si los visito tendrá que quedarme con ellos, y si me quedo con ellos no podré visitar Venezuela. En conclusión, no podré visitar Venezuela.
14. Un viajero se encuentra en la bifurcación de dos caminos, uno lleva al cielo, el otro al infierno. Hay dos guardas, uno de los cuales dice siempre la verdad y el otro dice siempre mentiras. ¿Qué pregunta debe dirigir el viajero a *uno sólo* de los guardas, para que de su respuesta pueda deducir cuál es el camino al cielo? El viajero no sabe cuál es el camino, ni cuál es el guarda veraz.

3.2 Completitud

Habiendo visto que en el sistema deductivo solo se deducen tautologías, cabe preguntar si se pueden deducir *todas* las tautologías. Un sistema con tal propiedad es *deductivamente completo*. Mostraremos que nuestro siste-

ma es completo. Más aún, suponiendo $\sigma_1, \dots, \sigma_n \models \alpha$ veremos que debe haber una deducción formal que lleve de las premisas a la conclusión. Antes necesitamos una definición técnica.

Definición. Sea v una valuación fija, entonces definimos para cada fórmula α :

$$\alpha^v = \begin{cases} \alpha & \text{si } v[\alpha] = V \\ \neg\alpha & \text{si } v[\alpha] = F \end{cases}$$

Lema 1. (a) $\alpha^v \vdash (\neg\alpha)^v$

- (b) $\alpha^v, \beta^v \vdash (\alpha \supset \beta)^v$
- (c) $\alpha^v, \beta^v \vdash (\alpha \wedge \beta)^v$
- (d) $\alpha^v, \beta^v \vdash (\alpha \vee \beta)^v$

Demostración.

(a) Consideramos los casos $v[\alpha] = V, v[\alpha] = F$ en la tabla siguiente:

α	β	$\alpha^v \vdash (\neg\alpha)^v$		
V	F	α	$\neg\neg\alpha$	Teo 9
F	V	$\neg\alpha$	$\neg\alpha$	Trivial

(b) Consideramos las cuatro posibilidades para $v[\alpha]$ y $v[\beta]$ e indicamos el teorema o regla formal que justifica la afirmación en cada caso.

α	β	$\alpha^v, \beta^v \vdash (\alpha \supset \beta)^v$	$(\alpha \wedge \beta)^v$	$(\alpha \vee \beta)^v$
V	V	α, β	$\alpha \supset \beta$	Re1
V	F	$\alpha, \neg\beta$	$\neg(\alpha \supset \beta)$	Teo 13
F	V	$\neg\alpha, \beta$	$\alpha \supset \beta$	Re1
F	F	$\neg\alpha, \neg\beta$	$\alpha \supset \beta$	Teo 4
			$\neg(\alpha \wedge \beta)$	Teo 3
			"	Re7
			$\neg(\alpha \wedge \beta)$	Teo 15
			"	Re7
			$\neg(\alpha \wedge \beta)$	"
			$\neg(\alpha \wedge \beta)$	Re8
			"	Teo 14

Lema 2. Sean p_1, p_2, \dots, p_n las letras proposicionales de α y sea v una valuación entonces $p_1^v, \dots, p_n^v \vdash \alpha^v$.

Demostración. Usamos inducción en la complejidad de α .

I. $\alpha = p$ (letra proposicional) entonces la afirmación se reduce a $p^v \vdash p^v$.

II. Supóngalo cierto para α y β . Lo demostramos para $\neg\alpha$ y $\alpha \square \beta$.

Sean $p_1, \dots, p_k, q_1, \dots, q_L$ las variables proposicionales de α y $q_1, \dots, q_L, r_1, \dots, r_m$ las de β , en donde q_1, \dots, q_L denotan las variables comunes. Sea v una valuación, por hipótesis de inducción tenemos

$$p_1^v, \dots, p_k^v, q_1^v, \dots, q_L^v \vdash \alpha^v$$

$$q_1^v, \dots, q_L^v, r_1^v, \dots, r_m^v \vdash \beta^v$$

Pero $\alpha^v \vdash (\neg\alpha)^v$ y $\alpha^v, \beta^v \vdash (\alpha \square \beta)^v$ por el lema 1. Entonces : $p_1^v, \dots, p_k^v, q_1^v, \dots, q_L^v, r_1^v, \dots, r_m^v \vdash (\neg\alpha)^v, (\alpha \square \beta)^v$. ■

Teorema de completitud (Emil Post 1921). Si $\models \alpha$ entonces $\vdash \alpha$.

Demostración. Sea α una tautología, entonces $\bar{v}[\alpha] = V$ y por lo tanto $\alpha^v = \alpha$, para todo v . Por el lema 2 tenemos que para cualquier valuación de las letras proposicionales p_1, \dots, p_n de α ,

$$p_1^v, \dots, p_n^v \vdash \alpha \tag{1}$$

Sea v una valuación de p_2, \dots, p_n . Se puede extender a p_1, p_2, \dots, p_n de dos maneras : $v(p_1) = V$ en cuyo caso $p_1^v = p_1$ y tenemos por (1)

$$p_1, p_2^v, \dots, p_n^v \vdash \alpha,$$

o $v(p_1) = F$ en cuyo caso $p_1^v = \neg p_1$ y tenemos

$$\neg p_1, p_2^v, \dots, p_n^v \vdash \alpha.$$

Por lo tanto, $p_2^v, \dots, p_n^v \vdash p_1 \supset \alpha, \neg p_1 \supset \alpha$; y por teorema 11 tenemos que para cualquier v :

$$p_2^v, \dots, p_n^v \vdash \alpha.$$

Si repetimos este proceso n veces podemos eliminar todas las premisas hasta obtener $\vdash \alpha$. ■

Corolario. Dadas $\sigma_1, \sigma_2, \dots, \sigma_n \models \alpha$ entonces $\sigma_1, \dots, \sigma_n \vdash \alpha$.

Demostración. Si $\sigma_1, \dots, \sigma_n \models \alpha$ entonces $\vdash \sigma_1 \supset (\sigma_2 \supset \dots (\sigma_n \supset \alpha))$. Por el Teorema de Completitud: $\vdash \sigma_1 \supset (\sigma_2 \supset \dots (\sigma_n \supset \alpha))$. Con este hecho y usando MP n veces obtenemos $\sigma_1, \dots, \sigma_n \vdash \alpha$ ■

Corolario. Dadas $\sigma_1, \sigma_2, \dots, \sigma_n$ existe un algoritmo para determinar si para una fórmula arbitraria α se tiene $\sigma_1, \sigma_2, \dots, \sigma_n \vdash \alpha$ ó $\sigma_1, \sigma_2, \dots, \sigma_n \not\vdash \alpha$.

Demostración. Por los teoremas de Validez y Completitud, basta construir la tabla de verdad de $\sigma_1 \supset (\sigma_2 \supset \dots (\sigma_n \supset \alpha))$ y determinar si esta fórmula es o no una tautología. ■

El algoritmo anterior no es práctico si el número de letras proposicionales que ocurren en $\sigma_1, \dots, \sigma_n$, α es muy alto; para 10 letras proposicionales el número de valuaciones que hay que considerar es $2^{10} = 1024$. De ahí la importancia de los métodos puramente deductivos.

Definición. Una colección Γ de fórmulas es *consistente* si no existe α tal que $\Gamma \vdash \alpha$ y $\Gamma \vdash \neg \alpha$, de lo contrario Γ es *inconsistente*

Teorema de la consistencia. Si $\Gamma = \{\sigma_1, \dots, \sigma_n\}$ es consistente entonces existe una valuación v tal que $v[\sigma_1] = v[\sigma_2] = \dots = v[\sigma_n] = V$.

Demostración. Si no existe tal valuación entonces para cualquier fórmula α tenemos trivialmente $\sigma_1, \dots, \sigma_n \models \alpha$. Por completitud, $\sigma_1, \dots, \sigma_n \vdash \alpha$. Similarmente $\sigma_1, \dots, \sigma_n \vdash \neg \alpha$. Por lo tanto Γ es inconsistente. ■

Ejercicios

1. Si se considera \leftrightarrow como símbolo primitivo con el significado:

$$\bar{v}[\alpha \leftrightarrow \beta] = V \text{ si y solo si } \bar{v}[\alpha] = \bar{v}[\beta],$$

entonces basta añadir los esquemas axiomáticos

$$A\ 10\ (\alpha \leftrightarrow \beta) \supset (\alpha \supset \beta)$$

$$A\ 11\ (\alpha \leftrightarrow \beta) \supset (\beta \supset \alpha)$$

$$A\ 12\ (\alpha \supset \beta) \supset ((\beta \supset \alpha) \supset (\alpha \leftrightarrow \beta))$$

para tener un sistema deductivo completo. (Ayuda: basta mostrar $\alpha^v, \beta^v \vdash (\alpha \leftrightarrow \beta)^v$.)

2. Si sólo se consideran las fórmulas formadas con los conectivos \neg y \supset , entonces A1, A2, A3 y MP constituyen un sistema deductivo completo.

3. Demuestre sin hallar una deducción formal:

a. $\vdash (p_1 \wedge p_2) \vee ((p_1 \wedge \neg p_2) \vee ((\neg p_1 \wedge p_2) \vee (\neg p_1 \wedge \neg p_2)))$

b. $p \supset (p \supset (p \supset (\neg p \supset r \vee s)))$

4. ¿Vale el converso del Teorema de la Consistencia?

5. Usando la idea de la demostración del teorema de la completitud deduzca formalmente las tautologías

a. $(p_1 \wedge p_2) \vee \neg(p_1 \vee p_2) \vee \neg(p_1 \supset p_2) \vee \neg(p_2 \supset p_1)$

b. $(p \supset q) \vee (q \supset p)$

6. Demuestre que $\Gamma \not\vdash \alpha$ si y sólo si existe una valuación que hace verdaderas las fórmulas de $\Gamma \cup \{\neg \alpha\}$

7. Considere el sistema deductivo que resulta de reemplazar A 3 por :

$$A3I. (\alpha \supset \beta) \supset (\neg \beta \supset \neg \alpha).$$

Denotamos por \vdash_{\perp} la deducibilidad en este sistema. Si α es una fbf, sea α_I la fórmula que resulta de reemplazar todas las ocurrencias de " \neg " en α de acuerdo con la sustitución

$$\neg\beta \vdash_{\perp} \neg(q_o \supset \beta)$$

en donde q_o es una letra proposicional fija. Por ejemplo, $[\neg((p \wedge \neg q) \supset \neg(r \vee s))]_I = \neg(q_o \supset ((p \wedge \neg(q_o \supset q)) \supset \neg(q_o \supset r \vee s)))$

- a. Demuestre que si $\vdash_{\perp} \text{subI } \alpha$ entonces $\models \alpha_I$.
 - b. Demuestre que si $\not\vdash_{\perp} p \vee \neg p$, $\not\vdash_{\perp} p \supset \neg \neg p$.
8. Considere el sistema deductivo que resulta de reemplazar A3 por el par de axiomas:

$$\begin{cases} \neg \neg \alpha \supset \alpha \\ \alpha \supset \neg \neg \alpha \end{cases}$$

usamos \Vdash_{\perp} para indicar deducibilidad en el nuevo sistema. Si α es una fbf, sea α^o la fórmula que resulta de borrar todas las negaciones de α .

- a. Demuestre que si $\Vdash_{\perp} \alpha$ entonces $\models \alpha^o$.
 - b. Demuestre que $\Vdash_{\perp} (p \supset q) \supset (\neg q \supset \neg p)$, $\Vdash_{\perp} p \vee \neg p$.
9. Demuestre que el sistema deductivo que resulta de reemplazar A3 por los tres axiomas :
- A3.1 $(\alpha \supset \beta) \supset (\neg \beta \supset \neg \alpha)$
A3.2 $\alpha \supset \neg \neg \alpha$
A3.3 $\neg \neg \alpha \supset \alpha$
- es completo, pero no lo es si se suprime A3.2 o A3.3.
- 10*. Demuestre que el sistema del ejercicio 7 no es deducible $\neg \neg p \supset p$ (Use la transformación: $\neg \beta \vdash_{\perp} (\beta \supset q_o)$)

11.** Demuestre que el sistema deductivo que resulta de reemplazar A3 por los tres axiomas:

$$A3.1 (\alpha \supset \beta) \supset (\neg \beta \supset \neg \alpha)$$

$$A3.2 \alpha \supset \neg \neg \alpha$$

a. Demuestre que en este sistema vale la regla:

“Si $\Gamma, \alpha \vdash \beta, \neg \beta$ entonces $\Gamma \vdash \neg \alpha$ ” (RA débil)

b. Demuestre que no vale:

“Si $\Gamma, \neg \alpha \vdash \beta, \neg \beta$ entonces $\Gamma \vdash \alpha$.” (RA fuerte).

c. Muestre que ni $\neg \neg p \supset p$ ni $p \supset (\neg p \supset p)$ son deducibles en este sistema.

12. Sea \vdash^* la deducibilidad en un sistema deductivo en el cual valen MP, TD y las reglas:

$$1. \alpha \supset \beta \vdash^* \neg \beta \supset \neg \alpha$$

$$2. \alpha \wedge \beta \vdash^* \alpha, \beta$$

$$3. \alpha, \beta \vdash^* \alpha \wedge \beta$$

$$4. \alpha \vdash^* \alpha \vee \beta$$

$$5. \beta \vdash^* \alpha \vee \beta$$

$$6. \alpha \supset \gamma, \beta \supset \gamma \vdash^* \alpha \vee \beta \supset \gamma$$

Defina $\alpha \vdash^* \beta$ si y solo si $\alpha \vdash^* \beta$ y $\beta \vdash^* \alpha$.

Dadas fórmulas α, β y γ tales que β es una subfórmula de α , sea $\alpha.\text{sub}\beta\gamma$ el resultado de substituir *algunas* ocurrencias de β en α por γ . Demuestre: Si $\beta \vdash^* \gamma$ entonces $\alpha \vdash^* \alpha(\beta/\alpha)\gamma$.

13. Sea \vdash^* la deducibilidad en un sistema cualquiera.

$\sigma_1, \dots, \sigma_n$ son \vdash^* -independientes si para cada i , $\sigma_1, \dots, \sigma_{i-1}, \sigma_{i+1}, \dots, \sigma_n \not\vdash^* \sigma_i$.

$\{\sigma_1, \dots, \sigma_n\}$ y $\{\tau_1, \dots, \tau_k\}$ son \vdash^* -equivalentes si $\sigma_1, \dots, \sigma_n \vdash^* \tau_i : y \tau_1, \dots, \tau_k \vdash^* \sigma_j$ para toda i, j .

Demuestre que para todo conjunto finito S de fórmulas existe $S' \subseteq S$ tal que S' es \vdash^* -independiente y \vdash^* -equivalente a S .

14. Demuestre que existe un conjunto infinito S para el cual ningún subconjunto S' es \vdash -equivalente a S .
- 15.* Sea α una fórmula tal que $\not\vdash \alpha$. Demuestre que si se añade α como esquema axiomático al sistema deductivo Ax 1 a Ax 9, mas MP, entonces el sistema resultante es inconsistente.
- 16.* Sea Γ un conjunto infinito de fórmulas y α una fórmula tal que $\Gamma \models \alpha$. ¿Vale que $\Gamma \vdash \alpha$? Note que el Corolario al Teorema de completitud no se puede aplicar aquí.
- 17* α es *completa* si para toda $\beta \in \Im(p_1, \dots, p_n)$ se tiene $\alpha \vdash \beta$ o $\alpha \vdash \neg\beta$ pero no ambos. Demuestre que α es completa si y solo si existe exactamente una valuación v de p_1, \dots, p_n tal que $\bar{v}[\alpha] = V$.



Capítulo 4

Equivalencia

4.1 Algebra Proposicional

Dos fórmulas α y β se dicen *equivalentes* si se tiene $\alpha \vdash \beta$ y $\beta \vdash \alpha$. Esta relación se denota $\alpha \text{ eq } \beta$. Es claro que “eq” es una relación de equivalencia entre fbs; es decir, es reflexiva: $\alpha \text{ eq } \alpha$; simétrica: si $\alpha \text{ eq } \beta$ entonces $\beta \text{ eq } \alpha$; y transitiva: si $\alpha \text{ eq } \beta$ y $\beta \text{ eq } \gamma$ entonces $\alpha \text{ eq } \gamma$.

Ejemplos

- $\alpha \text{ eq } \neg\neg\alpha$, por Teo 5 y Teo 9.
- $(\alpha \supset \beta) \text{ eq } (\neg\beta \supset \neg\alpha)$, de derecha a izquierda por A 3 + MP, de izquierda a derecha por MTT + TD.
- $\neg(\alpha \vee \beta) \text{ eq } \neg\alpha \wedge \neg\beta$. Para probar $\neg(\alpha \vee \beta) \vdash \neg\alpha \wedge \neg\beta$ considere:
 - $\alpha \supset \alpha \vee \beta$ A7
 - $\neg(\alpha \vee \beta) \supset \neg\alpha$ Ejemplo (b)
 - $\neg(\alpha \vee \beta)$ P
 - $\neg\alpha$ MP 2,3

Por tanto, $\neg(\alpha \vee \beta) \vdash \neg\alpha$. Similarmente $\neg(\alpha \vee \beta) \vdash \neg\beta$. Para el converso basta observar que por MTP (Teo. 6), $\neg\alpha \wedge \neg\beta, \alpha \vee \beta \vdash \beta, \neg\beta$ y aplicar RA.

La verificación de equivalencias se simplifica gracias a la parte (3) del resultado siguiente.

Teorema 1. *Las siguientes condiciones son equivalentes:*

- (1) $\alpha \text{ eq } \beta$
- (2) $\vdash \alpha \Leftrightarrow \beta$
- (3) $\bar{v}[\alpha] = \bar{v}[\beta]$ para toda V .

Demostración. (1) \Rightarrow (2). Si $\alpha \text{ eq } \beta$ entonces $\alpha \vdash \beta$ y $\beta \vdash \alpha$. Por TD: $\vdash \alpha \supset \beta$ y $\vdash \beta \supset \alpha$, entonces $\vdash (\alpha \supset \beta) \wedge (\beta \supset \alpha)$. Es decir, $\vdash \alpha \Leftrightarrow \beta$.

(2) \Rightarrow (3). Si $\vdash \alpha \Leftrightarrow \beta$ entonces $\models \alpha \Leftrightarrow \beta$ por el Teorema de validez. Por lo tanto $\bar{v}[\alpha \Leftrightarrow \beta] = V$ para toda v , y a la fuerza $\bar{v}[\alpha] = \bar{v}[\beta]$ para toda v .

(3) \Rightarrow (1). Si $\bar{v}[\alpha] = \bar{v}[\beta]$ para toda v , entonces $\bar{v}[\alpha \supset \beta] = V$ para toda v y $\vdash \alpha \supset \beta$. Por completitud: $\vdash \alpha \Leftrightarrow \beta$. Entonces $\vdash \alpha \supset \beta$ y $\vdash \beta \supset \alpha$ (Re 3, Re 4). Usando MP tenemos $\alpha \vdash \beta$ y $\beta \vdash \alpha$ ■

Si α y β son fbf con letras proposicionales p_1, \dots, p_n y q_1, \dots, q_m respectivamente, el teorema anterior nos dice que para determinar si $\alpha \text{ eq } \beta$, o no, basta construir las tablas de verdad de α y β con respecto a las letras $\{p_1, \dots, p_n\} \cup \{q_1, \dots, q_m\}$; entonces $\alpha \text{ eq } \beta$ si y solo si α y β tienen la misma tabla.

Fue probablemente Leibnitz (1646-1726) el primero que concibió la posibilidad de un álgebra de proposiciones, análoga al álgebra de las matemáticas, útil para “calcular verdades”, esta sería su famosa “characteristica universalis”. Tal idea vino a cristalizar parcialmente con el álgebra de proposiciones desarrollada por George Boole (1815-1864), hoy día llamada *álgebra booleana*. Aunque con diferente notación, los siguientes pares de equivalencias que se pueden comprobar fácilmente por deducción o por tablas de verdad, corresponden a las leyes del álgebra de Boole:

E1	$\alpha \wedge \alpha \text{ eq } \alpha$	$\alpha \vee \alpha \text{ eq } \alpha$	Idempotencia
E2	$\alpha \wedge \beta \text{ eq } \beta \wedge \alpha$	$\alpha \vee \beta \text{ eq } \beta \vee \alpha$	Commutatividad
E3	$\alpha \wedge (\beta \wedge \gamma) \text{ eq } (\alpha \wedge \beta) \wedge \gamma$	$\alpha \vee (\beta \vee \gamma) \text{ eq } (\alpha \vee \beta) \vee \gamma$	Asociatividad

E4	$\alpha \wedge (\beta \vee \gamma) \text{ eq } (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$	$\alpha \vee (\beta \wedge \gamma) \text{ eq } (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$	Distributividad
E5	$\neg\neg\alpha \text{ eq } \alpha$		Doble negación
E6	$\neg(\alpha \wedge \beta) \text{ eq } \neg\alpha \vee \neg\beta$	$\neg(\alpha \vee \beta) \text{ eq } \neg\alpha \wedge \neg\beta$	Leyes de Morgan
E7	$\alpha \wedge (\beta \vee \neg\beta) \text{ eq } \alpha$	$\alpha \vee (\beta \wedge \neg\beta) \text{ eq } \alpha$	
E8	$\alpha \wedge (\beta \wedge \neg\beta) \text{ eq } \beta \wedge \neg\beta$	$\alpha \vee (\beta \wedge \neg\beta) \text{ eq } \beta \vee \neg\beta$	

Las leyes anteriores se refieren tan sólo a los conectivos \neg , \wedge \vee , pero las siguientes equivalencias adicionales permiten eliminar los otros conectivos en favor de \supset , \wedge , \vee :

E9	$\alpha \supset \beta \text{ eq } \neg\alpha \vee \beta$
E10	$\alpha \leftrightarrow \beta \text{ eq } (\neg\alpha \vee \beta) \wedge (\alpha \vee \neg\beta)$
E11	$\alpha \leftrightarrow \beta \text{ eq } (\alpha \wedge \beta) \vee (\neg\alpha \wedge \neg\beta)$
E12	$\neg(\alpha \supset \beta) \text{ eq } \alpha \wedge \neg\beta$
E13	$\neg(\alpha \leftrightarrow \beta) \text{ eq } (\alpha \wedge \neg\beta) \vee (\beta \wedge \neg\alpha)$

Para poder hacer álgebra con las leyes anteriores necesitamos que “eq” se comporte como una especie de igualdad, es decir, además de ser relación de equivalencia debe cumplir que al substituir fórmulas equivalentes “dentro” de una fórmula se mantenga la equivalencia. Aunque esto parece intuitivamente obvio requiere una demostración. Dadas fórmulas α , β , γ sea $\alpha(\beta/\gamma)$ la fórmula que resulta de reemplazar una ocurrencia de β como subfórmula de α por γ . Por ejemplo si α es $\neg((p \supset q) \supset (r \vee p))$, β es $p \supset p$, y γ es $\neg q \supset \neg p$, entonces $\alpha(\beta/\gamma)$ es $((\neg p \supset \neg q) \supset (r \vee p))$. Evidentemente, si β no ocurre en α entonces $\alpha(\beta/\gamma)=\alpha$, y si β es α entonces $\alpha(\beta/\gamma)=\gamma$.

Lema 2. Sea v una valuación, si $\bar{v}[\beta]=\bar{v}[\gamma]$ entonces $\bar{v}[\alpha(\beta/\gamma)] = \bar{v}[\alpha]$.

Demostración. Suponemos β y γ fijas con $\bar{v}[\beta]=\bar{v}[\gamma]$. Si β no ocurre en α el resultado es trivial. Hacemos entonces inducción en las fórmulas α suponiendo que contienen a β por subfórmula, y llamando α° a $\alpha(\beta/\gamma)$.

I. $\alpha=p$ letra proposicional. Entonces $\beta=\alpha$ y por lo tanto $\alpha^\circ=\gamma$.

Por hipótesis $\bar{v}[\alpha^\circ]=\bar{v}[\gamma]=\bar{v}[\beta]=\bar{v}[\alpha]$.

(II) Suponga el lema cierto para α_1 y α_2 , y sea $\alpha = \neg\alpha_1$ ó $\alpha = \alpha_1 \square \alpha_2$. Si $\beta = \alpha$ el resultado se sigue como en I. Si $\beta \neq \alpha$ entonces β debe ser subfórmula de α_1 ó α_2 y por lo tanto $(\neg\alpha_1)^o = \neg\alpha_1^o$ y $(\alpha_1 \square \alpha_2)^o = \alpha_1^o \square \alpha_2^o$. Como por hipótesis de inducción tenemos $\bar{v}[\alpha_1^o] = \bar{v}[\alpha_1]$ y $\bar{v}[\alpha_2^o] = \bar{v}[\alpha_2]$ entonces $\bar{v}[(\neg\alpha_1)^o] = \bar{v}[\neg\alpha_1^o] = \bar{v}[\neg\alpha_1]$ y $\bar{v}[(\alpha_1 \square \alpha_2)^o] = \bar{v}[\alpha_1^o \square \alpha_2^o] = \bar{v}[\alpha_1 \square \alpha_2]$. ■

Corolario. $b \leftrightarrow \gamma \vdash \alpha \leftrightarrow \alpha(\beta/\gamma)$.

Demostración. Suponga $\bar{v}[\beta \leftrightarrow \gamma] = V$ entonces $\bar{v}[\beta] = \bar{v}[\gamma]$. Por el lema anterior: $\bar{v}[\alpha] = \bar{v}[\alpha(\beta/\gamma)]$ y por lo tanto $\bar{v}[\alpha \leftrightarrow \alpha(\beta/\gamma)]$. Esto demuestra que $\beta \leftrightarrow \gamma \vdash \alpha \leftrightarrow \alpha(\beta/\gamma)$. ■

Teorema de Substitución. Si $\beta \text{ eq } \gamma$ entonces $\alpha \text{ eq } \alpha(\beta/\gamma)$.

Demostración. Si $\beta \text{ eq } \gamma$ entonces $\vdash \beta \leftrightarrow \gamma$. Pero $\beta \leftrightarrow \gamma \vdash \alpha \leftrightarrow \alpha \leftrightarrow (\beta/\gamma)$ por el Corolario anterior. Por lo tanto $\vdash \alpha \leftrightarrow \alpha(\beta/\gamma)$ y $\alpha \text{ eq } \alpha(\beta/\gamma)$. ■

Este último teorema nos permite hacer, ahora sí, álgebra de proposiciones donde la sustitución de equivalencias produce nuevas equivalencias, como en álgebra ordinaria la sustitución de ecuaciones produce nuevas ecuaciones.

Ejemplos.

(a) $\alpha \vee \beta \text{ eq } \neg(\neg(\alpha \vee \beta))$, por E5; pero $\neg(\alpha \vee \beta) \text{ eq } \neg\alpha \wedge \neg\beta$ por E6. Substituyendo tenemos: $\alpha \vee \beta \text{ eq } \neg(\neg\alpha \wedge \neg\beta)$

(b) $\neg(\alpha \supset \beta) \text{ eq } \neg(\neg\alpha \vee \beta)$ Subst de E9

eq $\neg\neg\alpha \wedge \neg\beta$ E6

eq $\alpha \wedge \neg\beta$ Subst de E5.

Esto nos muestra que la equivalencia E11 se puede obtener de E5, E6, E11 por medio de nuestra álgebra de equivalencias.

(c) Usando las equivalencias dadas podemos probar nuevas leyes como la de *simplificación*: $\alpha \vee (\alpha \wedge \beta) \text{ eq } \alpha$,

$$\begin{aligned}
 \alpha \vee (\alpha \wedge \beta) &\equiv (\alpha \wedge (\beta \vee \neg \beta)) \vee (\alpha \wedge \beta) && \text{Subst E7} \\
 &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \neg \beta)) \vee (\alpha \wedge \beta) && \text{Subst E4} \\
 &\equiv ((\alpha \wedge \neg \beta) \vee (\alpha \wedge \beta)) \vee (\alpha \wedge \beta) && \text{Subst E2} \\
 &\equiv (\alpha \wedge \neg \beta) \vee ((\alpha \wedge \beta) \vee (\alpha \wedge \beta)) && \text{E3} \\
 &\equiv (\alpha \wedge \neg \beta) \vee (\alpha \wedge \beta) && \text{Subst E1} \\
 &\equiv \alpha \wedge (\neg \beta \vee \beta) && \text{E4} \\
 &\equiv \alpha && \text{E7}
 \end{aligned}$$

Ejercicios

1. Demuestre deductivamente:
 - a. $\alpha \supset \beta \equiv \neg \alpha \vee \beta$
 - b. $\alpha \supset (\beta \supset \gamma) \equiv \beta \supset (\alpha \supset \gamma) \equiv (\alpha \wedge \beta) \supset \gamma$
 - c. $\alpha \leftrightarrow \beta \equiv \neg \alpha \leftrightarrow \neg \beta$
 - d. $* \neg(p \leftrightarrow q) \leftrightarrow q \equiv \neg p$

2. Demuestre usando equivalencias E1 a E10 solamente:
 - a. $\alpha \equiv (\alpha \wedge \beta) \vee (\alpha \wedge \neg \beta)$
 - b. $\alpha \wedge (\alpha \vee \beta) \equiv \alpha$ (Simplificación)
 - c. $(\neg \alpha \vee \beta) \wedge (\alpha \vee \neg \beta) \equiv (\alpha \wedge \beta) \vee (\neg \alpha \wedge \neg \beta)$
 - d. $((((p \supset p) \supset p) \supset p) \supset p) \supset p \equiv p \vee \neg p$
 - e. $\neg(p \leftrightarrow q) \leftrightarrow q \equiv \neg p$

3. Demuestre E11, E12, E13 por álgebra, de E1-E10.

4. Son equivalentes los pares de fórmulas siguientes?
 - a. $p \supset (p \supset r), (p \supset q) \supset r$
 - b. $p \leftrightarrow (q \leftrightarrow r), (p \leftrightarrow q) \leftrightarrow r$
 - c. $p \supset r \vee s, (p \supset r) \vee (p \supset s)$

- d. $p \supset r \vee s, (p \supset r) \vee s$
 e. $p \supset r \vee s, \neg r \supset (p \supset s)$
 f. $p \vee q \supset r, (p \supset r) \vee (q \supset r)$
 f'. $p_1 \wedge p_2 \wedge \dots \wedge p_n, p_1 \wedge (p_1 \supset p_2) \wedge (p_2 \supset p_3) \wedge \dots \wedge (p_{n-1} \supset p_n)$
 g. $p \wedge \neg p, (q \supset \neg q) \wedge q$
 h. $p \vee \neg p, (r \supset s) \supset (\neg s \supset \neg r)$
5. Sea $\Gamma = \{\sigma_1, \dots, \sigma_n\}$. Para cualquier par de fórmulas α y β defina: $\alpha \sim \beta$ si y sólo si $\Gamma \vdash \alpha \leftrightarrow \beta$
 - Demuestre que \sim es una relación de equivalencia.
 - Demuestre que el Teorema de Substitución vale para \sim . Es decir, si $\beta \sim \gamma$ entonces $\alpha \sim \alpha(\beta/\gamma)$.
6. (Lenguaje vs. metalenguaje) En cada caso diga si las expresiones siguientes son fbf's, afirmaciones sobre fbf's, o expresiones sin sentido:
 - $(\alpha \leftrightarrow \beta) \text{ eq } (\beta \leftrightarrow \alpha)$
 - $(\alpha \text{ eq } \beta) \text{ eq } (\beta \text{ eq } \alpha)$
 - $(\alpha \text{ eq } \beta) \text{ eq } (\beta \leftrightarrow \alpha)$
 - $(\alpha \text{ eq } \beta) \text{ si y solo si } (\beta \text{ eq } \alpha)$
 - $(\alpha \leftrightarrow \beta) \text{ si y solo si } (\beta \leftrightarrow \alpha)$
 - $(\alpha \leftrightarrow \beta) \leftrightarrow (\beta \gamma \alpha)$

4.2 Conjuntos completos de conectivos

Un conjunto de conectivos es *completo* si toda fbf es equivalente a una fórmula que contiene a lo sumo conectivos del conjunto. Por las equivalencias E9, E10 y el Teorema de sustitución, es claro que es posible eliminar todas las ocurrencias de los conectivos \supset y \leftrightarrow de cualquier fórmula hasta obtener una fórmula equivalente, que a lo sumo contiene \neg , \wedge , \vee . Por lo

tanto $\{\neg, \wedge, \vee\}$ es completo. Mas aún $\{\neg, \wedge\}$ y $\{\neg, \vee\}$ son completos, como se desprende de las equivalencias:

E14	$\alpha \vee \beta \text{ eq } \neg \neg \alpha \vee \neg \neg \beta \text{ eq } \neg(\neg \alpha \wedge \neg \beta)$
E15	$\alpha \wedge \beta \text{ eq } \neg \neg \alpha \wedge \neg \neg \beta \text{ eq } \neg(\neg \alpha \vee \neg \beta)$

que se siguen de la ley de doble negación y las leyes de Morgan, y permiten eliminar \vee en favor de \wedge (y \neg) y viceversa.

Ejemplo

$$\begin{aligned} p \leftrightarrow (q \supset \neg r) &\text{ eq } [p \wedge (q \supset \neg r)] \vee [\neg p \wedge \neg (q \supset \neg r)] \\ &\text{ eq } [p \wedge (\neg q \vee \neg r)] \vee [\neg p \wedge \neg (\neg q \vee \neg r)] \end{aligned} \quad (1)$$

Utilizando E 14 obtenemos:

$$\begin{aligned} p \leftrightarrow (q \supset \neg r) &\text{ eq } \neg \{ \neg [p \wedge (\neg q \vee \neg r)] \wedge \neg [\neg p \wedge \neg (\neg q \vee \neg r)] \} \\ &\text{ eq } \neg \{ \neg [p \wedge \neg (\neg \neg q \wedge \neg \neg r)] \wedge \neg [\neg p \wedge \neg (\neg \neg q \wedge \neg \neg r)] \} \\ &\text{ eq } \neg \{ \neg [p \wedge \neg (q \wedge r)] \wedge \neg [\neg p \wedge (q \wedge r)] \} \end{aligned}$$

Utilizando E15 en (1), obtenemos:

$$p \leftrightarrow (q \supset \neg r) \text{ eq } \neg [\neg p \vee (\neg q \vee \neg r)] \vee \neg [p \vee (\neg q \vee \neg r)].$$

Lema. *El conjunto $\{\wedge, \vee, \supset, \leftrightarrow\}$ no es completo.*

Demostración. Sea L el conjunto de fbf's que contienen a lo sumo los conectivos \wedge, \vee, \supset y \leftrightarrow . Demostramos por inducción en fórmulas de L que si α tiene letras proposicionales p_1, \dots, p_n y v_o es la valuación $v_o(p_1) = \dots = v_o(p_n) = V$ entonces $\bar{v}_o[\alpha] = V$.

- I. Si $\alpha = p_i$ letra proposicional, el resultado es trivial.
- II. Suponga el lema cierto para α y β y sean p_1, \dots, p_n las letras proposicionales de $\alpha \supset \beta$. Por hipótesis $\bar{v}_o[\alpha] = \bar{v}_o[\beta] = V$. Por lo tanto, $\bar{v}_o[\alpha \wedge \beta] = \bar{v}_o[\alpha \vee \beta] = \bar{v}_o[\alpha \supset \beta] = \bar{v}_o[\alpha \leftrightarrow \beta] = V$. Para demostrar que $\{\wedge, \vee, \supset, \leftrightarrow\}$ no es completo basta observar que $\bar{v}_o[\neg p] = F$ y por lo tanto $\neg p$ no puede ser equivalente a ninguna fórmula de L. ■

Es posible expresar todas las fórmulas del C. de Proposiciones por medio de un solo conectivo. Se define el *símbolo de Sheffer* como el conectivo binario “|” cuya tabla de verdad es

$\alpha \beta$	$\alpha \beta$
V V	F
V F	V
F V	V
F F	V

{|} es completo ya que $\neg \alpha \equiv \alpha | \alpha$ y $\alpha \vee \beta \equiv (\alpha | \alpha) | (\beta | \beta)$, como se comprueba haciendo las correspondientes tablas. Otro conectivo, completo por sí solo es la *flecha de Pierce*, dado por la tabla

$\alpha \beta$	$\alpha \downarrow \beta$
V V	F
V F	F
F V	F
F F	V

No es difícil ver que $\alpha | \beta \equiv \neg(\alpha \wedge \beta)$ y $\alpha \downarrow \beta \equiv \neg \alpha \wedge \neg \beta$ de manera que $\alpha | \beta$ se podría interpretar como “no ambas α y β ” y $\alpha \downarrow \beta$ significaría “ni α ni β ”.

J. Nicod demostró que si todas las fórmulas se expresan en términos de | entonces el sistema que consiste en el axioma:

$$(\alpha | (\beta | \gamma)) | \{ [\tau | (\tau | \tau)] | [(\sigma | \beta) | ((\alpha | \sigma) | (\alpha | \sigma))] \}$$

y la regla: $\alpha, \alpha | (\beta | \gamma) \vdash \gamma$ es deductivamente completo.

Ejercicios

1. Demuestre que los conjuntos $\{\neg\}, \{\wedge\}, \{\vee\}, \{\supset\}, \{\leftrightarrow\}$ no son completos.
2. Demuestre que $\{\supset, \neg\}$ es completo.
3. *Demuestre que $\{\leftrightarrow, \neg\}$ no es completo (Ayuda: toda fórmula cuyos conectivos son a lo sumo \neg y \leftrightarrow , y cuyas letras proposicionales están entre p_1, \dots, p_n , $n \geq 2$, es verdadera para un número par de valuaciones de p_1, \dots, p_n)
4. Demuestre que $\{\downarrow\}$ es completo.
5. sea ∇ un conectivo binario tal que $\alpha \nabla \beta$ eq $\neg(\alpha \leftrightarrow \beta)$, ∇ tiene la tabla de "o" exclusivo. demuestre que $\{\wedge, \leftrightarrow, \nabla\}$ es completo.
6. Sea \perp el conectivo de una sola variable cuya tabla es

α	$\wedge(\alpha)$
V	F
F	F

Demuestre que $\{\supset, \perp\}$ es completo, pero $\{\wedge, \vee, \perp\}$ no lo es.

7. Se define un conectivo ternario $\#$ tal que $\#(\alpha, \beta, \gamma)$ es verdadera para las valuaciones mostradas en la tabla y falso para las demás

α	β	γ	$\#(\alpha, \beta, \gamma)$
V	V	F	V
V	F	F	V
F	V	V	V
F	F	F	V

- a. Demuestre que $\#(\alpha, \beta, \gamma)$ es equivalente a una fórmula con los conectivos $\neg, \wedge, \vee, \supset, \leftrightarrow$.

- b. Demuestre que $\{\#\}$ es completo.
- 8.* Defina un conectivo ternario M tal que para toda valuación v : $\bar{v}[M(\alpha, \beta, \gamma)] = V$ si y solo si a lo sumo uno de los valores $\bar{v}[\alpha], \bar{v}[\beta], \bar{v}[\gamma]$ es F . Demuestre que $\{M\}$ no es completo.
- 9.* Demuestre que $|$ y \downarrow son los únicos conectivos binarios completos.

4.3 Formas normales, funciones booleanas

Usaremos la notación $\sigma_1 \wedge \sigma_2 \wedge \dots \wedge \sigma_n$ para denotar cualquier forma de asociar la conjunción de $\sigma_1, \sigma_2, \dots, \sigma_n$. Por las equivalencias E2 y E3, cualquier rearrreglo de paréntesis y cualquier permutación de los σ_i en estas fórmulas produce una fórmula equivalente. Análogamente escribiremos $\sigma_1 \vee \sigma_2 \vee \dots \vee \sigma_n$.

$\sigma_1 \wedge \dots \wedge \sigma_n$ es una *conjunción elemental* si cada σ_i es una letra proposicional o la negación de una letra proposicional. Una *forma normal disyunta FND*, es una fórmula de la forma $\sigma_1 \vee \dots \vee \sigma_n$, $n \geq 1$, donde cada σ_i es una conjunción elemental.

Ejemplos:

1. $(p \wedge q \neg r) \vee (\neg p \wedge q \wedge r) \vee (\neg p \wedge \neg q \wedge r)$
2. $(p_1 \wedge p_2) \vee p_3 \vee (\neg p_1 \vee p_3)$
3. $(p \wedge r \wedge \neg q) \vee (\neg r \wedge q)$
4. $(p \wedge q) \vee (\neg p \wedge \neg q) \vee (q \wedge p)$

Una FND es *plena*, FNDP, si en cada α_i ocurren las mismas letras proposicionales, cada letra proposicional ocurre sólo una vez en cada α_i , y para $i \neq j$, α_j . Solamente el primero de los ejemplos anteriores es una FNDP. Si una FND no contiene una letra proposicional y su negación en la misma

conjunction elemental, es posible transformarla en una FNDP aplicando tanto como sea necesario la equivalencia $\sigma \equiv \alpha \wedge (\alpha \vee \neg \alpha) \equiv (\sigma \wedge \alpha) \vee (\sigma \wedge \neg \alpha)$

Ejemplo:

$$((\neg p_1 \wedge p_2) \vee p_1) \equiv ((\neg p_1 \wedge p_2) \vee (p_1 \wedge p_2) \vee (p_1 \wedge \neg p_2)).$$

Demostraremos que toda fórmula es equivalente a una FND.

Sea $\{V, F\}^n = \{V, F\} \times \dots \times \{V, F\}$, n veces. Una función $f: \{V, F\}^n \rightarrow \{V, F\}$ se llama una *función booleana*. La tabla de verdad de cualquier fbf $\alpha \in S(p_1, \dots, p_n)$ se puede ver como la tabla de una función booleana que llamaremos $f\alpha$. Si identificamos cada valuación v de p_1, \dots, p_n con la n-tupla $(v(p_1), \dots, v(p_n))$ de $\{V, F\}^n$, entonces $f\alpha(v) = \bar{v}[\alpha]$. El siguiente lema muestra que toda función booleana se puede representar por la tabla de una fbf.

Lema 1. Para cualquier función booleana $f: \{V, F\}^n \rightarrow \{V, F\}$ existe una fbf α de $S(p_1, \dots, p_n)$, en FND tal que $f = f\alpha$ (es decir la tabla de $f\alpha$ coincide con la de f). Si $f(v) = V$ por lo menos para una valuación v , entonces α puede escogerse de manera que sea FNDP.

Para probar este lema necesitamos un lema auxiliar. Recordemos que $p^v = p$ si $v[p] = V$ y $p^v = \neg p$ si $v[p] = F$.

Lema 2. Para cualquier valuación w de p_1, \dots, p_n , $\bar{w}[p_1 \bar{v} \wedge \dots \wedge p_n \bar{v}] = V$ si y solo si $w = v$.

Demostración. $w[p_1 \bar{v} \wedge \dots \wedge p_n \bar{v}] = V$ si y solo si $\bar{w}[p_1 \bar{v}] = \dots = \bar{w}[p_n \bar{v}] = V$

Consideramos casos:

(a) $p_i \bar{v} = p_i$ entonces $v[p_i] = V$ y además $w[p_i] = \bar{w}[p_i \bar{v}] = V$

(b) $p_i \bar{v} = \neg p_i$ entonces $v[p_i] = F$ y además $\bar{w}[\neg p_i] = \bar{w}[p_i \bar{v}] = V$, lo que implica $w[p_i] = F$. En cualquier caso $w[p_i] = v[p_i]$ ■

Demostración del Lema 1. Si $f(v) = F$ para toda v basta escoger $\alpha = p_1 \wedge \neg p_1$. Supongamos que $f(v) = V$, por lo menos para una valuación. sea $A = \{v_1, \dots, v_k\}$ el conjunto de valuaciones tales que $f(v_i) = V$ y defina $\alpha = \alpha_1 \vee \dots \vee \alpha_k$ donde $\alpha_i = p_1^{v_i} \wedge \dots \wedge p_n^{v_i}$. Entonces tenemos que para cualquier $v \in \{V, F\}^n$

$v[\alpha] = V$ si y solo si $\bar{v}[\alpha_i] = V$ para algún $i \in \{1, \dots, k\}$
 si y solo si $v[p_1 v_i \wedge \dots \wedge p_n v_i] = V$
 si y solo si $v = v_i$, por lema 1, para algún $i \in \{1, \dots, k\}$
 si y solo si $v \in A$
 si y solo si $f(v) = V$.

Esto significa que $\bar{v}[\alpha]$ y $f(v)$ son V para las mismas valuaciones. A la fuerza son F para las mismas valuaciones y tenemos $f(v) = \bar{v}[\alpha] = f\alpha(v)$ para toda v . ■

Ejemplo

Dada la función cuya tabla se muestra, podemos hallar los α_i 's y la FNDP que tiene la misma tabla

p_1	p_2	p_3	α_i 's
V	V	V	F
V	V	F	V
V	F	V	F
V	F	F	F
F	V	V	F
F	V	F	V
F	F	V	V
F	F	F	F

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

Teorema 3. *Toda fórmula α es equivalente a una FND. Si $\bar{v}[\alpha] = V$ por lo menos para una valuación v , entonces α es equivalente a una FNDP.*

Demostración. Si $\bar{v}[\alpha] = F$ para toda v entonces $\alpha \equiv p \wedge \neg p$. Si $\bar{v}[\alpha] = V$ para alguna v y p_1, \dots, p_n son sus letras proposicionales, considere la función $f\alpha$ dada por la tabla de α . Por el lema 1, existe una FNDP α' tal que $f\alpha = f\alpha'$. Por lo tanto α y α' tienen la misma tabla, lo cual significa que $\alpha \equiv \alpha'$. ■

Hay métodos diferentes del de la tabla de verdad para hallar FND's equivalentes a una fórmula dada. El siguiente algoritmo permite llegar siempre a una FND de una fórmula γ , utilizando solamente las equivalencias E1-E10.

Algoritmo

I. Elimine todas las ocurrencias de los conectivos \supset y \leftrightarrow por medio de las equivalencias $\alpha \supset \beta \equiv \neg \alpha \vee \beta$, E9 y $\alpha \leftrightarrow \beta \equiv (\alpha \wedge \beta) \vee (\neg \alpha \wedge \neg \beta)$, E10.

II. Aplique las leyes De Morgan: $\neg(\alpha \wedge \beta) \equiv \neg \alpha \vee \neg \beta$, $\neg(\alpha \vee \beta) \equiv \neg \alpha \wedge \neg \beta$, E6, de izquierda a derecha, de manera que las negaciones tengan cada vez un alcance menor. Cuando estas reglas no se pueden aplicar más es porque todas las ocurrencias de negaciones son de la forma $\neg p$ ó $\neg \neg \dots p$, con p una letra proposicional.

III. Utilice la equivalencia $\neg \neg \alpha \equiv \alpha$, E5 para eliminar negaciones repetidas, hasta que toda negación de γ sea de la forma $\neg p$, con p letra proposicional.

IV. Utilice las leyes distributivas $\alpha \wedge (\beta \vee \gamma) \equiv (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$ y $(\beta \vee \gamma) \wedge \alpha \equiv (\beta \wedge \alpha) \vee (\gamma \wedge \alpha)$, E4, E1, de izquierda a derecha, hasta que no sea posible aplicarlas más. Esto sucede cuando ningún “ \vee ” está bajo el alcance de un “ \wedge ”, lo cual implica que cada \wedge está flanqueado solamente por otros \wedge o por letras proposicionales o sus negaciones. De aquí se deduce que tenemos ya una FND.

Ejemplo

$$\begin{aligned}
 \neg((p \supset q) \vee (r \wedge p)) &\equiv \neg((\neg p \vee p) \vee (r \wedge p)) \\
 &\equiv \neg(\neg p \vee q) \wedge \neg(r \wedge p) \\
 &\equiv (\neg \neg p \wedge \neg q) \wedge (\neg r \wedge \neg p) \\
 &\equiv (p \wedge \neg q) \wedge (\neg r \wedge \neg p) \\
 &\equiv (p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg q \wedge r)
 \end{aligned}$$

La cadena de equivalencias puede continuarse para obtener una FNDP:

$$\begin{aligned}
 (p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg q \wedge r) &\equiv (p \wedge \neg q \wedge \neg r) \vee (q \wedge (\neg p \wedge \neg r)) \quad E2, E3 \\
 &\equiv (p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg p) \quad E8 \\
 &\equiv p \wedge \neg q \wedge \neg r \quad E7
 \end{aligned}$$

Si se utilizan todas las equivalencias E1 a E13, más las leyes de simplificación:

$$\begin{aligned}\alpha \vee (\alpha \wedge \beta) &\text{ eq } \alpha \\ \alpha \wedge (\alpha \vee \beta) &\text{ eq } \alpha\end{aligned}$$

se obtiene un método bastante efectivo para hallar FND's, y FNPD's.

Como otra aplicación del Lema 1 tenemos:

Teorema 4. *El máximo número de fórmulas no equivalentes que se pueden formar en el lenguaje $\mathfrak{S}(p_1, \dots, p_n)$ es 2^{2^n} .*

Demostración. Sea $\alpha, \alpha' \in \mathfrak{S}(p_1, \dots, p_n)$. Obviamente $\alpha \text{ eq } \alpha'$ si y solo si $f_\alpha = f_{\alpha'}$. Por el lema 1 esto implica que existe una correspondencia 1 a 1 entre las clases de equivalencia de fórmulas en $\mathfrak{S}(p_1, \dots, p_n)$ y las funciones de $\{V, F\}^n$ en $\{V, F\}$, como el número de estas funciones es $2^{(2^n)}$, ése será el número de clases de equivalencia, y por lo tanto el número máximo de fórmulas no equivalentes. ■

Ejemplo

En $\mathfrak{S}(p, q, r)$ se pueden formar $2^{2^3} = 256$ fórmulas no equivalentes.

Una *forma normal conjuntiva* FNC, tiene la forma $\alpha_1 \wedge \dots \wedge \alpha_n$ en donde cada una es una disyunción elemental. Es decir, α_i es de la forma $\sigma_1 \vee \dots \vee \sigma_k$ en donde cada σ_j es letra proposicional o su negación. Una FNC *plena* se define análogamente a una FNPD. Como es de esperar tenemos el teorema dual:

Teorema 5. *Toda fórmula α es equivalente a una FNC. Si α no es tautología, α es equivalente a una FNCP (forma normal conjuntiva plena).*

Demostración. Si α es tautología $\alpha \text{ eq } (p \vee \neg p)$. Si α no es tautología, entonces $\neg(\neg \alpha)$ es V para alguna valuación y $\neg \alpha \text{ eq } \alpha_1 \vee \dots \vee \alpha_n$, una FNPD. Por lo tanto, $\alpha \text{ eq } \neg \neg \alpha \text{ eq } \neg(\alpha_1 \vee \dots \vee \alpha_n) \text{ eq } \neg \alpha_1 \wedge \dots \wedge \neg \alpha_n$. Pero cada α_i tiene la forma $\alpha_i = \pm p_1 \wedge \dots \wedge \pm p_k$ y entonces $\neg \alpha_i \text{ eq } \neg(\pm p_1 \wedge \dots \wedge \pm p_k) \text{ eq }$

$\pm p_1 \dots \vee \pm p_k = D_i$, donde D_i es una disyunción elemental. Finalmente α es eq $(D_1 \wedge \dots \wedge D_n)$, una FNCP. ■

Ejercicios

1. Halle FND para las fórmulas siguientes

- $\neg(p \leftrightarrow q)$
- $p \supset (q \supset (r \supset (s \supset (t \supset u))))$
- $((p \supset p) \supset p)$
- $\neg(p_1 \wedge (\neg p_2 \vee (p_2 \wedge \neg p_3)))$
- $\neg(p \leftrightarrow q) \leftrightarrow q$
- $\neg(p \wedge q \supset (q \vee s \supset p))$
- $\neg((p \leftrightarrow q) \supset \neg(r \wedge \neg(s \supset q)))$
- $(p_1 \vee \neg p_2) \wedge (\neg p_1 \vee p_2) \wedge (p_1 \vee p_2 \vee \neg p_3)$

2. Halle FNDP:

- $p \supset p$
- $((p \vee \neg q) \leftrightarrow r) \supset (p \wedge r)$
- $(p_1 \wedge \neg p_2 \wedge p_3 \wedge \neg p_4 \wedge \neg p_5) \vee (p_6 \wedge \neg p_6)$

3. ¿Cuántas conjunciones elementales debe tener la FNDP, con respecto a las letras p_1, \dots, p_7 , de la fórmula siguiente?

$$p_1 \supset (p_2 \supset (p_3 \supset (p_4 \supset (p_5 \supset (p_6 \supset p_7))))).$$

4. Sea $f: \{V,F\}^4 \rightarrow \{V,F\}$ la función

$$f(x_1, x_2, x_3, x_4) = \begin{cases} V & \text{si el número de las veces es mayor que} \\ & \text{el de las efes en } (x_1, x_2, x_3, x_4) \\ F & \text{de lo contrario} \end{cases}$$

Halle una fbf cuya tabla de verdad represente esta función.

5. Halle fbf lo más sencillas posibles que representen las funciones siguientes de $\{V, F\}^n$ a $\{V, F\}$
- $f(x_1, \dots, x_n) = x_n$
 - $f(x_1, \dots, x_n) = \begin{cases} F & \text{si } x_1 = x_2 = \dots = x_n = V \\ V & \text{de lo contrario} \end{cases}$
 - $f(x_1, \dots, x_n) = \begin{cases} F & \text{si } x_1 = x_2 = \dots = x_n = V \text{ y } x_n = F \\ V & \text{de lo contrario} \end{cases}$
6. Halle 16 fórmulas de $\mathfrak{F}(p, q)$ de tal manera que toda otra fórmula de $\mathfrak{F}(p, q)$ sea equivalente a una de las 16.
7. ¿Cuál es el máximo número posible de fórmulas no equivalentes en $\mathfrak{F}(p)$ cuando $p = \{p_1, p_2, \dots\}$ es infinito enumerable?
8. ¿Dada $\alpha \in \mathfrak{F}(p, q)$ cuántas fórmulas equivalentes a α hay en $\mathfrak{F}(p, q)$?
9. Describa un algoritmo que produzca una FNC de una fórmula por una cadena de equivalencias, similar al que dimos para la FND.
10. Sea α una fórmula de $\mathfrak{F}(p_1, \dots, p_n)$ que no es una tautología. Cómo se puede usar la tabla de verdad de α con respecto a p_1, \dots, p_n , para hallar la FNCP de α con respecto a p_1, \dots, p_n ?
11. Halle FNC para cada fórmula del ejercicio 15.
12. Demuestre que si $\alpha \equiv \beta$, entonces es posible llegar a α a β por una cadena de equivalencias, usando solamente E1 a E10. (Ayuda: si $\alpha, \beta \in \mathfrak{F}(p_1, \dots, p_n)$ entonces $\alpha \equiv \beta$ si y solo si ambas tienen la misma FNDP con respecto a p_1, \dots, p_n , o ambas son equivalentes a $p_1 \wedge \neg p_1$).
13. Sea v_o la valuación tal que $v_o(p) = V$ para cualquier letra proposicional. sea $D_1 \wedge \dots \wedge D_n$ una FNC tal que $\bar{v}_o[D_1 \wedge \dots \wedge D_n] = V$.

- Demuestre que en cada disyunción elemental debe aparecer una letra proposicional no negada.
- Demuestre que $D_1 \wedge \dots \wedge D_n$ es equivalente a una fórmula sin negaciones.
- Sea α una fórmula arbitraria tal que $\bar{v}_o[\alpha] = V$ demuestre que α es equivalente a una fórmula sin negaciones. ¿Vale el converso?
- Halle una fórmula sin negaciones equivalente a

$$\neg((p \supset \neg q) \wedge (q \supset r)).$$

- Determine si la fórmula siguiente es equivalente a una fórmula sin negaciones:

$$\neg((p \leftrightarrow q) \supset \neg(r \wedge \neg(s \supset q)))$$

- Use el problema anterior para mostrar que toda fórmula α es equivalente a α' , o a $\neg\alpha'$, donde α' no tiene negaciones.
- Demuestre que si $\bar{v}_o[\alpha] = V$, α es equivalente a una fórmula que a lo sumo contiene \wedge y \supset .
- Una función $f: \{0,1\}^n \rightarrow \{0,1\}$ es una función de *valores binarios* de n variables. Demuestre que toda función de valores binarios $f(x_1, \dots, x_n)$ se puede obtener por composición a partir de las funciones $h(x) = 1 - x$, $g(x,y) = x \cdot y$ (Ayuda: identifique V con 1 y F con 0, entonces las tablas de h y g corresponden a las tablas de \neg y \wedge).
- Identificando V con 1 y F con 0, ¿cuáles son las funciones de valores binarios correspondientes a $p \vee q$, $p \supset q$, $p \leftrightarrow q$, $(p \supset q) \supset r$?
- Demuestre que toda función de valores binarios se puede obtener por composición de la función $g(x,y) = (1-x)(1-y)$. (Note que $x = g(g(x, x), g(x, x))$).

- Principios de dualidad.** Sea α una fórmula con los conectivos \neg , \wedge , y \vee . La *fórmula dual*, α^d , es el resultado de intercambiar \wedge y \vee en α . Demuestre:

- a. Si $\vdash \alpha$ entonces $\vdash \neg \alpha^d$
- b. Si $\alpha \text{ eq } \beta$ entonces $\alpha^d \text{ eq } \beta^d$.
- c. Sea α^* el resultado de cambiar cada letra proposicional de α por su negación, entonces $\alpha \text{ eq } \neg(\alpha^*)^d$.

20*. Dada una función $f: P \rightarrow [0, 1]$, donde P es un conjunto de letras proposicionales y $[0, 1]$ es el intervalo de números reales entre 0 y 1 inclusive. Defina $f: \mathfrak{F}(P) \rightarrow [0, 1]$ de acuerdo a las reglas siguientes:

$$\begin{aligned} f(\alpha \wedge \beta) &= \min(f(\alpha), f(\beta)) \\ f(\alpha \vee \beta) &= \max(f(\alpha), f(\beta)) \\ f(\neg \alpha) &= 1 - f(\alpha) \\ f(\alpha \supset \beta) &= \max(1 - f(\alpha), f(\beta)) \end{aligned}$$

Demuestre que $\vdash \alpha$ si y solo si $f(\alpha) \geq 1/2$ para toda función f de las letras proposicionales de α en $[0, 1]$.

21*. **Teorema de Interpolación.** Para cada fórmula α y letra proposicional p , sea $\alpha^{p,q} = \alpha(p/p \wedge \neg q) \vee \alpha(p/q \vee \neg q)$. Demuestre:

- a. $\alpha \vdash \alpha^{p,q}$
- b. Si $\alpha \vdash \beta$ y p no ocurre en β entonces $\alpha^{p,q} \vdash \beta$.
- c. Si $\alpha \vdash \beta$ entonces existe una fórmula γ , cuyas letras proposicionales son comunes a α y β tal que $\alpha \vdash \gamma \vdash \beta$.

Capítulo 5

Resolución en el cálculo de proposiciones

Extendemos la noción de equivalencia entre fbf's a conjuntos. Dos conjuntos de fórmulas bien formadas Σ y Σ' son *equivalentes*, $\Sigma \text{ eq } \Sigma'$, si $\Sigma \vdash \varphi$ para cada $\varphi' \in \Sigma'$ y $\Sigma' \vdash \varphi$ para cada $\varphi \in \Sigma$.

Si una fbf α tiene una FNC $D_1 \wedge \dots \wedge D_n$ entonces $\{\alpha\} \text{ eq } \{D_1, \dots, D_n\}$, un conjunto de disyunciones elementales que llamaremos Γ_α . En general, un conjunto arbitrario S de fbf's es equivalente a un conjunto de disyunciones elementales, es suficiente tomar

$$\Gamma_\Sigma = \bigcup_{\alpha \in \Sigma}^{\infty} \Gamma_\alpha$$

Como poseemos algoritmos para poner una fórmula en FNC, es claro que podemos construir Γ_Σ a partir de Σ . Podríamos trabajar entonces exclusivamente con disyunciones elementales.

Daremos aquí un sistema deductivo para el lenguaje consistente de disyunciones elementales exclusivamente, el cual es la base de muchos métodos de demostración por medio de computadores (*deducción automática*).

Sea \mathcal{D} (p_1, \dots, p_n) el conjunto de todas las disyunciones elementales cuyas letras proposicionales están entre p_1, \dots, p_n , junto con el símbolo especial f que denotará la *disyunción vacía*. Evidentemente la disyunción de disyunciones elementales es otra vez una disyunción elemental, en particular tenemos la convención sintáctica: $D \vee f = f \vee D = D$. La única disyunción sin letras proposicionales será f, es decir $\mathcal{D}(\emptyset) = \{f\}$. Extendemos la semántica de C. de Proposiciones a f definiendo $\bar{v}(f) = F$, para cualquier valuación v.

El sistema deductivo que llamaremos *Método de Resolución* contiene solamente dos reglas:

Regla de corte:

$$\frac{\begin{array}{c} D_1 \vee p \vee D_2 \\ \hline D_1' \vee \neg p \vee D_2' \end{array}}{D_1 \vee D_1' \vee D_2 \vee D_2'} \quad \frac{p}{\neg p} \quad f$$

Regla de simplificación:

$$\frac{D_1 \vee \pm p \vee D_2 \vee \pm p \vee D_3}{D_1 \vee D_2 \vee \pm p \vee D_3}$$

Aquí D_i ó D'_i pueden ser la disyunción vacía. Escribimos $\Gamma \vdash_{\mathcal{R}} D$ si existe una sucesión de disyunciones elementales: $D_1, D_2, \dots, D_n = D$ tal que cada D_i pertenece a Γ o viene de disyunciones anteriores por *corte* o *simplificación*.

Lema 1. Si $\Gamma \vdash_{\mathcal{R}} D$ entonces $\Gamma \models D$.

Demostración. Basta observar que si v hace verdaderas las dos premisas de la regla de corte, debe verificar alguna entre D_1, D_2, D_1', D_2' , de lo contrario debería verificar a p y a $\neg p$ que es el absurdo. Por lo tanto, v hace verdadera a $D_1 \vee D_1' \vee D_2 \vee D_2'$. Igualmente es obvio que la regla de simplificación preserva verdad. ■

El converso del lema anterior no vale (ejercicio 1). Sin embargo, probaremos otra forma de completitud. Γ es *satisfacible* si alguna valuación hace verdaderas todas sus fórmulas, de lo contrario es *insatisfacible*. Demostremos que Γ es insatisfacible si y solo si $\Gamma \vdash_{\overline{R}} f$.

Lema 2. *Sea $\Gamma \subseteq \mathcal{D}(p_1, \dots, p_n)$, $n \geq 1$, Γ insatisfacible, entonces para cada valuación v de p_1, \dots, p_{n-1} existe D_v que no contiene p_n tal que $\Gamma \vdash_{\overline{R}} D_v$ y $\overline{v}(D_v) = F$ (D_v puede ser f).*

Demuestração. Si existe $D \in \Gamma \cap \mathcal{D}(p_1, \dots, p_{n-1})$ tal que $\overline{v}(D) = F$ basta tomar $D_v = D$. Suponga ahora que para todo $D \in \Gamma \cap \mathcal{D}(p_1, \dots, p_{n-1})$ se tiene $\overline{v}(D) = V$.

1. Sea v_1 la extensión de v a p_1, \dots, p_n definida por $v_1(p_n) = V$. Como Γ es insatisfacible, existe $D \in \Gamma$ tal que $\overline{v}_1(D) = F$. Por hipótesis, D debe contener a p_n . Como $\overline{v}_1(D) = F$ y $\overline{v}_1(p_n) = V$ entonces p_n solo puede aparecer *negada* en D . Usando simplificación,

$$D \vdash_{\overline{R}} D_1 \vee \neg p_n \vee D_2$$

donde $\neg p_n$ ocurre sólo una vez y $\overline{v}_1(D_1 \vee \neg p_n \vee D_2) = \overline{v}_1(D) = F$. Por lo tanto, $\overline{v}(D_1 \vee D_2) = \overline{v}_1(D_1 \vee D_2) = F$. En conclusión:

$$\Gamma \vdash_{\overline{R}} D_1 \vee \neg p_n \vee D_2 \text{ con } \overline{v}(D_1 \vee D_2) = F \quad (1)$$

2. Ahora, sea v_1' la extensión de v a p_1, \dots, p_n donde $v_1'(p_n) = F$. Arguyendo como antes, debe existir D' tal que $v_1'(D') = F$ y por lo tanto D' debe contener p_n no negada. Usando simplificación, reducimos las ocurrencias de p_n a una y tenemos,

$$\Gamma \vdash_{\overline{R}} D_1' \vee p_n \vee D_2' \text{ con } \overline{v}(D_1' \vee D_2') = F \quad (2)$$

3. De (1) y (2), por la regla de Corte, obtenemos

$$\Gamma \vdash_{\overline{R}} D_1 \vee D_1' \vee D_2 \vee D_2'$$

con $\overline{v}(D_1 \vee D_1' \vee D_2 \vee D_2') = F$ y $(D_1 \vee D_1' \vee D_2 \vee D_2') \in \mathcal{D}(p_1, \dots, p_{n-1})$. Tómese $D_v = D_1 \vee D_1' \vee D_2 \vee D_2'$, note que D_v puede ser f . ■

Corolario 3. *Si $\Gamma \subseteq \mathcal{D}(p_1, \dots, p_n)$ es insatisfacible, existe $\Gamma' \subseteq \mathcal{D}(p_1, \dots, p_{n-1})$ insatisfacible y tal que $\Gamma \vdash_{\overline{R}} \Gamma'$.*

Demostración. Tome $\Gamma' = \{ D_v \mid v \text{ es valuación de } p_1, \dots, p_{n-1} \}$ donde D_v es la disyunción asociada a v por el lema anterior. Entonces $\Gamma \vdash_R \Gamma'$ y Γ' es insatisfacible ya que cada uno de sus elementos es falsificado por alguna valuación. ■

Teorema 4. Γ es insatisfacible si y solo si $\Gamma \vdash_R f$.

Demostración. Si $\Gamma \vdash_R f$ entonces $f \in \Gamma$ ó $\Gamma \vdash_R \{p, \neg p\}$ pues la única forma de introducir f es por la regla de corte. En el primer caso Γ es obviamente insatisfacible, en el segundo : $\Gamma \models p, \neg p$ por el lema 1, y Γ no puede ser satisfacible.

Conversamente, suponga que $\Gamma \subseteq \mathcal{D}(p_1, \dots, p_n)$ es insatisfacible. Aplicando el corolario 3, n veces, tenemos que

$$\Gamma \vdash_R \Gamma_{n-1} \vdash_R \Gamma_{n-2} \vdash_R \dots \vdash_R \Gamma_1 \vdash_R \Gamma_0 = \emptyset$$

donde $\Gamma_i \subseteq \mathcal{D}(p_1, \dots, p_n)$ y cada Γ_i es insatisfacible. En particular $\Gamma \vdash_R \Gamma_0$ donde $D_0 \subseteq \mathcal{D}(\emptyset)$ es decir $\Gamma_0 = \{f\}$. Por tanto $\Gamma \vdash_R f$. ■

Dados Σ y α sea $\Gamma_\Sigma, \neg \alpha$ un conjunto de disyunciones elementales equivalente a $\Sigma \cup \{\neg \alpha\}$, entonces el teorema anterior nos proporciona un método deductivo, pues :

$$\begin{aligned} \Sigma \vdash \neg \alpha &\text{ si y solo si } \Sigma \cup \{\neg \alpha\} \text{ es insatisfacible} \\ &\text{ si y solo si } \Gamma_\Sigma, \neg \alpha \text{ es insatisfacible} \\ &\text{ si y solo si } \Gamma_\Sigma, \neg \alpha \vdash_R f \end{aligned}$$

Ejemplo

Demostrar:

$$(p \supset q) \supset r, e \supset p \vdash \neg p \wedge (q \supset r)$$

Hallamos FNC's para las premisas y la negación de la conclusión :

$$\begin{aligned} (1) \quad (p \supset q) \supset r &\text{ eq } \neg(p \supset q) \vee r \text{ eq } (p \wedge \neg q) \vee r \\ &\text{ eq } (p \vee r) \wedge (\neg p \vee r) \text{ eq } \{p \vee r, q \vee r\} \end{aligned}$$

$$(2) r \supset p \text{ eq } \neg r \vee p \text{ eq } \{\neg r \vee p\}$$

$$(3) \neg(p \wedge (q \supset r)) \text{ eq } \neg p \vee \neg(q \supset r)$$

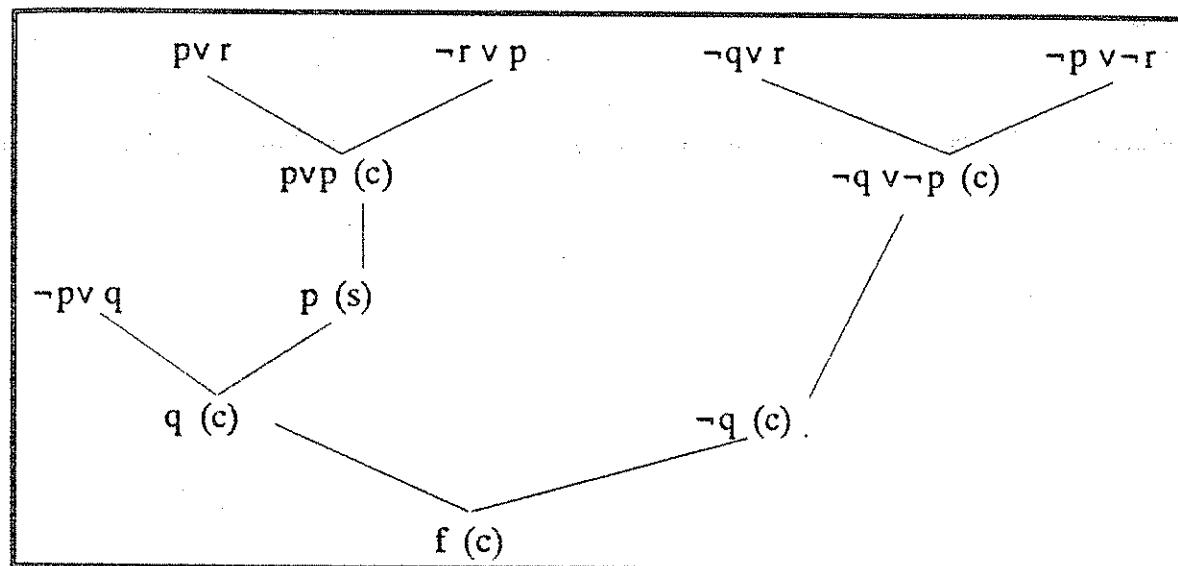
$$\text{eq } \neg p \vee \neg(q \wedge \neg r)$$

$$\text{eq } (\neg p \vee q) \vee (\neg p \vee \neg r) \text{ eq } \{\neg p \vee q, \neg p \vee \neg r\}$$

Debemos demostrar entonces que

$$\{p \vee r, \neg q \vee r, \neg r \vee p, \neg p \vee q, \neg p \vee \neg r\} \vdash_{\overline{R}} f.$$

El árbol siguiente describe una deducción



Ejercicios

1. Demuestre que $p \not\vdash_{\overline{R}} p \vee q$.
2. Demuestre que la regla de simplificación no es superflua. Es decir, demuestre que el teorema 4 no vale si se supone que la única regla del sistema $\vdash_{\overline{R}}$ es la de corte. (considere $\{q \vee q, \neg q \vee \neg q\}$)

3. Use el Método de Resolución para demostrar:

$$p \supset (q \supset r), (q \wedge p) \vee s \mid\mid \neg r \supset s$$

- 4.* Diseñe un programa de computador tal que si Γ es un conjunto insatisfacible de disyunciones elementales, produzca una deducción de $\Gamma \mid\mid_{\mathcal{R}} f$. Es decir el programa debe ser capaz de aplicar corte (y simplificación) repetidamente, agotando todas las posibilidades de aplicación.
5. Dado un conjunto finito Γ de disyunciones elementales defina:

$$\text{Simp}(\Gamma) = \{D \mid D \text{ resulta por corte de dos formas de } \Gamma\}$$

Ahora defina:

$$\Gamma_0 = \text{Simp}(\Gamma)$$

$$\Gamma_{n+1} = \text{Simp}(\text{Cort}(\Gamma_n)) \cup \Gamma_n.$$

Demuestre que existe N tal que $\Gamma_N = \Gamma_{N+1} = \dots$ y use este hecho para dar un algoritmo que determine si $\Gamma \mid\mid_{\mathcal{R}} f$ o no.

6. Demuestre que las reglas de Corte y simplificación son deducibles de A1-A9 + MP. Utilice éste hecho para dar otra prueba de la completitud del Cálculo de proposiciones.

Capítulo 6

Compacidad Conjuntos infinitos de premisas

Sea Γ un conjunto infinito de fórmulas y v una valuación definida en todas las letras proposicionales que ocurren en fórmulas de Γ , y *satisface* Γ si hace verdaderas todas las fórmulas, Γ es *satisfactible* si existe una valuación que lo satisface. Suponga que $\Gamma \models \alpha$ es decir, toda la valuación que satisface a Γ satisface a α . ¿Podemos concluir que $\Gamma \vdash \alpha$?

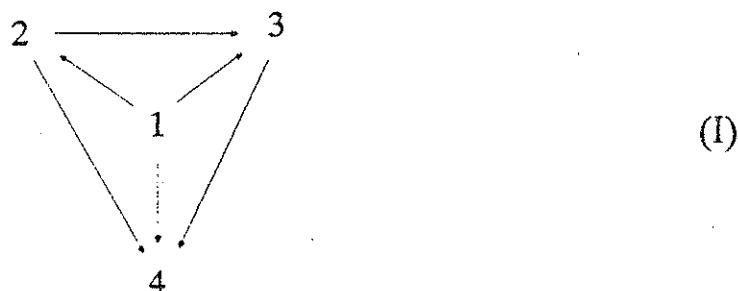
No podemos aplicar aquí el corolario al Teorema de Completitud pues tenemos infinitas premisas que no podemos reunir en una sola fórmula. Por otra parte, si fuese cierto que $\Gamma \vdash \alpha$, como una deducción formal solo puede utilizar finitas premisas, resultaría que α depende solamente de una parte finita de Γ !

A pesar de todo, la respuesta a la anterior pregunta es afirmativa como consecuencia de una muy importante propiedad del Cálculo de Proposiciones, el llamado *Teorema de Compacidad* que demostraremos en esta sección(capítulo). Para ello necesitaremos un resultado sobre grafos que tiene su propio interés.

6.1 Lema de Konig

Un *grafo dirigido* es una pareja $g = (A, R)$ en donde A es un conjunto de objetos llamados *vértices* y R es una relación binaria en A (es decir $R \subseteq A \times A$). Un grafo dirigido finito puede representarse geométricamente como un diagrama de puntos y flechas (de ahí su nombre).

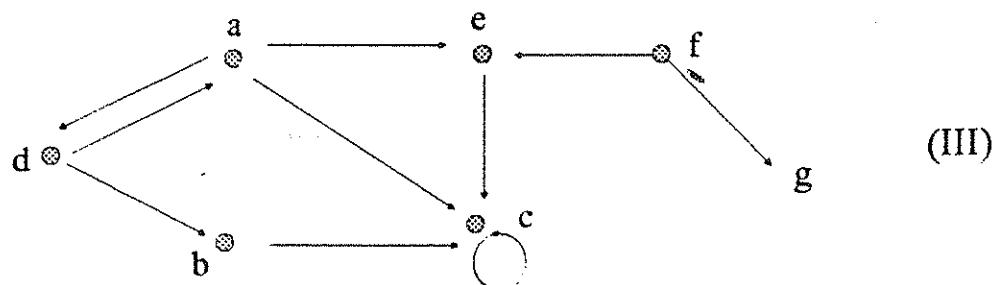
Cada vértice $x \in A$ se representa por un punto en el espacio y cada pareja $(a, b) \in R$ se representa por una *flecha* que parte de a y llega a b . Así, si $G = (\{1, 2, 3, 4\},)$ tenemos la representación gráfica :



Un interesante grafo infinito es

$$G = (\mathbb{N}^+, \{(n, m) \mid m = pn \text{ con } p \text{ primo}\}) \quad (\text{II})$$

Un grafo puede darse por su diagrama :

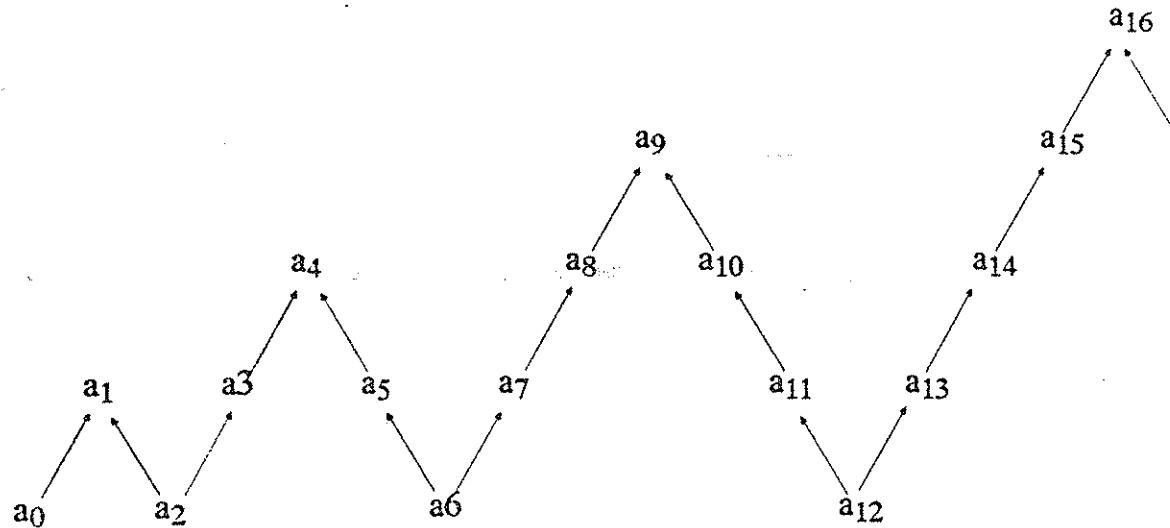


Definición 1. Un *camino* en un grafo dirigido (A, R) es una sucesión de vértices distintos (a_1, \dots, a_n) tal que $(a_i, a_{i+1}) \in R$ para $i = 1, 2, \dots, n-1$. Un

camino infinito es una sucesión enumerable de vértices distintos (a_1, a_2, \dots) tal que $(a_i, a_{i+1}) \in R$ para todo $i \in \mathbb{N}$. Un *ciclo* es un camino (a_1, \dots, a_n) tal que $(a_n, a_1) \in R$. Si $(a, a) \in R$ entonces (a) será un ciclo que llamaremos *lazo*.

En el ejemplo (I) tenemos el camino $(1, 2, 4, 5)$ pero no hay ciclos. En el ejemplo (II) tenemos el camino infinito $(1, 2, 2^2, 2^3, \dots)$ y no hay ciclos. En el ejemplo (III), (d, b, a, e, c) es un camino, (a, d, b) un ciclo y (c) un lazo.

Un grafo infinito puede no tener caminos infinitos, por ejemplo :



tiene caminos finitos arbitrariamente largos pero no tiene caminos infinitos.

Definición 2. Un *árbol* es un grafo dirigido T con un vértice distinguido a_0 , la *raíz* de T , tal que

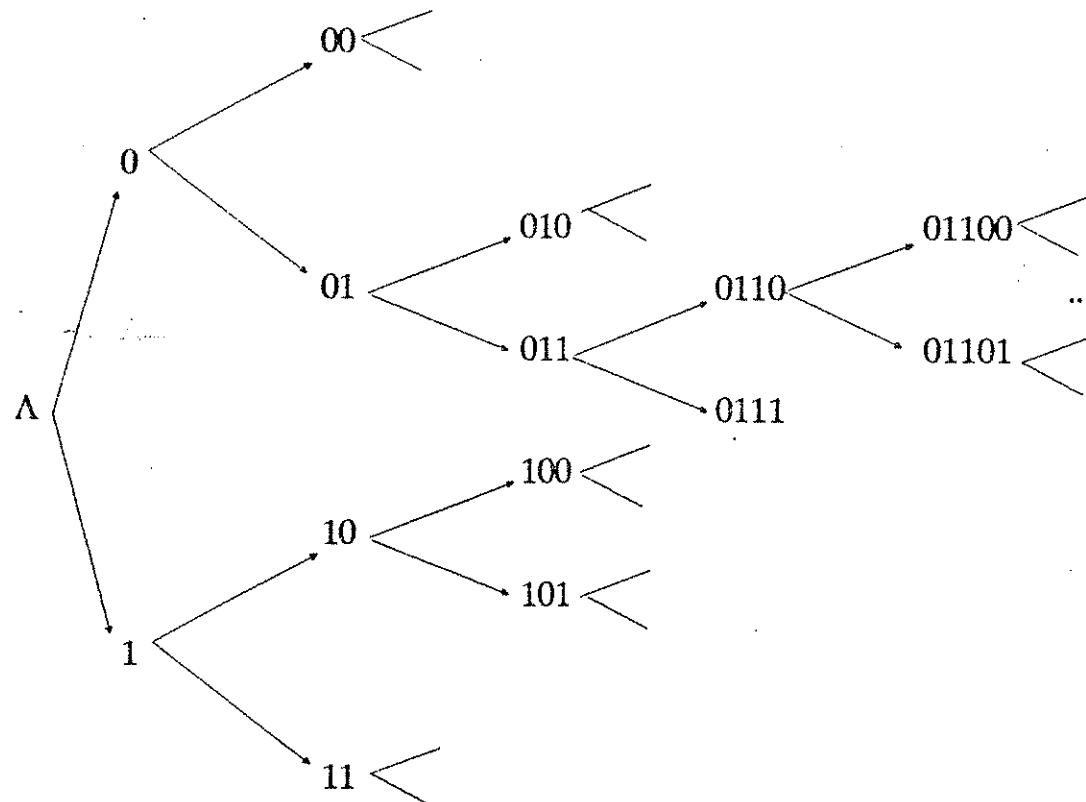
- (i) T no tiene lazos.
- (ii) Todo vértice de T distinto de a_0 es accesible desde a_0 por un único camino finito.

El *nivel* de un vértice de T es el número de flechas en el camino de a_0 al vértice (así el nivel de a_0 es 0).

Se puede demostrar que en un árbol a cada vértice, excepto a_0 , llega exactamente una flecha. Sin embargo de cada vértice puede salir cualquier número de flechas, o ninguna. Si hay una flecha de \underline{a} a \underline{b} se dice \underline{b} es un *sucesor inmediato* de \underline{a} . Un árbol es de *ramificación finita* si todo vértice \underline{a} tiene un número finito de sucesores inmediatos.

Ejemplo

Sea $T = (\{0,1\}^*, R)$, donde $\{0,1\}^*$ es el conjunto de sucesiones finitas (incluyendo la sucesión vacía) de 0's y 1's, y en donde sólo hay flechas de $s_1 \dots s_n 0$ a $s_1 \dots s_n 0$ y a $s_1 \dots s_n 1$:



Entonces T es un árbol de ramificación finita. El ejemplo (II) también es un árbol, con raíz 1, pero no es de ramificación finita.

Lema de König. *Sea T un árbol infinito de ramificación finita entonces T tiene un camino infinito.*

Demostración. Como una infinidad de vértices son accesibles de la raíz a_0 hay una infinidad de caminos finitos distintos que parten de a_0 . Sean b_1, \dots, b_k los sucesores inmediatos de a_0 (que son una cantidad finita por hipótesis); uno de los b_i debe pertenecer a una infinidad de tales caminos (de lo contrario solo finitos caminos partirían de a_0). Sea a_1 tal b_i , entonces de a_1 parten infinitos caminos. Repitiendo el argumento, obtenemos por inducción una sucesión de vértices del árbol :

$$a_0, a_1, a_2, \dots, a_n, \dots$$

tal que a_{n+1} es sucesor inmediato de a_n y de cada a_n parten infinitos caminos finitos. Evidentemente éste es un camino infinito. ■

Ejercicios

1. Determine si los grafos siguientes son o no son árboles en caso afirmativo diga si son de ramificación finita o no.
 - a. $(\mathbb{N}, <)$
 - b. $(\mathbb{N}, \{(n, n+1) | n \in \mathbb{N}\})$
 - c. $(\mathbb{N}, |)$ ($\underline{a} | \underline{b} \Leftrightarrow \underline{a}$ divide a \underline{b} y $\underline{a} \neq \underline{b}$)
 - d. $(\{1, 2, 4, 5, 6, 8, 9\}, |)$
 - e. $(\mathbb{N}, \{(n, m) | n < m < n^2 + 1\})$
2. a. Demuestre que si x es un vértice diferente de la raíz en un árbol T , debe llegar exactamente una flecha a x .

b. Demuestre que un árbol no tiene ciclos.
3. Sea $G = (A, R)$ un grafo dirigido, demuestre que si $\underline{a}, \underline{b} \in G$ y existe una sucesión $\underline{a} = a_1, a_2, \dots, a_n = \underline{b}$ tal que $(a_i, a_{i+1}) \in R$, entonces existe un camino de \underline{a} a \underline{b} .
4. Demuestre que el Lema de König es falso si el árbol no es de ramificación finita.

5. Generalice el Lema de König a grafos arbitrarios:

Sea (A, R) un grafo dirigido tal que $\{b \mid (a, b) \in R\}$ es finito para todo $a \in A$.

Sea $a_0 \in A$ tal que una infinidad de vértices son accesibles de a_0 , entonces existe un camino infinito en T que parte de a_0 .

6.* Explique porqué el método indicado en la prueba del Lema de König para “construir” una rama infinita, no proporciona realmente un algoritmo para generar dicha rama, que se pueda programar automáticamente (suponiendo que el árbol está dado de manera que es posible recorrerlo mecánicamente).

6.2 Compacidad

El Lema de König de la sección anterior sirve para demostrar el siguiente teorema, uno de los más importantes de la Lógica Matemática, para conjuntos enumerables de fórmulas. De hecho el teorema es equivalente al Lema de König (Ejercicio 3). |

Teorema de Compacidad. *Sea Γ un conjunto infinito de fórmulas tal que todo subconjunto finito de Γ es satisfactible, entonces Γ es satisfactible.*

Demostración. Damos una demostración para Γ infinito enumerable. El teorema vale también para conjuntos no enumerables, pero su demostración requiere métodos más avanzados. Como cada fórmula de Γ tiene un número finito de letras proposicionales, el total de letras que ocurren en las fórmulas de Γ es a lo sumo enumerable y podemos suponer $\Gamma \subseteq \mathfrak{I} (p_1, p_2, p_3, \dots)$. Suponga que cada subconjunto finito de Γ es satisfactible y defina $\Gamma_n = \Gamma \cup \mathfrak{I}(p_1, \dots, p_n)$, entonces tenemos

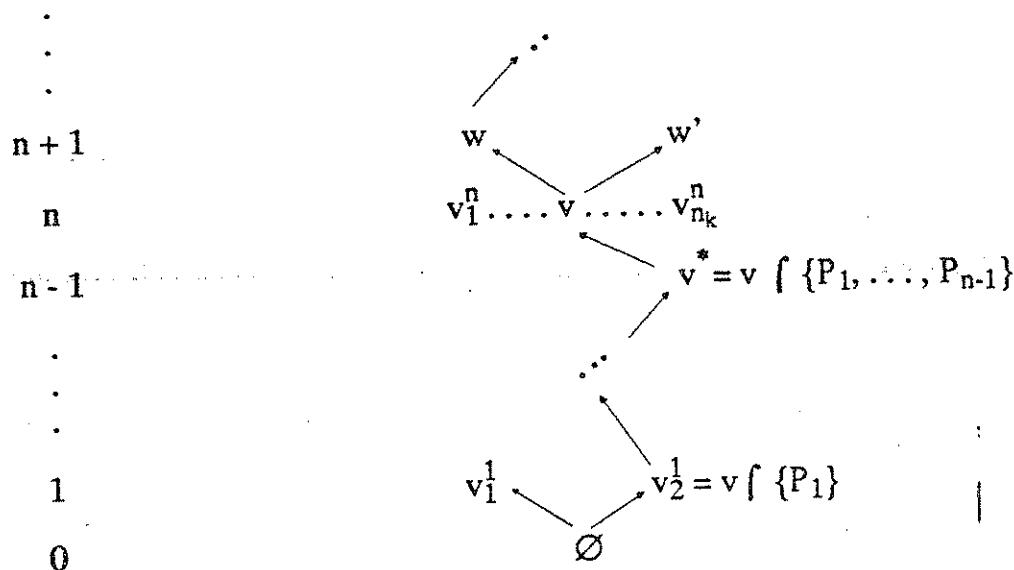
$$(i) \quad \Gamma_1 \subseteq \Gamma_2 \subseteq \dots \subseteq \Gamma_n \subseteq \Gamma_{n+1} \subseteq \dots$$

∞

$$(ii) \quad \Gamma = \bigcup_{i=1}^{\infty} \Gamma_i$$

Como el número máximo de fórmulas no equivalentes de Γ_n es $2^{(2^n)}$, existe un conjunto finito $\Gamma_n' \subseteq \Gamma_n$ tal que $\Gamma_n' \equiv \Gamma_n$; por hipótesis Γ_n' es satisfactible y fortiori Γ_n debe ser satisfactible. Sea $S_n = \{v_1^n, \dots, v_{k_n}^n\}$ el conjunto de valuaciones de $\{p_1, \dots, p_n\}$ que satisface a Γ_n , entonces S_n es finito y no vacío.

Formamos un árbol infinito de la manera siguiente. Los vértices de nivel $n \geq 1$ son las valuaciones en S_n . Hay una flecha de $v \in S_n$ a $w \in S_{n+1}$ si w extiende a v , es decir w coincide con v en p_1, \dots, p_n . En el nivel 0 hay un vértice, \emptyset , con flechas de \emptyset a los elementos de S_1 .



Si $v \in S_n$, su restricción a $\{p_1, \dots, p_{n-1}\}$, v^* , satisface a Γ_{n-1} ya que $\Gamma_{n-1} \subseteq \Gamma_n$, por lo tanto $v^* \in S_{n-1}$ y evidentemente hay una flecha de v^* a v . Siguiendo de esta manera, es fácil concluir que debe haber un camino único de \emptyset a v : $(\emptyset, v \mid \{p_1\}, v \mid \{p_1, p_2\}, \dots, v \mid \{p_1, \dots, p_{n-1}\}, v)$. Además de cada vértice parten a lo sumo dos flechas. Por el Lema de Konig debe existir entonces una cadena infinita:

$$\emptyset \rightarrow v_1 \rightarrow v_2 \rightarrow v_3 \rightarrow \dots \quad v_i \in S_i.$$

Defina $v : \{p_1, p_2, \dots\} \rightarrow \{V, F\}$ por $v(p_i) = v_i(p_i)$. Como v_n extiende a v_i para $i \leq n$ tenemos que $v(p_i) = v_i(p_i) = v_n(p_i)$ para $i = 1, 2, \dots$, es decir v coincide con v_n en $\{p_1, \dots, p_n\}$, por lo tanto $v(\alpha) = v_n(\alpha) = v$ para las fórmulas de Γ_n .

Como cada $\alpha \in \Gamma$ está en algún Γ_n , tenemos que v satisface todas las fórmulas de Γ . ■

Corolario 1 (Completitud Generalizada). *Sea Γ un conjunto infinito, entonces $\Gamma \vdash \alpha$ si y solo si $\Gamma \models \alpha$.*

Demuestra. De izquierda a derecha es sencillo; si $\Gamma \vdash \alpha$ debe existir un subconjunto finito $\Gamma' \subseteq \Gamma$ tal que $\Gamma' \vdash \alpha$, pues una deducción solo usa un número finito de premisas. Por validez: $\gamma' \models \alpha$, y con mayor razón $\Gamma \models \alpha$.

Suponga ahora que $\Gamma \models \alpha$ pero $\Gamma \not\vdash \alpha$, entonces para cualquier subconjunto finito $\Gamma' \subseteq \Gamma$ debemos tener $\Gamma' \not\vdash \alpha$, y por completitud $\Gamma' \not\models \alpha$, lo cual implica que $\Gamma' \cup \{\neg \alpha\}$ es satisfactible. Entonces cualquier subconjunto finito de $\Gamma \cup \{\neg \alpha\}$ mismo también lo es. Por lo tanto $\Gamma \not\models \alpha$, que contradice la hipótesis. ■

El teorema de compacidad tiene otras consecuencias interesantes. Recuerde que Γ es *consistente* si no se puede deducir formalmente una contradicción usando premisas de Γ .

Corolario 2. *Sea Γ un conjunto infinito, Γ es consistente si y solo si es satisfactible.*

Demuestra. Si Γ es satisfactible, todo lo que de él se deduzca debe serlo por tanto Γ debe ser consistente. Suponga Γ es consistente y por el Teorema de Consistencia para conjuntos finitos debe ser satisfactible. Por compacidad, Γ es satisfactible. ■

Ejercicios

1. Demuestre que si $\Gamma \models \alpha$ pero $\Gamma' \not\models \alpha$, para todo subconjunto propio de Γ , entonces Γ es finito.
2. Sea $\alpha_1, \alpha_2, \dots$ una colección infinita de fórmulas tal que toda valuación v satisface alguna α_n (puede ser distinta para cada v). Demuestre que se pueden escoger finitas fórmulas $\alpha_{i_1}, \dots, \alpha_{i_k}$ tales que $\models \alpha_{i_1} \vee \dots \vee \alpha_{i_k}$.

3. Pruebe el Lema de König usando el Teorema de Compacidad.

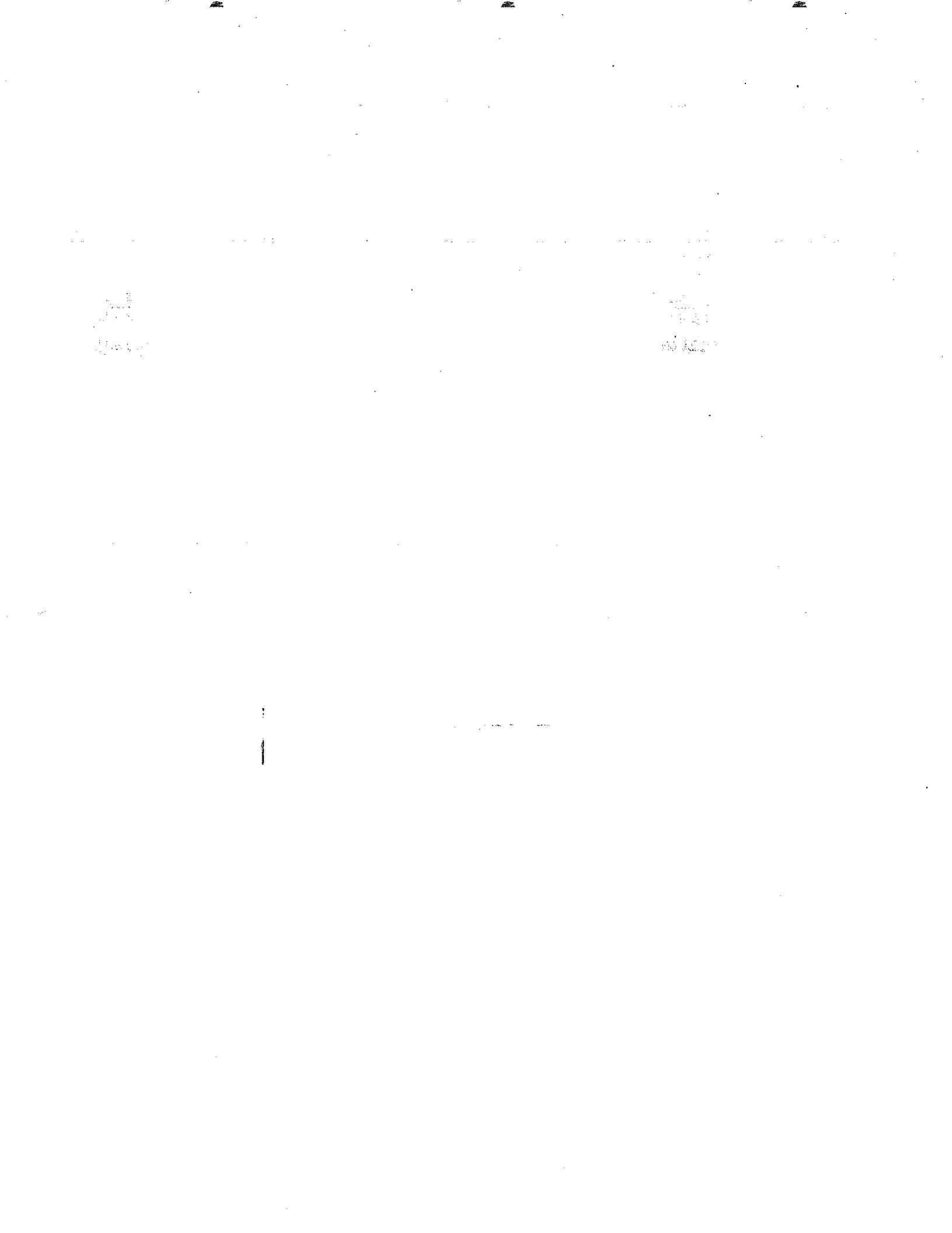
Ayuda: Dado el árbol T , tome una letra proposicional p_a para cada vértice a de T , y forme el siguiente conjunto de fórmulas:

$$\Gamma = \{ \bigvee_{a \in T_n} p_a \mid n \in \mathbb{N} \} \cup \{ p_b \supset p_a \mid (a, b) \in R \},$$

donde T_n es el nivel n de T y \bigvee denota disyunción exclusiva.

Demuestre que Γ es satisfactible y que para cualquier valuación v satisfaga a Γ . $\{a \mid p_a = v\}$ forma una cadena de T que pasa por todos los niveles.

4. Demuestre que Completitud Generalizada implica Compacidad.
5. Un grafo es 4-colorable si cada uno de sus vértices se puede colorear de manera que dos vértices conectados por una flecha tengan colores distintos. Demuestre que si todo subgrafo finito de un grafo infinito G es 4-colorable (Un teorema famoso de Appel y Haken, 1976, dice que todo grafo finito *planar*, es decir que se puede representar en el plano sin que se crucen sus flechas, es 4-colorable; este ejercicio muestra que lo mismo sucede para grafos infinitos planares).



Segunda parte

Cálculo de
Predicados



Capítulo 1

Simbolización

El Cálculo de Proposiciones considera solamente combinaciones de proposiciones por medio de los conectivos, sin entrar a analizar la estructura interna de los enunciados atómicos. De esta manera sólo refleja un aspecto parcial de las conexiones lógicas que puede haber entre las proposiciones. Hay argumentos y enunciados lógicamente válidos de los cuales no puede dar cuenta el Cálculo de Proposiciones, por ejemplo el siguiente enunciado es evidéntemente válido:

“Si hay una causa de todos los fenómenos, entonces todo fenómeno tiene causa”

pero si lo simbolizamos en el Cálculo de Proposiciones lo mejor que podemos obtener es algo como: $p \supset q$, ya que el antecedente de la implicación, “Hay una causa de todos los fenómenos”, no es negación, conjunción, o disyunción, etc. de otras proposiciones, y lo mismo sucede con el consecuente: “todo fenómeno tiene causa”. Como $p \supset q$ no es una tautología, no podrá deducirse en el Cálculo de Proposiciones. Aún si forzamos la situación y leemos el consecuente como : “ser fenómeno implica tener causa”, tenemos $p \supset (q \supset r)$ que está lejos de ser una tautología.

Para entender la “lógica” de la proposición anterior es necesario analizar su estructura más profundamente. Ello será posible con la ayuda del lenguaje del *Cálculo de Predicados* que ahora introducimos.

1.1 Predicados, relaciones, cuantificadores

Las proposiciones más sencillas posibles son del tipo *sujeto-predicado*, por ejemplo:

“Sócrates es hombre” H(s)

“Sócrates es mortal” M(s)

“2 es par” P(2).

Si simbolizamos el sujeto Sócrates por la letra minúscula s, y los predicados “ser hombre”, “ser mortal”, y “ser par” por las letras mayúsculas H, M, P, respectivamente, podemos simbolizar las tres proposiciones: H(s), M(s), y P(2). Tradicionalmente se utiliza la notación prefija, es decir el predicado antes de el sujeto.

Cuando el predicado hace referencia a otros individuos como en

“Juan conoce a María” C(j,m)

“3 es mayor que 2” R(3,2)

nos apartamos de la gramática tradicional, y en lugar de suponer que el predicado es “conocer a María” o “ser mayor que 2”, consideramos que hay en cada oración dos sujetos (Juan y María, 3 y 2, respectivamente) conectados en pie de igualdad por un *predicado de dos posiciones*, es decir, una *relación binaria*. Así, si C denota la relación “- conoce a -” y R la relación “- es mayor que -”, e insistimos en utilizar notación prefija, las dos proposiciones anteriores se simbolizarán C(j,m) y R(3,2).

La misma observación vale para proposiciones que envuelven más de dos individuos (sujetos) como

“Juan conoció a María en Cali” E(j,m,c)

Nótese sin embargo que ciertos predicados ternarios son en realidad combinaciones de predicados binarios:

"Juan conoce a María y a Pedro" $C(j,m) \wedge C(j,p)$.

Consideremos proposiciones más complejas como la siguiente:

"Todos los hombres son mortales" (1)

Aunque según la gramática tradicional esta es una oración del tipo sujeto-predicado, donde el sujeto sería "todos los hombres" y el predicado : "ser mortal", éste no es el punto de vista del Cálculo de Predicados. En realidad la insistencia en considerar toda proposición categórica como del tipo sujeto-predicado constituyó un obstáculo para el desarrollo de la lógica por siglos, y fue ocasión de disputas filosóficas interminables (v.gr. el problema de los "universales"). Es claro que la propiedad de "ser mortal" se enuncia para cada uno de los individuos del conjunto de los hombres y no para el conjunto en sí. El sujeto es pues un individuo *indeterminado* que recorre el conjunto de los hombres. Usando esta idea podemos reescribir (1) de manera que este sujeto variable se haga patente:

"Para todo x : si x es hombre, entonces x es mortal"

Si introducimos el símbolo \forall , llamado *cuantificador universal*, para abreviar la expresión "para todo", y los símbolos de predicado H y M , podemos entonces expresar (1) como :

$$\forall x(H(x) \supset M(x))$$

Análogamente, la proposición:

"Algún hombre es mortal" (2)

puede reescribirse:

"Existe x tal que : x es hombre y x es mortal".

Si introducimos el *cuantificador existencial*, \exists , para abreviar "Existe _ tal que : ", la proposición (2) queda simbolizada por:

$$\exists x(H(x) \wedge M(x)).$$

En general, si A y B denotan propiedades, las cuatro *proposiciones categóricas* de Aristóteles, componente fundamental de sus silogismos, pueden simbolizarse:

- | | |
|---------------------|--------------------------------------|
| (1) Todo A es B | $\forall x(A(x) \supset B(x))$ |
| (2) Algún A es B | $\exists x(A(x) \wedge B(x))$ |
| (3) Ningún A es B | $\neg\exists x(A(x) \wedge B(x))$ |
| (4) Algún A no es B | $\exists x(A(x) \wedge \neg B(x))$. |

Aristóteles llamaba a estas sentencias universal afirmativa (1), particular afirmativa (2), universal negativa (3) y particular negativa (4), respectivamente. Note que (3) significa lo mismo que:

$$(3') \text{ Todo A es B} \quad \forall x(A(x) \supset \neg B(x)).$$

Similarmente, (4) significa lo mismo que

$$(4') \text{ No todo A es B} \quad \neg\forall x(A(x) \supset B(x)).$$

Finalmente, volvamos al primer ejemplo de este capítulo:

“Si hay una causa de todos los fenómenos, entonces todo fenómeno tiene causa”.

En primer lugar debemos averiguar cuales son los predicados fundamentales involucrados en la proposición. Tenemos un predicado de dos posiciones: “ser causa de...” y otro de una posición: “ser fenómeno”, cuya simbolización indicamos con la ayuda de variables:

$$\begin{aligned} \text{“x es causa de y”} & C(x,y) \\ \text{“x es fenómeno”} & F(x). \end{aligned}$$

La oración puede simbolizarse entonces:

$$\{\exists x(\forall y(F(y) \supset C(x,y)))\} \supset \{\forall y(F(y) \supset \exists x C(x,y))\}.$$

Uno de nuestros objetivos será mostrar que con el simbolismo introducido, expresiones lógicamente válidas como la anterior o equivalencias como la

de (3) y (3') más arriba pueden deducirse formalmente de unas pocas leyes básicas que extienden a las del Cálculo de Proposiciones, pero ésto debe esperar un poco. Terminamos con ejemplos adicionales de simbolización que el lector debe justificar.

Ejemplos

(a) Todo amigo de Juan y de Pedro es amigo de todo el mundo,

$$\begin{array}{ll} \text{"x es amigo de y"} & A(x,y) \\ \text{"Pedro"} & p \\ \text{"Juan"} & j \\ \forall x((A(x,j) \wedge A(x,p)) \supset \forall y A(x,y)) \end{array}$$

(b) Todo lo que me gusta es ilegal, inmoral o engorda,

$$\forall x(G(x) \supset (I(x) \vee M(x) \vee E(x)))$$

(en éste y los siguientes ejemplos dejamos al lector indicar el significado de los símbolos).

(c) Me gusta todo lo que es ilegal, inmoral o engorda,

$$\forall x((I(x) \vee M(x) \vee E(x)) \supset G(x)).$$

(d) Ningún número par divide a todos los demás números,

$$\neg \exists x(N(x) \wedge P(x) \wedge \forall y(N(y) \supset D(x,y))).$$

(e) "No hay un entero más grande que los demás enteros,

$$\neg \exists x(E(x) \wedge \supset y(E(y) \wedge y \neq x \supset M(x,y))).$$

(f) "Cualquiera puede inventar una disculpa",

$$\forall x(\exists y(I(x,y) \wedge D(y))).$$

(g) "El tren va de Cali a Medellín",

$$V(t,c,m).$$

(h) "Pablo se dirige velozmente hacia su casa",

$$D(p,c).$$

(i) "Todo primo mayor que 2 es impar",

$$\forall x(P(x) \wedge M(x, 2) \supset I(x)).$$

(j) "Si alguien puede hacerlo, Juan puede hacerlo",

$$\exists x(H(x)) \supset H(j).$$

(k) "Todo político honesto vota solamente por alguien diferente de si mismo",

$$\forall x(P(x) \wedge H(x) \supset \forall y(V(x,y) \supset \neg I(y,x))).$$

(l) "Perro no come perro",

$$\forall x(P(x) \supset \neg \exists y(P(y) \wedge C(x,y))).$$

Ejercicios

1. Simbolice, indicando el significado de los símbolos
 - a. Todos los pájaros cantores vuelan.
 - b. Algun pájaro cantor no vuela.
 - c. Ningún pez trepa montañas.
 - d. Todos tienen un superior.
 - e. Hay un superior para todos.
 - f. Juan no quiere a nadie.

- g. Todo el mundo quiere a alguien pero nadie quiere a todo el mundo.
- h. Nadie es inmortal.
- i. Juan no vota por ningún político.
- j. Juan sólo vota por políticos honestos.
- k. Algunos políticos no votan por sí mismos
- l. Un político no es necesariamente honesto porque todos voten por él.
- m. Todos los peces, excepto los tiburones, son amables con los niños.
- n. Todos los perros son animales, luego todas las cabezas de perro son cabezas de animal.
- o. El barbero afeita a todos aquellos que no se afeitan a sí mismos.
- p. Juan puede engañar a alguna gente todos los días, y puede engañar a toda la gente algunos días, pero no puede engañar a toda la gente todos los días (Use una relación ternaria).
- q. Si alguien está aquí entonces no está en Roma; alguien está aquí, por lo tanto nadie está en Roma.
- r. Algunos de los seguidores de Aristóteles siguen a Aquino. Ningún seguidor de Aristóteles sigue a los idealistas. Por tanto, ningún seguidor de Aquino es idealista.
- s. Por todo punto exterior a una recta, pasa una paralela a la recta.
- t. Algún país más fuerte que todos sus vecinos declarará la guerra todos los demás países.
- u. Todo primo mayor que todos los números perfectos es un primo de Fermat.
- v. Si hay un justo, Sodoma será salvada.
2. Suponga que $R(x,y)$ significa “ x es padre de y ” y $M(x,y)$ significa “ x es madre de y ”. c denota a Juan y d a Pedro. Las siguientes sentencias expresan relaciones familiares entre Juan y Pedro diga cuáles son estas:
- a. $\exists x \exists y (M(x,c) \wedge M(x,d) \wedge R(y,c) \wedge R(y,d))$
- b. $\exists x \exists y \exists z (R(x,y) \wedge R(x,z) \wedge M(y,c) \wedge R(z,d))$
- c. $\exists x (R(c,x) \wedge M(x,d))$
- d. $\exists x (R(c,x) \wedge R(x,d))$

3. Simbolice los siguientes argumentos ¿son intuitivamente válidos?

- a. Hay un barbero en Plutón que afeita a todos los platonianos que no se afeitan a sí mismos

Hay un barbero en Plutón que no es platoniano o se afeita a sí mismo.

- b. Todos los caballos son animales

Todas las colas de caballo son colas de animal.

Nota (sobre los silogismos)

En la edad media se simbolizó con las letra A, I, E, O, tomadas de las palabras latinas “affirmo nego”, los cuatro tipos de proposiciones categóricas:

A	ff	Todo P es Q	(universal afirmativa)
I	rmo	Algún P es Q	(particular afirmativa)
n	E	Ningún P es Q	(universal negativa)
g	O	Algún P no es Q	(particular negativa).

Simbolicemos por PQ cualquiera de las anteriores formas categóricas. Los predicados P y Q son llamados *términos* de PQ. Un *silogismo* consiste en una sucesión de tres proposiciones categóricas construidas con tres términos de manera que cualesquiera dos de ellas tengan exactamente un término común. Las siguientes estructuras dan todas las posibilidades según la posición relativa de los términos en cada proposición, y se llaman *modos* del silogismo: *primero, segundo, tercero, y cuarto modo*, respectivamente:

- | | | | | |
|----|-----------|-----------|----|-----------|
| a) | PM | PM | MP | MP |
| b) | <u>MQ</u> | <u>QM</u> | MQ | <u>QM</u> |
| c) | PQ | PQ | PQ | PQ |

(a) es la *premisa mayor*, (b) es la *premisa menor* y (c) es la *conclusión*. Similarmente, P, M y Q son el *término mayor*, el *término medio* y el *término menor* del silogismo, respectivamente. Como cada una de las tres sentencias puede pertenecer a una de las formas A, E, I, O, hay $4^3 = 64$ silogismos en cada modo. Los 4 modos dan entonces en total $4^4 = 256$ silogismos.

Ejemplos:

I	MP	Algún M es P	$\exists x(M(x) \wedge P(x))$	
E	MQ	Ningún M es Q	$\neg \exists x(M(x) \wedge Q(x))$	3 ^{er} modo
O	PQ	Algún P no es Q	$\exists x(P(x) \wedge \neg Q(x))$	
A	PM	Todo P es M	$\forall x(P(x) \supset M(x))$	
O	MQ	Algún M no es Q	$\exists x(M(x) \wedge \neg Q(x))$	1 ^{er} modo
O	PQ	Algún P no es Q	$\exists x(P(x) \wedge \neg Q(x))$	

La lógica clásica mostraba que solamente 16 de los 256 silogismos son "válidos". El primer ejemplo arriba es válido, pero el segundo no lo es. Los estudiantes de lógica debían memorizar aquellos famosos versos medioe-vales que comenzaban:

B a r b a r a, C e l a r e n t, ...

debidos a William de Sherwood (1205-1268), formados por palabras sin sentido de manera que las vocales de cada palabra codificasen los silogismos válidos; los dos primeros (del primer modo) serían por ejemplo:

A	PM	Todo P es M	E	PM	Ningún P es M
A	MQ	Todo M es Q	A	QM	Todo Q es M
A	PQ	Todo P es Q	E	PQ	Ningún P es Q

1.2 Igualdad, funciones

Aunque la relación de identidad “-es igual a-” podría considerarse como una relación más, ésta recibe un tratamiento especial en el Cálculo de Predicados. Se simbolizará siempre por “=” (en notación infija: “ $x = y$ ”). La razón para esta distinción reside en que la identidad es la única relación que satisface la siguiente ley lógica, llamada *Principio de Leibnitz*:

$$x = y \supset (P(x) \leftrightarrow P(y))$$

cualquiera que sea el predicado P . Otras leyes que la igualdad cumple son:

“Dos cosas iguales a una tercera son iguales entre sí”

$$\forall x \forall y (\exists z (x=z \wedge y=z) \supset x=y)$$

que también podría simbolizarse:

$$\forall x \forall y \forall z ((x=z \wedge x=z) \supset x=y)$$

o el *principio de identidad*: $\forall x (x=x)$. Más adelante veremos que todas las propiedades de la igualdad pueden derivarse del Principio de Leibnitz y el Principio de Identidad.

La igualdad permite expresar proposiciones con contenido numérico, por ejemplo:

(a) “Pedro es el único hermano de Juan”

$$H(p,j) \wedge \forall x (H(x,j) \supset x=p).$$

(b) “Hay por lo menos dos políticos honestos”

$$\exists x \exists y (\neg(x=y) \wedge P(x) \wedge P(y) \wedge H(x) \wedge H(y)).$$

(c) “Hay exactamente dos políticos honestos”

$$\exists x \exists y (\neg(x=y) \wedge P(x) \wedge H(x) \wedge P(y) \wedge H(y) \wedge \forall z (P(z) \wedge H(z) \supset (z=x \wedge z=y))).$$

(d) "Hay a lo sumo dos políticos honestos:

Podríamos entender esta oración como la disyunción: "no hay políticos honestos o hay exactamente un político honesto o hay exactamente dos políticos honestos", y proceder a simbolizar cada parte; sin embargo, hay una simbolización más sencilla:

$$\exists x \exists y \forall z (P(z) \wedge H(z) \supset (z=x \vee z=y)),$$

la proposición dice que el conjunto de los políticos honestos está incluido en un conjunto que a lo sumo tiene 2 elementos.

Se acostumbra a utilizar la abreviación $\exists !x P(x)$ para la fórmula

$$\exists x (P(x) \wedge \forall y (P(y) \supset y=x))$$

que expresa "existe un único x con la propiedad P". Igualmente podemos introducir, para cada natural n, fórmulas:

$$\exists^{=n} x P(x), \quad \exists^{\geq n} x P(x), \quad \exists^{\leq n} x P(x),$$

cuyo significado es "Existen exactamente n x's con la propiedad P", "Existen al menos n x's con la propiedad P" y "Existen a lo sumo n x's con la propiedad P", respectivamente.

Considera ahora una sentencia como

$$\text{"El padre de Juan conoce a Luis"} \quad (1)$$

aquí uno de los sujetos de la proposición tiene una descripción compleja: "el padre de Juan"; si lo simbolizaramos por una letra, p, la fórmula resultante: $C(p,l)$ no reflejaría la relación de paternidad de p con Juan. Una simbolización más fina utilizando $R(x,y)$ por "x es padre de y", $C(x,y)$ por "x conoce a y", sería:

$$\forall x (R(x,j) \supset C(x,l)) \text{ ó } \exists x (R(x,j) \wedge C(x,l))$$

sin embargo, ésta fórmulas no expresan el hecho implícito en la proposición original por el uso del artículo definido "el", de que *el* padre de Juan es único. Una versión más aproximada sería entonces:

$$\exists !x R(x,j) \wedge \forall x (R(x,j) \supset C(x,l)).$$

Pero hay una solución mejor, más natural y cercana al uso ordinario del lenguaje. El padre de Juan puede considerarse como la imagen de una función f definida por $f(x)$ = “el padre de x ”, entonces (1) puede simbolizarse literalmente como :

$$C(f(j),l).$$

Damos otro ejemplo:

“El cuadrado de todo número par es par”

$$\forall x (N(x) \wedge P(x) \supset P(g(x))),$$

donde $g(x)$ denota la función “el cuadrado de x ”. Si no utilizamos el símbolo de función tendríamos que introducir una relación $R(x,y)$ para expresar “ x es el cuadrado de y ” ($x = f(y)$) y tendríamos la simbolización:

$$\forall x (N(x) \wedge P(x) \supset \forall y (R(y,x) \supset P(y)))$$

que en realidad solo sería equivalente a la original en presencia de la fórmula que expresa que R es una función:

$$\forall x \exists y (R(y,x) \wedge \forall z (R(z,x) \supset z=y))$$

Los símbolos de función son fundamentales en Matemáticas; considere, por ejemplo la oración:

“Si dos números son menores que 1 entonces su producto es menor que su suma”

$$\forall x \forall y (R(x,1) \wedge R(y,1) \supset R(x \cdot y, x+y)) \quad (2)$$

Aquí hemos utilizado directamente la notación infija, en lugar de la más engorrosa $.(x,y)$, $+(x,y)$ para suma y producto. Por otra parte los símbolos de función no son necesarios. Si las relaciones $S(x,y,z)$, $M(x,y,z)$ denotan las relaciones $x+y=z$, $x \cdot y =z$, respectivamente; los símbolos $+$ y \cdot pueden eliminarse en presencia de las sentencias:

$$\forall x \forall y \exists z (S(x,y,z))$$

$$\forall x \forall y \exists z \forall w (S(x,y,z) \wedge S(x,y,w) \supset z=w) \quad (*)$$

que dicen que la relación S es una función de sus dos primeras variables en la tercera, y las análogas para M . Por ejemplo, la ley commutativa, $\forall x \forall y (x+y=y+x)$ se simbolizaría:

$$\forall x \forall y \forall z (S(x,y,z) \supset S(y,x,z)).$$

Otras simbolizaciones equivalentes (en presencia de *) serían:

$$\forall x \forall y \forall z (S(x,y,z) \Leftrightarrow S(y,x,z))$$

$$\forall x \forall y \forall z (S(x,y,z) \wedge S(y,x,z)).$$

La oración (2) arriba se escribiría:

$$\forall x \forall y [R(x,1) \wedge R(y,1) \supset \forall z \forall w (M(x,y,z) \wedge S(x,y,w) \supset R(y,w))].$$

La asociatividad de la suma: $(x+y)+z = x+(y+z)$, podría escribirse:

$$\forall r \forall s \forall t [S(x,y,r) \wedge S(r,z,s) \wedge S(y,z,t) \supset S(x,t,s)]$$

o también:

$$\exists r \exists s \exists t [S(x,y,r) \wedge S(r,z,s) \wedge S(y,z,t) \wedge S(x,t,s)].$$

Ejercicios

1. Simbolice sin usar abreviaciones:

- Existen por lo menos tres partículas elementales.
- Existe exactamente un primo par.
- Existe más de un primo par.
- Por cada par de puntos distintos pasa una y sólo una recta.

- e. Cada recta pasa por dos puntos distintos.
- f. Dos rectas son paralelas si son iguales o si no tienen ningún punto en común.
- g. Por cada punto exterior a una recta pasa una y sólo una paralela.
2. Simbolice, utilizando símbolos de función si ello es apropiado:
- Nadie puede ser su propio padre.
 - Cualquier hermano del padre de Juan es tío de Juan.
 - f es una función uno a uno pero no sobreyectiva.
3. Exprese en notación prefija por medio de las funciones suma y multiplicación solamente $(x+y)^2 = x^2 + 2xy + y^2$.
4. Elimine los símbolos de función usando los predicados $S(x,y,z)$ por $x+y=z$, $M(x,y,z)$ por $x \cdot y=z$.
- $\forall x (x^2 = x \supset x=0 \vee x=1)$
 - $\exists x (x +(x+(x+x)) = x)$
 - $x \cdot (y+z) = (x \cdot y) + (x \cdot z)$
 - $\forall x \forall y \forall z (x+y = x+z \supset y=z)$
 - $\forall x (x \neq 0 \supset \exists y (x \cdot y = 1))$
 - $(x+y)^2 = x^2 + 2xy + y^2$
- Explique el significado de las sentencias:
- $\forall x \forall y (M(x,y,5) \supset x=5 \vee x=1)$
 - $\forall x \exists r \exists s \exists t \exists u [S(x,x,r) \wedge M(r,r,s) \wedge M(x,x,t) \wedge S(t,t,u) \wedge u \neq 3]$.
5. Simbolice utilizando la definición ε , δ , y símbolos de función:
"cos x es continua en $x=5$ ".
6. Simbolice :
"El padre del padre de Pedro es hijo del padre de Luis".
- Utilizando un símbolo de función para "el padre de x ".
 - Sin utilizar símbolo de función para "el padre de x ".

1.3 Conceptos primitivos y definidos

Una de las ventajas de la simbolización es que permite establecer de una manera precisa las relaciones de definibilidad entre ciertos conceptos. Para determinar claramente el significado de un discurso, debemos tener en mente un *universo* fijo, es decir un conjunto de individuos sobre el cual "varían" las variables, y al cual se refieren los cuantificadores, además de ciertos conceptos *primitivos* que pueden consistir en predicados (relaciones), funciones e individuos especiales en dicho universo. A partir de los conceptos primitivos otros conceptos pueden ser *definidos*. El simbolismo del Cálculo de Predicados permite expresar estas definiciones de una manera exacta.

Ejemplo 1

Universo: Conjunto de los seres humanos (vivos actualmente)

Predicados Primitivos:	x es hombre	A(x)
	x es mujer	B(x)
	x es madre de y	M(x,y)
	x es padre de y	P(x,y).

Nótese que no introducimos un símbolo de predicado para el universo, éste queda implícito. Usando los símbolos indicados para los predicados primitivos, podemos definir nuevos predicados:

$$x \text{ es hijo de } y: \quad A(x) \wedge (M(y,x) \vee P(y,x)) \quad (1)$$

$$x \text{ es abuela paterna de } y: \quad \exists z(M(x,z) \wedge P(z,y)) \quad (2)$$

$$x \text{ no tiene hijas:} \quad \neg \exists y(B(y) \wedge (M(x,y) \vee P(x,y))) \quad (3)$$

$$x \text{ es hermano de } y \\ (\text{por padre y madre}): \quad A(x) \wedge \neg(x=y) \exists z((M(z,x) \wedge M(z,y)) \wedge \exists w(P(w,x) \wedge P(w,y))) \quad (4)$$

Considere ahora la fórmula (3) que dice "x no tiene hijas", contiene un sujeto indeterminado x sobre el cual la fórmula afirma algo. Si quisieramos simbolizar la misma afirmación de un individuo específico, digamos Pedro, p, simplemente substituiríamos la variable x por p en la fórmula, obteniendo:

$$\neg \exists y (B(y) \wedge (M(p,y) \vee P(p,y))).$$

Sin embargo esto no funciona siempre, supóngase que queremos expresar "y no tiene hijas"; la sustitución de x por la variable y daría:

$$\neg \exists y (B(y) \wedge (M(y,y) \vee P(y,y))) \quad (5)$$

que es una afirmación sin mucho sentido ("Ninguna mujer es madre o padre de si misma"), pero no es exactamente lo que queremos decir. Antes de hacer la sustitución debemos cambiar la variable cuantificada, y , de la fórmula (3) por alguna otra (excepto x, pues ésta distorsionaría nuevamente el significado de la fórmula), digamos w; queda:

$$\phi(x): \neg \exists w (B(w) \wedge (M(x,w) \vee P(x.w))) \quad (6)$$

Esta fórmula significa exactamente lo mismo que (3): "x no tiene hijas"; las variables cuantificadas, y en (3), o w en (6), son "mudas" en el mismo sentido que la x de una integral $\int f(x) dx$. La afirmación "y no tiene hijas" queda simbolizada entonces:

$$\phi(y): \neg \exists w (B(w) \wedge (M(y,w) \vee P(y.w))).$$

Si una fórmula tiene todas sus variables cuantificadas evidentemente no define un predicado o relación, no tiene sujeto o sujetos, éste es el caso de (5) arriba, pero sin embargo hace una aserción bien definida sobre el universo con respecto a los predicados y relaciones primitivos o sobre los predicados definidos. Por ejemplo:

$$\forall x \forall y \forall z (M(y,x) \wedge M(z,x) \supset z=y),$$

que es evidentemente verdadera para los conceptos simbolizados, o la siguiente simbolización no totalmente ortodoxa del adagio "A quién Dios no le dá hijos, el Diablo le dá sobrinos":

$$\forall x (\neg \exists y ("y \text{ es hijo de } x") \supset \exists z \exists w ("x \text{ es hijo de } w" \wedge "w \text{ es hermano de } x"))$$

la cual es evidentemente falsa (lo cual no le quita su valor como descripción de una situación social tal vez común), y cuya simbolización completa dejamos terminar al lector.

Ejemplo 2

Universo: \mathbb{N} , el conjunto de los números naturales

Predicados primitivos: $x < y$ $R(x,y)$

Funciones primitivas: $x+y$ $f(x,y)$
 $x \cdot y$ $g(x,y)$

Individuos primitivos: 0 a
1 b

Algunos predicados definidos:

x es positivo: R(a,x)

x es par: $\exists y(f(y,y)=x) \vee \exists y(g(f(b,b),y)=x)$

x divide a y: $\exists z(g(x,z)=y)$

x es el mínimo número natural: $\forall y(R(x,y) \vee y=x)$

$$x \text{ es primo: } R(b, x) \wedge \forall y \forall z (x = g(y, z) \supset (y = b \vee z = b))$$

$$(y^2+1) \text{ es primo: } \forall w \forall z (f(g(y,y), b) = g(w,z) \supset (w=b \vee z=b)).$$

En ocasiones algunos de los conceptos que se toman como primitivos son definibles de los demás conceptos primitivos, por ejemplo es reducible a + y 0:

$$x \ y: \exists z (\neg(z=a) \wedge f(z,x)=y),$$

lo cual significa que podemos eliminar R de las fórmulas anteriores. Podemos también hacer aserciones sobre los números naturales:

"Hay un máximo natural": $\exists x \forall y (\neg(y=x) \supset R(y,x))$

la cual es obviamente falsa, o

"Todo natural mayor que 1 tiene un divisor primo"

$$\forall x \{ R(b, x) \supset \exists y [\exists z (g(z, y) = x) \wedge R(b, y) \wedge \forall y' \forall z' (y = g(y', z') = (y' = b \vee z' = b))] \}$$

la cual es bien sabido que es verdadera. Lo mismo sucede con el Teorema de Euclides sobre la infinidad de los primos, que podríamos expresar por:

$$\forall x \exists y [R(x, y) \wedge \forall z \forall w (g(z, w) = y \supset (z = b \vee w = b))].$$

Con el simbolismo del ejemplo anterior podemos expresar prácticamente todas las proposiciones de la Teoría Elemental de Números; sin embargo, no tenemos un método mecánico, análogo a las tablas de verdad o algo así, para determinar si una tal proposición es verdadera o no. Por ejemplo, uno de los famosos problemas no resueltos de la Teoría de Números es si existe o no una infinidad de *primos gemelos* (es decir, primos consecutivos como 5 y 7, 11 y 13, 17 y 19, etc.), lo cual significa que no sabemos si la fórmula siguiente es verdadera o no.

$$\forall x \exists y (R(x, y) \wedge \text{Prim}(y) \wedge \text{Prim}(f(y, f(b, b))))$$

donde $\text{Prim}(x)$ es la fórmula que dice " x es primo". Explícitamente

$$\forall x \exists y [R(x, y) \wedge \forall z \forall w (y = g(z, w) \supset (z = b \vee w = z)) \wedge \forall z \forall w (f(y, f(b, b)) = g(z, w) \supset (z = b \vee w = b))].$$

Ejercicios

1. Dado el universo, los conceptos primitivos y el simbolismo del ejemplo 1, simbolice:
 - a. x es nieto de y
 - b. x es tío de y
 - c. x e y son primos hermanos
 - d. x es huérfano

- e. x tiene una sola hermana
 f. x tiene dos hijas
 g. x tiene un hermano que no tiene hijos.
2. Dado el universo, conceptos primitivos y simbolismo del Ejemplo 2, simbolice:
- $2 + 2 = 4$
 - $x \equiv y \pmod{z}$
 - $x = \max(y, z)$
 - $x = \text{mod}(y, z)$
 - "Todo número natural mayor que 3 es la suma de dos primos" (conjetura de Goldbach).
3. En el contexto del ejemplo 2, determine si las siguientes proposiciones son verdaderas o no:
- $\forall x \exists y (f(y, y) = x \vee f(y, b))$
 - $\exists x \exists y (\neg(x = a) \wedge g(g(b, b), g(x, x)) = g(y, y))$
 - $\neg \exists x \neg \exists y \neg(f(x, y) = x)$
 - $\forall x \exists y \exists z \exists w \exists u (x = f(f(g(y, y), g(z, z)), f(g(w, w), g(u, u))))$
 - $[\exists x (g(x, x) = a) \wedge \forall y (\exists x (g(x, x) = y) \supset \exists x (g(x, x) = f(y, b))] \supset \forall y \exists x (g(x, x) = y)$.
4. Tomando como universo los números naturales y utilizando solamente las funciones primitivas :

$$\begin{array}{ll} x + y & f(x, y) \\ x^2 & h(x) \end{array}$$

dé una fórmula cuyo significado sea: " $x = y \circ z$ ".

5. Dados los siguientes universos y funciones primitivas :

Universo: Números reales, \mathbb{R}

funciones primitivas:	$x+y$	$f(x, y)$
	$x \circ y$	$g(x, y)$

defina las relaciones: " $x=0$ ", " $x>0$ ", " $x<y$ ".

6. Repita el ejercicio 5, tomando ahora como universo el conjunto de los números enteros, \mathbb{Z} . Es decir, muestre qué el cero y los positivos son definibles en \mathbb{Z} de la suma y la multiplicación solamente.

Ayuda: Recuerde el teorema de Lagrange de las sumas de cuatro cuadrados.

7. Repita el ejercicio 6, tomando como universo el conjunto de los racionales \mathbb{Q} .

1.4 Sintaxis

Definimos en esta sección de una manera precisa el lenguaje del Cálculo de Predicados que hemos introducido informalmente en las secciones anteriores.

El alfabeto del lenguaje del Cálculo de Predicados consiste en los siguientes símbolos:

Símbolos lógicos

conectivos proposicionales	$\neg \wedge \vee \supset \leftrightarrow$
cuantificador universal	\forall
cuantificador existencial	\exists
igualdad	$=$
variables	$x_1 x_2 x_3 \dots$
símbolos técnicos	$() ,$

Símbolos no lógicos

símbolos de predicado

monádico	$P_1^1 P_2^1 P_3^1 \dots$
binario	$P_1^2 P_2^2 P_3^2 \dots$

ternario	$P_1^3 P_2^3 P_3^3 \dots$
etc.	
símbolos de función	
monádica	$f_1^1 f_2^1 f_3^1 \dots$
binario	$f_1^2 f_2^2 f_3^2 \dots$
etc.	
símbolos de constante	$c_1 c_2 c_3 \dots$

El lenguaje formal contendrá dos tipos de objetos: *términos* (que denotarán individuos) y *fórmulas* (que denotarán afirmaciones). Estos se definen inductivamente:

I. Términos

- (i) Las variables y los símbolos de constante son términos.
- (ii) Si t_1, \dots, t_n son términos y f es un símbolo de función n -aria entonces $f(t_1, \dots, t_n)$ es término.

II. Fórmulas

- (i) Si t_1 y t_2 son términos, $t_1=t_2$ es fórmula *atómica*.
- (ii) Si t_1, \dots, t_n son términos y R es símbolo de relación n -aria entonces $R(t_1, \dots, t_n)$ es fórmula *atómica*.
- (iii) Si ϕ y ψ son fórmulas, entonces $\neg\phi$, $(\phi \wedge \psi)$, $(\phi \vee \psi)$, $(\phi \supset \psi)$ y $(\phi \leftrightarrow \psi)$ son fórmulas.
- (iv) Si ϕ es fórmula y x es variable, entonces $\forall x\phi$ y $\exists x\phi$ son fórmulas.

Como hemos visto en la sección anterior, en la práctica solo precisamos algunos símbolos no lógicos. Un *léxico de primer orden* es un subconjunto del conjunto de los símbolos no lógicos. Dado un léxico L , los *términos* y *fórmulas sobre L* serán aquellos que se pueden construir utilizando solamente los símbolos de relación, función y constante de L . Usaremos como es costumbre las letras x, y, z, w, \dots para denotar variables, c, d, e, \dots para denotar símbolos de constante; P^n, Q^n, R^n, \dots denotarán símbolos de relación n -aria, y escribiremos simplemente P, Q, R, \dots si la *aridad* n es clara del contexto. Igualmente f^n, g^n, h^n, \dots o simplemente f, g, h, \dots , si n es claro del contexto, denotarán símbolos de función n -aria.

Ejemplo

Sea $L = \{P^1, R^2, f^1, g^3, c\}$ entonces los siguientes son términos sobre L :

$$x, c, f(x), g(c, x, c), f(f(f(c))), g(g(x, f(g(x, x, y))), c), c, c$$

en cambio, $g(x, g(g(c, c), c, f(z)), w)$ no es un término (¿por qué?).

Las siguientes son fórmulas atómicas sobre L :

$$c=x, f(f(f(c)))=g(c, f(x), c), P(x), P(c), P(g(f(x), f(x), f(x)))$$

$$R(c, g(f(x), f(f(c)), f(f(f(g(c, f(x), f(f(c))))))), R(x, x),$$

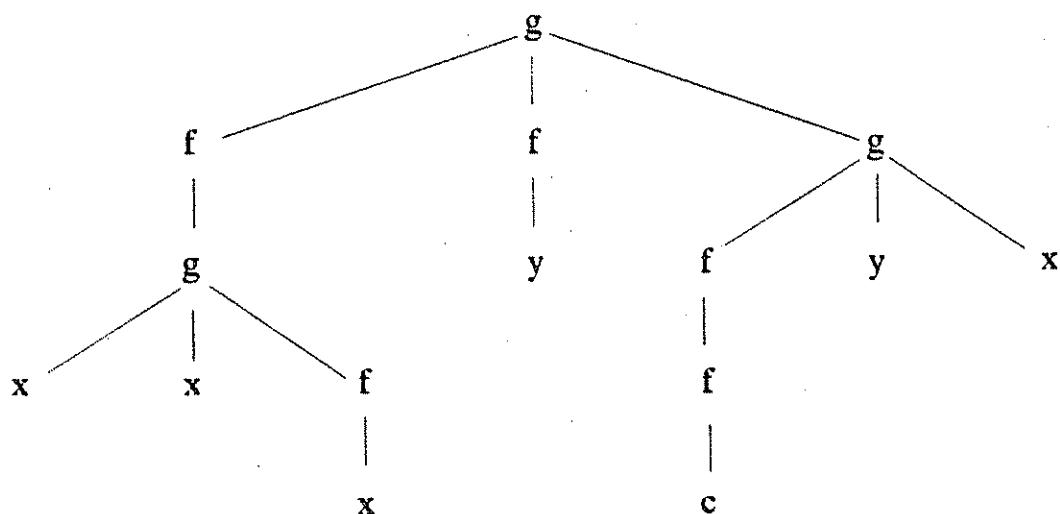
y las siguientes son fórmulas sobre L :

$$\neg \forall x(x=y) \quad \forall x((f(g(x, x, x))=c \wedge \neg P x) \supset (\neg \exists z(R(z, f(x)) \vee x=y))).$$

Observe que un símbolo de función o de relación debe ir seguidos de términos solamente, por lo tanto para el léxico del ejemplo anterior las siguientes expresiones no pertenecen al lenguaje, no son términos ni fórmulas:

$$f(R(x, x)), R(P(y), R(x, y)), R(x, y)=R(y, x).$$

La sintaxis del lenguaje del Cálculo de Predicados es compleja. Ya a nivel de términos se presenta toda la complejidad del Cálculo de Proposiciones; a todo término se puede asociar un árbol, por ejemplo el árbol de $g(f(g(x, x, f(x)), f(y), g(f(f(c)), y, x))$ sería:



Tenemos un *principio de inducción en términos*:

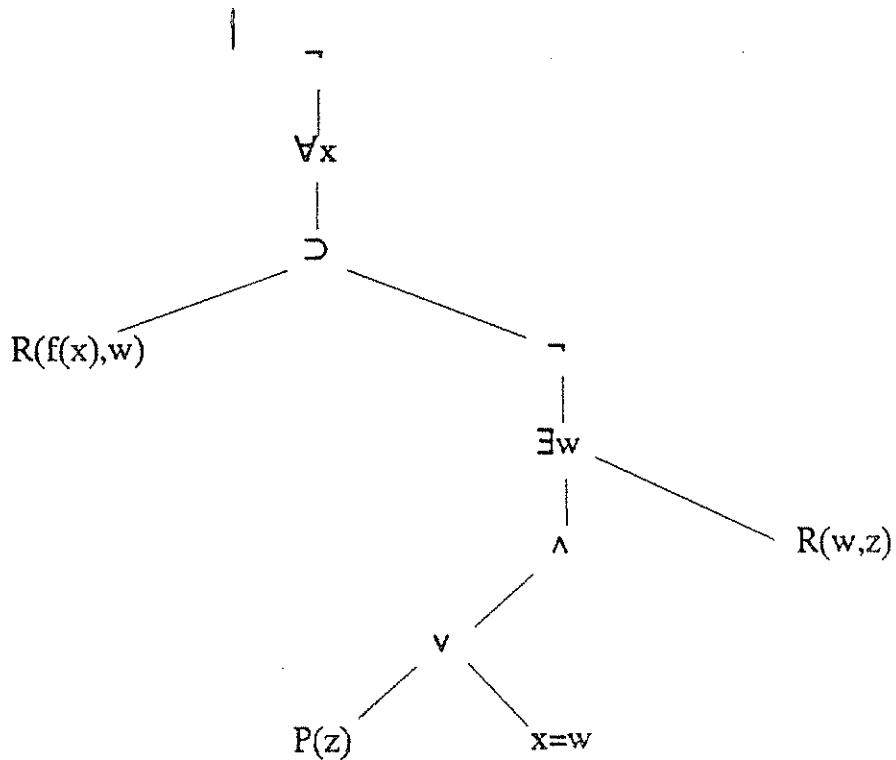
Sea L un léxico; si se tiene una propiedad de cadenas de símbolos tal que:

- (i) las variables tienen la propiedad, las constantes de L tienen la propiedad y
- (ii) para todo símbolo de función $f^n \in L$ y términos t_1, \dots, t_n sobre L que tienen la propiedad, $f(t_1, \dots, t_n)$ también tiene la propiedad; entonces todo término sobre L tiene la propiedad.

Igualmente podemos definir funciones en términos por recurrencia, y es posible decidir por medio de un algoritmo si una cadena de símbolos es un término sobre un léxico dado L o no.

Como el desarrollo es paralelo al que dimos para el Cálculo de Proposiciones no entraremos en detalles y dejaremos algunas de las afirmaciones anteriores como ejercicio.

Para fórmulas tenemos propiedades análogas. Toda fórmula tiene un árbol donde los nodos terminales son fórmulas atómicas. Por ejemplo la fórmula: $\neg \forall x(R(f(x), w) \supset \neg \exists w((P(z) \vee (x=w) \wedge R(w, z)))$ tiene el árbol:



El *principio de inducción en fórmulas* toma la forma:

Si una propiedad vale para las fórmulas atómicas sobre L, y cuando vale para fórmulas ϕ, ψ sobre L, también vale para $\phi \square \psi$ (\square conectivo) y para $\forall x\phi$, $\exists x\phi$ (x variable), entonces vale para toda fórmula sobre L.

Una variable puede aparecer varias veces en una fórmula, cada una de estas apariciones se llama una *ocurrencia* de la variable.

Definición. Una ocurrencia de una variable x en una fórmula ϕ se dice *ligada* si está dentro de una subfórmula que comienza por $\forall x$ o por $\exists x$, de lo contrario se dice *libre*.

Por ejemplo, en la fórmula cuyo árbol se indica arriba todas las ocurrencias de la variable x son ligadas, la primera ocurrencia de w es libre y las demás ligadas, y todas las ocurrencias de z son libres. Por supuesto las variables que siguen inmediatamente después del símbolo de cuantificación están siempre ligadas:

$$\neg \forall x(R(f(x), w) \supset \neg \exists w((P(z) \vee (w=x)) \wedge R(w, z)))$$

↑ libres ↑ _____ ↑

El conjunto de las variables libres de una fórmula puede definirse por recurrencia. Sin embargo, algo que es típico en el Cálculo de Predicados, debemos ocuparnos de los términos antes de llegar a las fórmulas, y definir recursivamente primero las variables de un término:

I. Términos:

$$V(x) := \{x\} \text{ si } x \text{ variable}$$

$$V(c) := \emptyset \text{ si } c \text{ símbolo de constante}$$

$$V(f(t_1, \dots, t_n)) := V(t_1) \cup \dots \cup V(t_n)$$

II. Fórmulas:

$$VL(t_1=t_2) := V(t_1) \cup V(t_2)$$

$$VL(R(t_1, \dots, t_n)) := V(t_1) \cup \dots \cup V(t_n)$$

$$VL(\neg\phi) := VL(\phi)$$

$$VL(\phi \square \psi) := VL(\phi) \cup VL(\psi)$$

$$VL(\forall x\phi) := VL(\phi) \setminus \{x\}$$

$$VL(\exists x\phi) := VL(\phi) \setminus \{x\}$$

Utilizaremos la notación $\phi(x_1, \dots, x_n)$ en ocasiones para indicar que $VL(\phi) \subseteq \{x_1, \dots, x_n\}$. Decimos que ϕ es una *sentencia* si no tiene variables libres, es decir $VL(\phi) = \emptyset$.

Intuitivamente, una fórmula con variables libres expresa un predicado o relación entre esas variables y una sentencia expresa en cambio una proposición categórica. Análogamente un término con variables expresaría una función y un término sin variables expresaría un individuo específico, estos últimos se llamarán *términos cerrados*. La siguiente operación sintáctica es fundamental para el desarrollo del Cálculo de Predicados.

Definición. Si ϕ es una fórmula, x una variable y t un término, $(\frac{x}{t})\phi$ denota la fórmula que resulta de substituir todas las ocurrencias *libres* de x en ϕ por t .

Esta operación puede definirse rigurosamente por recurrencia pasando primero por los términos:

I. Substitución en términos.

$$\left(\frac{x}{t}\right)y := \begin{cases} t & \text{si } y \text{ es } x \\ y & \text{de lo contrario} \end{cases}$$

$$\left(\frac{x}{t}\right)c := c$$

$$\left(\frac{x}{t}\right)f(r_1, \dots, r_n) := f\left(\left(\frac{x}{t}\right)r_1, \dots, \left(\frac{x}{t}\right)r_n\right)$$

II. Substitución en fórmulas.

$$\left(\frac{x}{t}\right)R(r_1, \dots, r_n) := R\left(\left(\frac{x}{t}\right)r_1, \dots, \left(\frac{x}{t}\right)r_n\right)$$

$$\left(\frac{x}{t}\right)(r_1 = r_2) := \left(\frac{x}{t}\right)r_1 = \left(\frac{x}{t}\right)r_2$$

$$\left(\frac{x}{t}\right)\neg\phi := \neg\left(\frac{x}{t}\right)\phi$$

$$\left(\frac{x}{t} \right) (\phi \square \psi) := \left(\frac{x}{t} \right) \phi \square \left(\frac{x}{t} \right) \psi$$

$$\left(\frac{x}{t} \right) \forall y \phi := \begin{cases} \forall y \phi & \text{si } y \text{ es } x \\ \forall y \left(\frac{x}{t} \right) \phi & \text{de lo contrario} \end{cases}$$

(análogamente para $\exists x \phi$).

De una manera similar puede definirse la sustitución simultánea de todas las ocurrencias libres de las variables x_1, \dots, x_n en la fórmula ϕ por los términos t_1, \dots, t_n , respectivamente:

$$\left(\frac{x_1 \dots x_n}{t_1 \dots t_n} \right) \phi$$

En el contexto de la notación $\phi(x_1, \dots, x_n)$ que exhibe las variables libres de ϕ , utilizaremos la notación $\phi(t_1, \dots, t_n)$ para denotar la sustitución de las x_i libres por los t_i en ϕ .

Ejercicios

1. Dado el léxico $L = \{ R^3, f^2, c \}$ determine si las expresiones siguientes son términos sobre L , fórmulas sobre L , o ninguno de los dos.
 - a. $f(c,c)$
 - b. $f(f(x),c)$
 - c. $R(f(c,c),c,f(c,x))$
 - d. $R(c,x)$
 - e. $f(R(x,c,x),y)$
 - f. $R(x,y,f(x,y)) \supset \neg(x=y)$
 - g. $\forall x(f(f(f(x,c),x),c)=z)$
 - h. $\forall x(f(x,x) \supset R(x,x,x))$
 - i. $f(c,x) = f(f(c,x),c)$
 - j. $R(x,f(f(x,f(f(x,y),y)),y),y)$

- k. $R(x, f(x, f(x, f(x, f(x, f(x, y))))))$
- l. $\forall x \forall y \forall z (R(x, y, z) = R(x, y, z))$
2. ¿Cuáles son los términos y fórmulas atómicas sobre el léxico $L = \emptyset$?
3. Dada una fórmula ϕ sean:
- $V(\phi) = \{x \mid x \text{ es una variable que ocurre en } \phi\}$
 - $S(\phi) = \{f \mid f \text{ es un símbolo de función que ocurre en } \phi\}$
 - $R(\phi) = \{R \mid R \text{ es un símbolo de relación que ocurre en } \phi\}$
- De definiciones inductivas de $V(\phi)$, $S(\phi)$ y $R(\phi)$.
4. ¿Cuál es el significado de la función siguiente?
- | | |
|------------------------------|--|
| $h(t_1 = t_2) = 1$ | $h(\phi \square \psi) = h(\phi) + h(\psi) + 1$ |
| $h(R(t_1, \dots, t_n)) = 0$ | $h(\forall x \phi) = h(\phi) + 1$ |
| $h(\neg \phi) = h(\phi) + 1$ | $h(\exists x \phi) = h(\phi) + 1.$ |
5. Señale todas las ocurrencias de variables libre y ligadas en las fórmulas siguientes:
- $$\exists y (\forall x (S(x, y, z) \supset B(y)) \supset (\forall w R(z, w) \wedge R(w, x)))$$
- $$\neg \forall z \exists w \neg \forall x \neg (S(z, w, x) \wedge (\neg \exists y (\neg R(w, y)) \vee B(w)))$$
6. Dé una definición inductiva de $(\frac{x_1 \dots x_n}{t_1 \dots t_n})\phi$
7. Muestre que no necesariamente coinciden las expresiones:
- $$\left(\frac{x_1 x_2}{t_1 t_2} \right) \phi \quad \left(\frac{x_1}{t_1} \right) \left(\frac{x_2}{t_2} \right) \phi \quad \left(\frac{x_2}{t_2} \right) \left(\frac{x_1}{t_1} \right) \phi$$
8. Describa un algoritmo para determinar si una sucesión de símbolos es un término o nó.

9. Como estamos usando notación prefija para los términos, los paréntesis y comas son realmente innecesarios, el término $g(f(g(x, x, f(x)), f(y), g(f(f(c)), y, x))$ podría escribirse:

$gfgxxfxfygffcyx.$

- Dé un algoritmo para construir el árbol de un término expresado en esta notación.
- Dé un algoritmo para decidir si una cadena de símbolos de función, variables y constantes es un término en esta notación.

(Ayuda: se pueden generalizar los métodos utilizados para la notación polaca del Cálculo de Proposiciones).

- 10.* Dé un algoritmo para determinar si una cadena de símbolos es una fórmula (bien formada) de primer orden sobre el léxico $L=\{P_1, R^2, f^1, g^2, c\}$.

11. Dé un algoritmo que recorriendo una fórmula de izquierda a derecha una sola vez marque las ocurrencias libres de la variable x .

Semántica

2.1 Estructuras

Una escogencia de universo, y de predicados, funciones o individuos primitivos en dicho universo se llama una *estructura* (de primer orden) si el universo es un conjunto no vacío. Se acostumbra denotar las estructuras como tuplas

$$\mathfrak{N} = \langle A, R_1^{\mathfrak{N}}, \dots, R_k^{\mathfrak{N}}, f_1^{\mathfrak{N}}, \dots, f_r^{\mathfrak{N}}, c_1^{\mathfrak{N}}, \dots, c_s^{\mathfrak{N}} \rangle$$

donde A es el universo, $R_1^{\mathfrak{N}}, \dots, R_k^{\mathfrak{N}}$, las relaciones primitivas, $f_1^{\mathfrak{N}}, \dots, f_r^{\mathfrak{N}}$, las funciones primitivas y $c_1^{\mathfrak{N}}, \dots, c_s^{\mathfrak{N}}$ los individuos distinguidos primitivos. Por ejemplo,

$$\mathfrak{N} = \langle Z, <, +, :, 0, 1 \rangle \quad (1)$$

Dado un léxico L, se dice que una estructura es *de tipo L* si sus conceptos primitivos se pueden simbolizar utilizando (todos) los símbolos de L. Por ejemplo la estructura anterior es de tipo $L = \{ R^2, f^2, g^2, c, d \}$. Otras estructuras del mismo tipo serían:

$$\mathfrak{S} = \langle \mathbb{R}, -, :, 0, \sqrt{2} \rangle \quad (2)$$

$$C = \langle P(\{1,2,3\}), \subseteq, \cup, \cap, \emptyset, \{1,2,3\} \rangle \quad (3)$$

En general, una sentencia ϕ sobre un léxico dado L no es verdadera ni falsa (es una mera sucesión de símbolos), pero si se fija una estructura \mathfrak{N} de tipo L y se interpreta el rango de los cuantificadores de ϕ lo mismo que de sus símbolos no lógicos por el universo, relaciones, funciones y constantes de \mathfrak{N} , respectivamente, la sentencia adquiere un significado preciso, y por lo tanto resulta verdadera o falsa. En caso de que sea verdadera escribimos:

$$\mathfrak{N} \models \phi$$

y decimos que \mathfrak{N} es *modelo de* ϕ (\mathfrak{N} satisface ϕ , ϕ es verdadera en \mathfrak{N}). En caso de que resulte falsa escribimos: $\mathfrak{N} \not\models \phi$.

Ejemplo 1

La siguiente sentencia sobre $L = \{ R^2, f^2, g^2, c, d \}$

$$\phi : \forall x \forall y [R(x,y) \supset \exists z (\neg(z=c) \wedge f(x,g(x,x))=y)]$$

es falsa en las estructura (1) arriba, pero es verdadera en (2) y (3). En (1) la sentencia dice que para cualquier par de enteros x, y se tiene:

$$x < y \supset \exists z \in \mathbb{Z} (z \neq 0 \wedge x + z^2 = y)$$

lo cual es falso para $x=1, y=3$, pues $1 + z^2 = 3$ con $z \in \mathbb{Z}$ es imposible. En (2) la sentencia dice que para cualquier par de reales x, y se tiene:

$$x > y \supset \exists z \in \mathbb{R} (z \neq 0 \wedge x - z^2 = y)$$

lo cual es cierto pues si $x > y$ entonces $x - y > 0$, y todo positivo en \mathbb{R} tiene raíz cuadrada (positiva), es decir $x - y = z^2, z \in \mathbb{R}^+$, y así $x - z^2 = y$. Finalmente en (3) la sentencia significa: que para cualquier par de subconjuntos S, T de $\{1,2,3\}$

$$S \subsetneq T \supset \exists Z \subseteq \{1,2,3\} (Z \neq \emptyset \wedge S \cup (Z \cap Z) = T)$$

Lo cual es verdaderamente cierto tomando $Z = T \setminus S$. En suma tenemos:

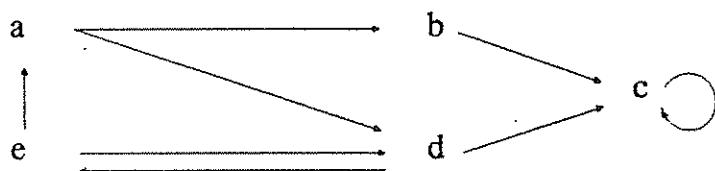
$$\mathfrak{N} \not\models \phi, \mathfrak{S} \models \phi, \mathfrak{C} \models \phi.$$

Ejemplo 2

Si $L = \{R^2\}$, una estructura de tipo L es un par de la forma $\langle A, R \rangle$ donde $A \neq \emptyset$ y $R \subseteq A \times A$. Si el universo A es finito podemos representar la estructura por un grafo dirigido. Por ejemplo

$$\mathfrak{N} = \langle \{a, b, c, d, e\}, \{(a, b), (a, d), (b, c), (d, c), (d, e), (e, a), (e, d), (c, c)\} \rangle$$

puede representarse por



y del grafo es evidente que $\mathfrak{N} \models \forall x \exists y R(x, y)$, pues esta última sentencia dice que de todo nodo sale una flecha. Igualmente:

$$\begin{aligned} \mathfrak{N} \models \exists x [R(x, x) \wedge \forall y (R(y, y) \supset y = x)] & \text{(abreviando } \mathfrak{N} \models \exists !x R(x, x)) \\ \mathfrak{N} \not\models \forall x \exists y \exists z [\neg(y = z) \wedge R(x, y) \wedge R(x, z)] \end{aligned}$$

o lo que es lo mismo:

$$\mathfrak{N} \models \neg \forall x \exists y \exists z [\neg(y = z) \wedge R(x, y) \wedge R(x, z)]$$

El cardinal de una estructura es por definición el cardinal de su universo. Existen sentencias que sólo tienen modelos infinitos.

Ejemplo 3

La sentencia

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

tiene muchos modelos, por ejemplo $\langle \mathbb{N}, < \rangle$ o $\langle \mathbb{R}, > \rangle$, pero todos deben ser infinitos, pues si $\langle A, R \rangle$ es un tal modelo, R debe ser una relación transitiva e irreflexiva en A. Sea $a_0 \in A$, por la última parte de la sentencia

debe existir $a_1 \in A$ tal que a_0Ra_1 , y por la irreflexividad (segunda conjunta de la sentencia) debemos tener $a_0 \neq a_1$. Otra vez por la última conjunta existe $a_2 \in A$ tal que a_1Ra_2 , por transitividad también a_0Ra_2 , y por la irreflexividad esto significa $a_2 \neq a_0$, $a_2 \neq a_1$. Continuando de esta manera obtenemos que A debe contener una cadena infinita:

$$a_0 \rightarrow a_1 \rightarrow a_2 \rightarrow a_3 \dots$$

en donde todos los a_i son distintos entre sí.

Dada una estructura \mathfrak{N} de tipo L , cada fórmula sobre L $\phi(x_1, \dots, x_n)$ con variables libres x_1, \dots, x_n define en \mathfrak{N} una relación n -aria. dados elementos a_1, \dots, a_n del universo de \mathfrak{N} escribimos:

$$\mathfrak{N} \models \phi[a_1, \dots, a_n]$$

y decimos que \mathfrak{N} *satisface* ϕ en (a_1, \dots, a_n) , si la n -tuple (a_1, \dots, a_n) cumple la relación definida por ϕ en \mathfrak{N} .

Ejemplo 4

Si $\phi(x)$ es la fórmula $\forall y(y=x \vee R(x,y))$, tenemos

$$\langle \mathbb{N}, < \rangle \models \phi[0], \langle \mathbb{N}, < \rangle \not\models \phi[1]$$

Si $\phi(x,y)$ es la fórmula sobre $L = \{R^2, f^2, c\}$:

$$\forall z[(\exists w(g(w,z)=x) \wedge \exists w(g(w,z)=y)) \supset z=c]$$

y $\mathfrak{N} = \langle \mathbb{N}, <, \cdot, 1 \rangle$ entonces tenemos $\mathfrak{N} \models \phi[10, 21]$, $\mathfrak{N} \not\models \phi[12, 21]$, pues $\phi(x,y)$ define en \mathfrak{N} la relación de ser relativamente primos.

La relación de verdad entre estructuras y fórmulas:

$$\mathfrak{N} \models \phi[a_1, \dots, a_n]$$

se puede definir rigurosamente por recursión en fórmulas. Para ello, dc-

bemos comenzar haciendo la siguiente observación. Dado un léxico L y una estructura de tipo L, $\mathfrak{N} = \langle A, \dots \rangle$, cada término t sobre L con sus variables libres entre x_1, \dots, x_n define una función n-aria: $t^{\mathfrak{N}}: A^n \rightarrow A$ de acuerdo a la siguiente:

Definición (inductiva)

- I. $t^{\mathfrak{N}}[a_1, \dots, a_n] = a_i \quad \text{si } t \text{ es } x_i$
 $t^{\mathfrak{N}}[a_1, \dots, a_n] = c^{\mathfrak{N}} \quad \text{si } t \text{ es } c, \text{ símbolo de } L.$
- II. Si t_1, \dots, t_k son términos con sus variables entre x_1, \dots, x_n para los cuales $t_i^{\mathfrak{N}}$ esta definida, y f es símbolo de función n-aria de L:
 $f(t_1, \dots, t_k)^{\mathfrak{N}}[a_1, \dots, a_n] = f^{\mathfrak{N}}(t_1^{\mathfrak{N}}[a_1, \dots, a_n], \dots, t_k^{\mathfrak{N}}[a_1, \dots, a_n]).$

Evidentemente la función $t^{\mathfrak{N}}$ sólo depende de las variables que realmente aparecen en t, pero la prueba rigurosa de este hecho exige inducción (ejercicio). En especial, si t es un término cerrado (es decir sin variables) su valor no depende de a_1, \dots, a_n y lo denotamos sencillamente por $t^{\mathfrak{N}}$.

Procedemos ahora a definir la verdad. Por simplicidad, utilizaremos la notación \bar{a} para denotar la tupla a_1, \dots, a_n de elementos de A, con n un natural arbitrario.

Definición

- I. Si ϕ es atómica debe tener la forma $t_1=t_2$ o $R(t_1, \dots, t_k)$ donde los t_i son términos y R símbolo de predicado k-ario:
 - a) $\mathfrak{N} \models t_1=t_2[\bar{a}] \Leftrightarrow t_1^{\mathfrak{N}}[\bar{a}]=t_2^{\mathfrak{N}}[\bar{a}]$.
 - b) $\mathfrak{N} \models R(t_1, \dots, t_k)[\bar{a}] \Leftrightarrow (t_1^{\mathfrak{N}}[\bar{a}], \dots, t_k^{\mathfrak{N}}[\bar{a}]) \in R^{\mathfrak{N}}$.
- II. Suponga que la relación ha sido definida para toda fórmula con m o menos ocurrencias de los símbolos $\neg, \wedge, \vee, \supset, \exists$. Si ϕ es una fórmula con $m+1$ ocurrencias de tales símbolos debe tener una de las formas $\neg\psi, \psi \Box \psi', \forall\psi, \exists x\psi$, en cada caso definimos :
 - a) $\mathfrak{N} \models (\neg\psi)[\bar{a}] \Leftrightarrow \mathfrak{N} \not\models \psi[\bar{a}]$
 - b) $\mathfrak{N} \models (\psi \wedge \psi')[\bar{a}] \Leftrightarrow \mathfrak{N} \models \psi[\bar{a}] \text{ y } \mathfrak{N} \models \psi'[\bar{a}]$

- c) $\mathfrak{N} \models (\psi \vee \psi')[\bar{a}] \Leftrightarrow \mathfrak{N} \models \psi[\bar{a}] \text{ ó } \mathfrak{N} \models \psi'[\bar{a}]$
d) $\mathfrak{N} \models (\psi \supset \psi')[\bar{a}] \Leftrightarrow \mathfrak{N} \models \psi[\bar{a}] \text{ implica } \mathfrak{N} \models \psi'[\bar{a}]$
e) Suponiendo que las variables libres de ψ están entre x, x_1, \dots, x_n :

$$\mathfrak{N} \models (\forall x \psi[\bar{a}] \Leftrightarrow \mathfrak{N} \models \psi[b, \bar{a}], \text{ para todo } b \in A)$$

$$f) \mathfrak{N} \models (\exists x \psi[\bar{a}] \Leftrightarrow \mathfrak{N} \models \psi[b, \bar{a}], \text{ para alguna } b \in A)$$

Por supuesto, la anterior no es sino una formulación precisa del significado ordinario de los símbolos lógicos, pero su carácter inductivo será muy útil para la demostración de resultados no triviales. el que valga o no $\mathfrak{N} \models \phi[a_1, \dots, a_n]$ depende solamente de los valores a_{i_1}, \dots, a_{i_k} para los cuales las variables x_{i_1}, \dots, x_{i_k} aparezcan realmente libres en ϕ ; en particular, si ϕ es una sentencia tenemos:

$$\mathfrak{N} \models \phi \Leftrightarrow \mathfrak{N} \models \phi[a_1, \dots, a_n], \text{ para todo } a_1, \dots, a_n \in A$$

Podríamos haber dado la definición inductiva para los símbolos \neg, \wedge, \forall solamente, pues si se consideran los demás como abreviaciones:

$\psi \vee \psi'$	abrevia $\neg(\neg\psi \wedge \neg\psi')$
$\psi \supset \psi'$	abrevia $\neg(\psi \wedge \neg\psi')$
$\exists x \psi$	abrevia $\neg \forall x \neg \psi$,

las partes c, d y f de la definición serían demostrables, así por ejemplo:

$\mathfrak{N} \models (\exists x \psi)[\bar{a}]$	$\Leftrightarrow \delta \models (\neg \forall x \neg \psi)[\bar{a}]$	Abreviación
	$\Leftrightarrow \delta \models (\forall x \neg \psi)[\bar{a}]$	Definición (a)
	$\Leftrightarrow \delta \models (\neg gy)[b, \bar{a}]$	para algún $b \in A$, Definición (e)
	$\Leftrightarrow \delta \models \psi[b, \bar{a}]$	para algún $b \in A$, Definición (a)

Ejercicios

1. Halle una sentencia ϕ que sea verdadera en una de las estructuras y falsa en la otra.
- | | |
|---|--|
| a. $\mathfrak{N} = \langle \mathbb{Q}, +, \cdot, 1 \rangle$ | $\mathfrak{S} = \langle \mathbb{R}, +, \cdot, 1 \rangle$ |
| b. $\mathfrak{N} = \langle \mathbb{N}, + \rangle$ | $\mathfrak{S} = \langle \mathbb{Z}, + \rangle$ |
| c. $\mathfrak{N} = \langle \mathbb{Q}^+, +, \cdot \rangle$ | $\mathfrak{S} = \langle \mathbb{R}^+, \cdot \rangle$ |
| *d. $\mathfrak{N} = \langle \mathbb{N}, +, 0 \rangle$ | $\mathfrak{S} = \langle \mathbb{N}, \cdot, 1 \rangle$ |
| e. $\mathfrak{N} = \langle \mathbb{N}, < \rangle$ | $\mathfrak{S} = \langle \mathbb{N}, \text{divide} \rangle$ |
| f. $\mathfrak{N} = \langle \mathbb{N}, > \rangle$ | $\mathfrak{S} = \langle \mathbb{N}, > \rangle$ |
| g. $\mathfrak{N} = \langle \mathbb{N}, < \rangle$ | $\mathfrak{S} = \langle \mathbb{N}, \{(n, n+1) n \in \mathbb{N}\} \rangle$ |
| h. $\mathfrak{N} = \langle \mathbb{Z}, \leq \rangle$ | $\mathfrak{S} = \langle \mathbb{Q}, \leq \rangle$ |
| i. $\mathfrak{N} = \langle [0,1] \cup [2,3], < \rangle$ | $\mathfrak{S} = \langle [0,1] \cup [2,3], < \rangle$ |
| j. $\mathfrak{N} = \langle \mathbb{Z}_2, + \bmod 2 \rangle$ | $\mathfrak{S} = \langle \mathbb{Z}_3, + \bmod 3 \rangle$ |
| k. $\mathfrak{N} = \langle \mathbb{R}, + \rangle$ | $\mathfrak{S} = \langle \mathbb{R} - \{0\}, \cdot \rangle$ |
- 2.** ¿Puede dar una sentencia verdadera en $\langle \mathbb{Q}, < \rangle$ y falsa en $\langle \mathbb{R}, < \rangle$ (o viceversa)?
3. Dé una estructura \mathfrak{N} que sea modelo y una estructura \mathfrak{S} que no sea modelo de la sentencia:
- $$\forall x (R(x, 0) \supset \exists z (\neg(z = 0) \wedge f(x, x) = x))$$
4. Sea $\mathfrak{N} = \langle \{1, 2, 3, 4\}, \{(1, 2), (2, 3), (3, 2), (4, 4)\} \rangle$
 Determine si \mathfrak{N} es o no modelo de las sentencias.
- $\forall x \exists y R(x, y)$
 - $\forall x \forall y \forall z (R(x, y) \wedge R(x, z) \supset y = z)$
 - $\forall x \forall y \forall z (R(x, y) \wedge R(y, z) \supset R(x, z))$
 - $\forall x R(x, x) \supset \exists y \forall z R(y, z)$
 - $\exists x \forall y \forall z ((y \neq x \wedge R(y, z)) \supset R(z, y))$
5. Halle un modelo finito para la conjunción de las siguientes sentencias

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

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

$$\forall x \forall y R(x,y) \supset \neg R(y,x))$$

- 6.* Sin usar el símbolo "=" y cuando solamente los símbolos de predicado P^1 , Q^1 (una variable), dé una sentencia cuyos modelos sean todos de cardinalidad mayor o igual que 4.
7. Dé una sentencia que sólo tenga modelos infinitos, sobre el léxico $\{f^1\}$.
8. Dé una sentencia que sea verdadera en todas las estructuras finitas pero sea falsa en alguna estructura infinita.
9. Dado $L = \{R^2, f^1, g^2, h^1, c\}$ determine si las sentencias siguientes son verdaderas o no en $\mathfrak{N} = \mathbb{R}, ., ||, -, \cos, \circ$
- $\forall x \forall y (h(g(x,y)) = g(h(x),h(y)))$
 - $\forall x \forall y \{R(c,y) \supset \exists z [R(c,z) \wedge \forall w (R(f(f(x,w)),z) \supset R(f(g(h(x),h(w))),x))]$
 - $\forall x \{R(c,x) \supset \exists y [R(c,y) \wedge \forall z \forall w (R(f(g(z,w)),y) \supset R(f(h(z),h(w)),x))]\}$
10. Dé una estructura que sea modelo simultáneamente de todas las sentencias siguientes:
- $\forall x \forall y \forall z (f(x,y) = f(x,z) \supset y = z)$
 - $\neg \exists x (f(x,b) = a)$
 - $\forall x (f(x,a) = x)$
 - $\forall x \forall y \forall z (f(x,f(y,z)) = f(f(x,y),z))$
 - $\forall x (g(x,b) = x)$
 - $\forall x \forall y \forall z (g(x,f(y,z)) = f(g(x,y),g(x,z)))$
 - $\forall x [\phi(a) \wedge \forall x (.g(x) \supset \phi(f(x,b))] \supset \forall x \phi(x),$
donde $\phi(x)$ es cualquier fórmula con una variable libre sobre el léxico $L = \{f^2, g^2, a, b\}$.
- 11.* Demuestre que si \mathfrak{N} es una estructura finita de tipo L , hay un algoritmo para determinar si una sentencia cualquiera ϕ sobre L es verdadera o no en \mathfrak{N} .

12. a. Sea t un término cuyas variables estan entre x, x_1, \dots, x_n . Muestre que si la variable x no ocurre en t :

$$t^{\mathfrak{N}}[a, a_1, \dots, a_n] = t^{\mathfrak{N}}[b, a_1, \dots, a_n]$$

para todo $a, b \in A$.

- b. Sea $\phi(x, x_1, \dots, x_n)$ una fórmula. Muestre que si x no ocurre libre en ϕ entonces

$$\mathfrak{N} \models \phi[a, a_1, \dots, a_n] \Leftrightarrow \mathfrak{N} \models \phi[b, a_1, \dots, a_n]$$

para todo $a, b \in A$.

2.2 Validez

Una sentencia es *lógicamente válida* (o *universalmente válida*) si resulta verdadera cualquiera que sea la interpretación que se dé al rango de sus cuantificadores y a los símbolos no lógicos. Más precisamente:

Definición. Una sentencia ϕ sobre L es *lógicamente válida* si y solamente si para toda estructura \mathfrak{N} de tipo L , $\mathfrak{N} \models \phi$. Este hecho se denota por:

$$\models \phi$$

En general, una fórmula $\phi(x_1, \dots, x_n)$ es lógicamente válida si

$$\models \forall x_1, \dots, \forall x_n \phi(x_1, \dots, x_n).$$

Es evidente que las siguientes sentencias y fórmulas son lógicamente válidas:

$$\forall x(P(x) \vee \neg P(x))$$

$$\forall y(R(x,y) \supset R(x,y))$$

$$\forall x(x=x)$$

$$(x = y \wedge y = z) \supset (x = z)$$

$$\forall y (\forall x P(x) \supset P(y))$$

$$R(y,y) \supset \exists z R(z,y)$$

$$\neg \forall x P(x) \supset \exists x \neg P(x)$$

Sin embargo la validez de una sentencia no siempre es inmediata.

Ejemplo 1

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

$$\not\models \forall y \exists x R(x,y) \supset \exists x \forall y R(x,y)$$

Para ver la primera afirmación observe que dada cualquier estructura $\mathfrak{N} = \langle A, R^{\mathfrak{N}} \rangle$ hay dos casos:

caso 1: $\mathfrak{N} \not\models \exists x \forall y R(x,y)$ entonces automáticamente

$$\mathfrak{N} \models \exists x \forall y R(x,y) \supset \forall y \exists x R(x,y),$$

caso 2: $\mathfrak{N} \models \exists x \forall y R(x,y)$, entonces llamando a_0 al elemento de A para el cual $\mathfrak{N} \models \forall y R(a_0,y)$, tenemos para cualquier $b \in A$: $\mathfrak{N} \models R(a_0,b)$, y por lo tanto $\mathfrak{N} \models \exists x R(x,b)$; como b era arbitrario $\mathfrak{N} \models \forall y \exists x R(x,y)$. En suma:

$$\mathfrak{N} \models \exists x \forall y R(x,y) \supset \forall y \exists x R(x,y)$$

Para mostrar que la fórmula conversa no es lógicamente válida es suficiente observar que:

$$\langle \mathbb{Z}, \leq \rangle \not\models \forall y \exists x R(x,y) \supset \exists x \forall y R(x,y) \text{ pues}$$

$$\langle \mathbb{Z}, \leq \rangle \models \forall y \exists x R(x,y) \text{ pero } \langle \mathbb{Z}, \leq \rangle \not\models \exists x \forall y R(x,y)$$

Ejemplo 2

$$\models \exists x \exists y \exists z \forall w (w=x \vee w=y \vee w=z) \wedge \forall x \exists y (f(y)=x) \supset \forall x \forall y [f(x)=f(y) \supset x=y]$$

Esto resulta de que toda función sobreyectiva de un conjunto finito en sí mismo debe ser inyectiva. Dejamos al lector los detalles.

La noción de validez lógica extiende la noción de validez en el Cálculo de Proposiciones, es decir las tautologías deben ser lógicamente válidas, aunque esto debe ser intuitivamente obvio, damos una demostración rigurosa.

Lema. Si ϕ tiene la forma de una tautología entonces $\models \phi$.

Demostración. Sean x_1, \dots, x_n las variables libres de ϕ . Por hipótesis existen subfórmulas ψ_1, \dots, ψ_k de ϕ tales que si se consideran como letras proposicionales entonces $\phi \in \mathfrak{I}(\psi_1, \dots, \psi_k)$ es decir ϕ es una combinación de ellas por medio de los conectivos proposicionales, y esta combinación es una tautología. Las variables libres de ψ_1, \dots, ψ_k deben estar también entre x_1, \dots, x_n . Sea $\mathfrak{M} = A, \dots$ una estructura cualquiera y $a_1, \dots, a_n \in A$; definimos una valuación

$$v: \{\psi_1, \dots, \psi_k\} \rightarrow \{V, F\}$$

por

$$v[\psi_i] = \begin{cases} V & \text{si } \mathfrak{M} \models \psi_i[a_1, \dots, a_n] \\ F & \text{de lo contrario.} \end{cases}$$

Se puede mostrar, por inducción en fórmulas que para toda $\theta \in \mathfrak{I}(\psi_1, \dots, \psi_k)$ se tiene:

$$\bar{v}[\theta] = V \Leftrightarrow \mathfrak{M} \models \theta[a_1, \dots, a_n]$$

Si θ es alguna ψ_i resulta por definición.

Si θ es $\neg\theta'$ tenemos:

$$\begin{aligned} \bar{v}[\neg\theta'] = V &\Leftrightarrow \bar{v}[\theta'] = F \\ &\Leftrightarrow \mathfrak{M} \not\models \theta'[a_1, \dots, a_n] \text{ (hipótesis de inducción)} \\ &\Leftrightarrow \mathfrak{M} \models \neg\theta'[a_1, \dots, a_n] \end{aligned}$$

Si θ es $(\theta' \wedge \theta'')$, tenemos:

$$\begin{aligned} \bar{v}[\theta' \wedge \theta''] = V &\Leftrightarrow \bar{v}[\theta'] = \bar{v}[\theta''] = V \\ &\Leftrightarrow \mathfrak{M} \models \theta'[a_1, \dots, a_n] \text{ y } \mathfrak{M} \models \theta''[a_1, \dots, a_n] \\ &\quad \text{(hipótesis de inducción)} \\ &\Leftrightarrow \mathfrak{M} \models (\theta' \wedge \theta'')[a_1, \dots, a_n] \end{aligned}$$

Como ϕ tiene la forma de una tautología con respecto a las ψ_i , debemos tener $\bar{v}[\phi] = V$ y por tanto $\mathfrak{N} \models \phi[a_1, \dots, a_n]$. Con esto termina la demostración. ■

Algunos esquemas aparentemente válidos deben tomarse con cierto cuidado. Considérense por ejemplo el siguiente esquema llamado de *especificación universal*:

$$\forall x \phi(x) \supset \phi(y) \quad (1)$$

donde $\phi(x)$ es una fórmula cualquiera con la variable libre x . En forma tan general no es lógicamente válido, para verlo supngá que $\phi(x)$ es la fórmula $\exists y R(x,y)$, entonces (1) sería:

$$\forall x \exists y R(x,y) \supset \exists y R(y,y)$$

fórmula que es falsa en $\langle Z, \langle \rangle \rangle$. Lo que ha sucedido es que al substituir x por y en $\phi(x)$, la variable substituida, y , ha quedado atrapada por un cuantificador, distorsionándose el significado de la fórmula. El ejemplo anterior justifica la siguiente definición:

Definición. Sea $\phi(x, x_1, \dots, x_n)$ una fórmula, la variable y es libre para x en ϕ si ninguna ocurrencia libre de x en ϕ está dentro de una subfórmula de ϕ que comience con $\forall y$ o $\exists y$. En general, un término t es libre para x en ϕ si todas las variables de t son libres para x en ϕ .

El efecto de la anterior definición es que si t es libre para x en ϕ , entonces las variables de las ocurrencias de t en $(\frac{x}{t})\phi$ que resultan de la sustitución están libres en $(\frac{x}{t})\phi$.

Es fácil ver que siempre se tiene:

x es libre para x en ϕ ,

si t es un término cerrado, t es libre para x en ϕ .

Lema. Si t es libre para x en ϕ entonces $\models \forall x \phi \supset (\frac{x}{t})\phi$.

Demostración. Aunque este resultado es intuitivamente verdadero, su prueba rigurosa es técnicamente muy complicada. Dejamos parte como ejercicio. Suponga que las variables libres de ϕ y las variables de t están contenidas en x, x_1, \dots, x_n . Si $\mathfrak{N} \models (\forall x \phi)[a_1, \dots, a_n]$ entonces $\mathfrak{N} \models \phi[a, a_1, \dots, a_n]$, para cualquier $a \in A$, en particular: $\mathfrak{N} \models \phi[t^{\mathfrak{N}}[a, a_1, \dots, a_n], a_1, \dots, a_n]$. Por el ejercicio 6,

$$\mathfrak{N} \models \left(\frac{x}{t} \right) \phi[a, a_1, \dots, a_n]$$

Lo anterior muestra que para cualquier estructura \mathfrak{N} y $a, a_1, \dots, a_n \in A$,

$$\mathfrak{N} \models (\forall x \phi \supset \left(\frac{x}{t} \right) \phi)[a, a_1, \dots, a_n],$$

(note que hemos incluido la posibilidad de que la variable x ocurra en t , pero ésto no es necesario). Por lo tanto:

$$\vdash \forall x \forall x_1 \dots \forall x_n (\forall x \phi \supset \left(\frac{x}{t} \right) \phi) \blacksquare$$

De manera análoga podemos mostrar que si t, t_1 y t_2 son libres para x en ϕ entonces:

$$\left(\frac{x}{t} \right) \phi \supset \exists x \phi$$

$$t_1 = t_2 \supset \left(\left(\frac{x}{t_1} \right) \phi \supset \left(\frac{x}{t_2} \right) \phi \right)$$

¿Cómo determinar las fórmulas lógicamente válidas? Este puede considerarse el problema central de la lógica, y en particular de cualquier teoría expresable axiomáticamente, pues si $\Gamma = \{\psi_1, \dots, \psi_n\}$ es un conjunto de axiomas (por ejemplo los axiomas de la Teoría de Grupos), los teoremas de la teoría serán aquellas sentencias ϕ tales que $(\phi_1 \wedge \dots \wedge \phi_n) \supset \phi$ sea válida. A primera vista para determinar la validez de una sentencia sería necesario examinar todas las estructuras. Pero esto es imposible dada la infinitud de posibles estructuras. Más aún, para una estructura fija, de universo infinito, puede ser imposible determinar una sentencia si es verdadera o falsa; éste es el caso de $\langle \mathbb{N}, +, \cdot \rangle$. Paradójicamente el problema de la validez es más sencillo. Veremos que todas las fórmulas válidas pueden deducirse de un

pequeño conjunto de esquemas válidos, incluyendo algunos de los que hemos discutido en esta sección.

Ejercicios

1. ¿Cuáles de las siguientes sentencias son lógicamente válidas?
 - a. $\forall x \exists y (x=y)$
 - b. $\forall x \exists y (\neg(x=y))$
 - c. $\exists x \forall y (x=y)$
 - d. $\forall x. p y \forall z ((\neg(x=y) \wedge y=z) \supset \neg(x=z))$
 - e. $\forall x \forall y ((P(x) \leftrightarrow P(y)) \supset x=y)$
 - f. $\exists x \forall y (x=y) \supset (\forall x P(x) \vee \forall x \neg P(x))$
 - g. $\exists x R(x,x) \supset (\forall y \forall w \neg R(y,w) \supset \exists x R(x,x))$
 - h. $\forall y \forall w (\neg \exists y R(f(x,w),w) \supset (\exists y R(f(y,w),w) \supset \forall x (f(x,y)=w)))$
 - i. $\forall x (P(x) \supset W(x)) \supset (\forall x P(x) \supset \forall x Q(x))$

2. ¿Cuáles de los siguientes casos del esquema de especificación universal son lógicamente válidos ?
 - a. $\forall x \exists y R(x,y) \supset \exists y R(f(x,y),y)$
 - * b. $\forall x \forall y R(x,y) \supset \forall y R(y,y)$
 - c. $\forall x [\exists y \forall x R(x,y) \vee Q(x)] \supset [\exists y \forall x R(x,y) \vee Q(f(y,y))]$
 - d. $\forall x [P(x) \vee Q(x)] \supset [P(x) \vee Q(x)]$
 - e. $\forall x \exists y R(f(y,x),x) \supset \exists y R(f(f(y,f(z,x)),f(z,x)))$

3. En cada caso diga si el término t es libre para x en ϕ , y en caso afirmativo dé $(\frac{x}{t})\phi$.

- | | t | ϕ |
|----|---------------|--|
| a. | y | $\forall x \forall y (f(x,y)=f(y,x)) \supset g(f(x,y))=f(g(x),g(y))$ |
| b. | $f(x,f(y,c))$ | $\exists y (P(y) \vee R(x,y)) \supset \exists z (f(x,z)=c)$ |
| c. | $f(y,z)$ | $f(x,z)=c \supset \forall z (f(y,z)=z)$ |

- 4.* Dado un término fijo t , dé una definición por recurrencia en de la función:

$$F(\phi) = \begin{cases} 1 & \text{si } t \text{ es libre para } x \text{ en } \phi \\ 0 & \text{de lo contrario} \end{cases}$$

- a. Muestre que dadas dos sentencias ϕ y ψ entonces $\models \phi \supset \psi$ si y solamente si todo modelo de ϕ es modelo de ψ .
- * b. Sea $\phi(x, x_1, \dots, x_n)$ una fórmula y t un término cuyas variables están entre x, x_1, \dots, x_n . Demuestre que si t es libre para x en ϕ entonces:

$$\mathfrak{N} \models \left(\frac{x}{t} \right) \phi[a, a_1, \dots, a_n] \Leftrightarrow \mathfrak{N} \models \phi[t^{\mathfrak{N}}[a, a_1, \dots, a_n], a_1, \dots, a_n]$$

Ayuda: use inducción en fórmulas. Primero es necesario probar por inducción en términos cuyas variables están entre x, x_1, \dots, x_n que:

$$\left(\frac{x}{t} \right) t_1^{\mathfrak{N}}[a, a_1, \dots, a_n] = t_1^{\mathfrak{N}}[t^{\mathfrak{N}}[a, a_1, \dots, a_n], a_1, \dots, a_n]$$

2.3 Isomorfismo

Sean \mathfrak{N} y \mathfrak{S} estructuras del mismo tipo L con dominio A y B , respectivamente. Un *isomorfismo* de \mathfrak{N} en \mathfrak{S} es una función $F: A \rightarrow B$ tal que:

- i. F es una biyección
- ii. Para cada símbolo de relación R^n de L y cualesquier $a_1, \dots, a_n \in A$: $(a_1, \dots, a_n) \in R^{\mathfrak{N}}$ si y solo si $(F(a_1), \dots, F(a_n)) \in R^{\mathfrak{S}}$.
- iii. Para cada símbolo de función f^n de L y cualesquier $a_1, \dots, a_n \in A$: $F(f^{\mathfrak{N}}(a_1, \dots, a_n)) = f^{\mathfrak{S}}(F(a_1), \dots, F(a_n))$.
- iv. Para cada símbolo constante de L : $F(c^{\mathfrak{N}}) = c^{\mathfrak{S}}$

Escribimos $\mathfrak{N} \sim \mathfrak{S}$ si existe un isomorfismo de \mathfrak{N} en \mathfrak{S} , y decimos que \mathfrak{N} y \mathfrak{S} son *isomorfas*. Un isomorfismo de una estructura en si misma se llama un *automorfismo*.

Ejemplo 1

Si $\mathfrak{R} = \langle \mathbb{R}, <, +, 0 \rangle$ y $\mathfrak{S} = \langle \mathbb{R}^+, >, \cdot, 1 \rangle$ la función $F: \mathbb{R} \rightarrow \mathbb{R}^+$, $F(a) = e^{-a}$ es un isomorfismo de \mathfrak{R} en \mathfrak{S} . Por tener inverso, $F^{-1}(b) = -\log_e(b)$, F es biyectiva. Además F es estrictamente decreciente, por tanto:

- ii. $(a, b) \in \mathbb{R}^{\mathfrak{R}} \Leftrightarrow a < b \Leftrightarrow e^{-a} > e^{-b} \Leftrightarrow (F(a), F(b)) \in \mathbb{R}^{\mathfrak{S}}$
- iii. $F(f^{\mathfrak{R}}(a, b)) = F(a+b) = e^{-(a+b)} = e^{-a}e^{-b} = F(a) \cdot F(b) = f^{\mathfrak{S}}(F(a), F(b))$
- iv. $F(c^{\mathfrak{R}}) = F(0) = e^0 = 1 = c^{\mathfrak{S}}$

Ejemplo 2

$F(n) = n + 5$ es un automorfismo de $\langle \mathbb{Z}, < \rangle$, pues F es evidentemente una biyección de \mathbb{Z} y además se cumple (ii):

$$n < m \Leftrightarrow n+5 < m+5, \quad n, m \in \mathbb{Z}$$

Sin embargo, el único automorfismo de $\langle \mathbb{Z}, <, 0 \rangle$ es la identidad (ejercicio).

Dos estructuras isomorfas sólo se distinguen en la "substancia" de sus elementos y no en las interrelaciones entre dichos elementos, por lo tanto deben tener las mismas propiedades. Esto se expresa rigurosamente en el siguiente teorema cuya demostración dejamos para el final de la sección.

Teorema del isomorfismo *Sea F un isomorfismo entre estructuras \mathfrak{R} y \mathfrak{S} de tipo L , y sea $\phi(x_1, \dots, x_n)$ una fórmula sobre L , entonces para cualesquier elementos a_1, \dots, a_n del universo de \mathfrak{R} :*

$$\mathfrak{R} \models \phi[a_1, \dots, a_n] \Leftrightarrow \mathfrak{S} \models \phi[f(a_1), \dots, F(a_n)]$$

Colorario Si $\mathfrak{R} \simeq \mathfrak{S}$ entonces para toda sentencia ϕ :

$$\mathfrak{R} \models \phi \Leftrightarrow \mathfrak{S} \models \phi$$

Ejemplo 3

Del ejemplo 1 resulta que $\langle \mathbb{R}, + \rangle \simeq \langle \mathbb{R}^+, \cdot \rangle$ podemos probar sin embargo que $\langle \mathbb{R}, + \rangle \neq \langle \mathbb{R}, \cdot \rangle$; Para ello consideraremos la sentencia ϕ :

$$\exists x \exists y [f(x, x) = x \wedge f(y, y) = y \wedge x \neq y]$$

cuyo significado en la primera estructura es: "la ecuación $x + x = x$ tiene dos soluciones distintas" y en la segunda: "la ecuación $x \cdot x = x$ tiene dos soluciones distintas". Obviamente $\langle \mathbb{R}, + \rangle \not\models \phi$, $\langle \mathbb{R}, \cdot \rangle \models \phi$, y las estructuras no pueden ser isomórficas por el corolario.

Ejemplo 4

La función $F(k) = -k$ es un automorfismo de $\langle \mathbb{Z}, +, 0 \rangle$ pues $F(0) = -0 = 0$ y $F(k + k') = -(k + k') = -k + (-k') = F(k) + F(k')$. Por lo tanto para cualquier fórmula $\phi(x_1, \dots, x_n)$ sobre el léxico $\{+, 0\}$ y enteros k_1, \dots, k_n tenemos:

$$\langle \mathbb{Z}, +, 0 \rangle \models \phi[k_1, \dots, k_n] \Leftrightarrow \langle \mathbb{Z}, +, 0 \rangle \models \phi[-k_1, \dots, -k_n].$$

Para demostrar el Teorema de Isomorfismo necesitamos antes el siguiente resultado.

Lema Si F es un isomorfismo de \mathfrak{R} en \mathfrak{S} entonces para cualquier término t (sobre el léxico L) con variables entre x_1, \dots, x_n y elementos a_1, \dots, a_n del universo de \mathfrak{R} :

$$F(t^{\mathfrak{R}}(a_1, \dots, a_n)) = t^{\mathfrak{S}}(F(a_1), \dots, F(a_n))$$

Demostración. Por inducción en términos

I. t es x_i :

$$F(t^{\mathfrak{R}}(a_1, \dots, a_n)) = F(a_i) = t^{\mathfrak{S}}(F(a_1), \dots, F(a_n)).$$

II. t es c :

$$F(t^{\mathfrak{R}}(a_1, \dots, a_n)) = F(c^{\mathfrak{R}}) = c^{\mathfrak{S}} = t^{\mathfrak{S}}(F(a_1), \dots, F(a_n)).$$

III. Supóngalo cierto para t_1, \dots, t_k cuyas variables están entre x_1, \dots, x_n , sea f símbolo de función k -ádica. Si t es $f(t_1, \dots, t_k)$:

$$\begin{aligned}
 F(t^{\mathfrak{R}}(a_1, \dots, a_n)) &= F[f^{\mathfrak{R}}(t_1^{\mathfrak{R}}(a_1, \dots, a_n), \dots, t_k^{\mathfrak{R}}(a_1, \dots, a_n))] && \text{Definición} \\
 &\quad f^{\mathfrak{S}}(F[t_1^{\mathfrak{R}}(a_1, \dots, a_n)], \dots, F[t_k^{\mathfrak{R}}(a_1, \dots, a_n)]) && \text{F isomorfismo} \\
 &\quad f^{\mathfrak{S}}(t_1^{\mathfrak{S}}(F(a_1), \dots, F(a_n)), \dots, t_k^{\mathfrak{S}}(F(a_1), \dots, F(a_n))) && \text{Hipótesis} \\
 &\quad t^{\mathfrak{S}}(F(a_1), \dots, F(a_n)) && \text{Definición} \blacksquare
 \end{aligned}$$

Demostración del teorema de isomorfismo. Lo haremos por inducción en fórmulas. Para simplificar la notación utilizaremos \bar{a} para referirnos a la tupla (a_1, \dots, a_n) y $F(\bar{a})$ para $(F(a_1), \dots, F(a_n))$.

I. ϕ es atómica, entonces ϕ es " $t_1 = t_2$ " o " $R(t_1, \dots, t_k)$ " donde cada $t_i \in L$ tiene sus variables entre x_1, \dots, x_n

$$\begin{array}{lll}
 \text{a) } \mathfrak{N} \models t_1 = t_2[\bar{a}] & \Leftrightarrow t_1 \mathfrak{R}[\bar{a}] = t_2 \mathfrak{R}[\bar{a}] & \text{Definición} \\
 & \Leftrightarrow F(t_1 \mathfrak{R}[\bar{a}]) = F(t_2 \mathfrak{R}[\bar{a}]) & F \text{ es uno a uno} \\
 & \Leftrightarrow t_1 \mathfrak{S}[F(\bar{a})] = t_2 \mathfrak{S}[F(\bar{a})] & \text{Lema} \\
 & \Leftrightarrow \mathfrak{i} \models t_1 = t_2[F(\bar{a})] & \text{Definición} \\
 \\
 \text{b) } \mathfrak{N} \models R(t_1, \dots, t_k)[\bar{a}] & \Leftrightarrow (t_1 \mathfrak{R}[\bar{a}], \dots, t_k \mathfrak{R}[\bar{a}]) \in R^{\mathfrak{R}} & \text{Definición} \\
 & \Leftrightarrow (F(t_1 \mathfrak{R}[\bar{a}]), \dots, F(t_k \mathfrak{R}[\bar{a}])) \in R^{\mathfrak{S}} & F \text{ es isomorfismo} \\
 & \Leftrightarrow (t_1 \mathfrak{S}[F(\bar{a})], \dots, t_k \mathfrak{S}[F(\bar{a})]) \in R^{\mathfrak{S}} & \text{Lema} \\
 & \Leftrightarrow \mathfrak{i} \models R(t_1, \dots, t_k)[F(\bar{a})] & \text{Definición}
 \end{array}$$

II. Hacemos el paso de inducción para \neg , \wedge y \forall . Suponga la equivalencia cierta para toda fórmula $\phi(x_1, \dots, x_m)$ con a lo sumo m ocurrencias de \neg \wedge \forall . Si en ϕ ocurren $m+1$ tales símbolos tenemos uno de los siguientes casos ϕ es $\neg\phi$, ϕ es $(\psi \wedge \psi')$, o ϕ es $\forall x \psi$:

$$\begin{array}{lll}
 \text{a) } \mathfrak{N} \models (\neg \psi)[\bar{a}] & \Leftrightarrow \mathfrak{N} \not\models \psi[\bar{a}] & \text{Definición} \\
 & \Leftrightarrow \mathfrak{i} \not\models \psi[F(\bar{a})] & \text{Hipótesis de inducción} \\
 & \Leftrightarrow \mathfrak{i} \models (\neg \psi)[F(\bar{a})] & \text{Definición} \\
 \\
 \text{b) } \mathfrak{N} \models (\psi \wedge \psi')[\bar{a}] & \Leftrightarrow \mathfrak{o} \models \psi[\bar{a}] \text{ y } \mathfrak{N} \models \psi'[\bar{a}] & \text{Definición} \\
 & \Leftrightarrow \mathfrak{i} \models \psi[F(\bar{a})] \text{ y } \mathfrak{S} \models \psi'[F(\bar{a})] & \text{Hipótesis de inducción} \\
 & \Leftrightarrow \mathfrak{i} \models (\psi \wedge \psi')[F(\bar{a})] & \text{Definición} \\
 \\
 \text{c) } \mathfrak{N} \models (\forall x \psi)[a_1, \dots, a_n] & \Leftrightarrow \mathfrak{N} \models \psi[a_1, \dots, a_n, a] & \text{Para cualquier } a \in A
 \end{array}$$

$$\begin{aligned}
 &\Leftrightarrow \mathfrak{I} \models \psi[F(a_1), \dots, F(a_n), F(a)] \text{ para cualquier } a \in A \text{ por hipótesis de inducción} \\
 &\Leftrightarrow \mathfrak{i} \models \psi[F(a_1), \dots, F(a_n), b] \text{ para cualquier } b \in L \text{ por ser } F \text{ sobreyectiva} \\
 &\Leftrightarrow \mathfrak{i} \models (\forall x \psi)[F(a_1), \dots, F(a_n)] \blacksquare
 \end{aligned}$$

El converso del Teorema de Isomorfismo no es cierto; hay estructuras que satisfacen las mismas sentencias y no son isomorfas, por ejemplo $\langle \mathbb{Q}, + \rangle$ y $\langle \mathbb{R}, + \rangle$. La demostración de este hecho es sin embargo difícil.

Ejercicios

- Demuestre que \sim es una relación de equivalencia entre estructuras del mismo tipo.
- Demuestre que el único automorfismo de $\langle \mathbb{Z}, +, 0 \rangle$ es la identidad.
- Utilice el automorfismo del ejemplo 4 para demostrar que en \mathbb{Z} la multiplicación no es definible de $+$ y 0 . Es decir no existe una fórmula $\phi(x,y,z)$ sobre el léxico $\{+, 0\}$ que defina la relación: " $x = y * z$ " en $\langle \mathbb{Z}, +, 0 \rangle$. (Ayuda: si existiera ϕ tendríamos

$$\langle \mathbb{Z}, +, 0 \rangle \models \phi[1,1,1] \Leftrightarrow \langle \mathbb{Z}, +, 0 \rangle \models \phi[-1,-1,-1]$$

- Demuestre que $\langle \mathbb{R}, + \rangle$ y $\langle \mathbb{R} - \{0\}, * \rangle$ no son isomorfas; $\langle \mathbb{R}, +, * \rangle$ y $\langle \mathbb{C}, +, * \rangle$ no son isomorfas.
- Sean $\mathfrak{N} = \langle A, P_1, P_2 \rangle$, $\mathfrak{S} = \langle B, Q_1, Q_2 \rangle$ estructuras monádicas. Demuestre que $\mathfrak{N} \sim \mathfrak{S}$ si y sólo si:

$$|P_1 \cap P_2| = |Q_1 \cap Q_2| \quad |P_1^c \cap P_2| = |Q_1^c \cap Q_2|$$

$$|P_1 \cap P_2^c| = |Q_1 \cap Q_2^c| \quad |P_1^c \cap P_2^c| = |Q_1^c \cap Q_2^c|$$

¿Cómo se generaliza esto a estructuras monádicas $\langle A, P_1, \dots, P_m \rangle$, $\langle B, P_1, \dots, P_m \rangle$?

6. Sea $L = \{R^2\}$ y $\phi(x_1, \dots, x_n)$ una fórmula de L .

Demuestre que si $r_1 < \dots < r_n$ y $s_1 < \dots < s_n$ son dos sucesiones crecientes de reales entonces

$$\langle \mathbb{R}, < \rangle \models \phi[r_1, \dots, r_n] \Leftrightarrow \langle \mathbb{R}, < \rangle \models \phi[s_1, \dots, s_n]$$

(Ayuda: construya un automorfismo apropiado).

- 7.* Demuestre que para cada par de números reales distintos r, s , se puede dar una fórmula $\phi(x)$ tal que

$$\langle \mathbb{R}, <, +, \cdot, 0, 1 \rangle \models \phi[r], \quad \langle \mathbb{R}, <, +, \cdot, 0, 1 \rangle \not\models \phi[s]$$

Concluya de lo anterior que el único automorfismo de $\langle \mathbb{R}, +, \cdot, 0, 1 \rangle$ es la identidad.

- 8.* $F: A \rightarrow B$ es un *holomorfismo* de \mathfrak{N} en \mathfrak{I} , si para todo símbolo de relación R^n , función f^n y constante c , y elementos a_1, \dots, a_n de A

- $(a_1, \dots, a_n) \in R^{\mathfrak{N}} \Rightarrow (F(a_1), \dots, F(a_n)) \in R^{\mathfrak{I}}$
- $F(f^{\mathfrak{N}}(a_1, \dots, a_n)) = f^{\mathfrak{I}}(F(a_1), \dots, F(a_n))$
- $F(c^{\mathfrak{N}}) = c^{\mathfrak{I}}$

Demuestre:

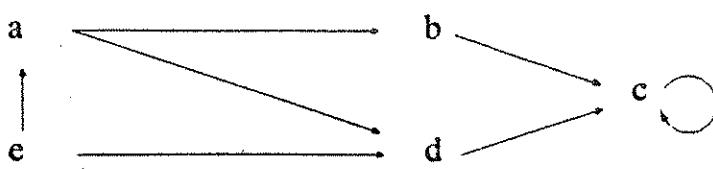
- a. si $\phi(x_1, \dots, x_n)$ es una fórmula de L en donde solo ocurren \wedge, \vee, \exists entonces $\mathfrak{N} \models \phi[a_1, \dots, a_n] \Rightarrow \mathfrak{I} \models \phi[F(a_1), \dots, F(a_n)]$.

- b. Si F es sobreyectiva entonces (a) vale cuando ϕ solo contiene $\wedge, \vee, \exists, \forall$.

- 9.* Demuestre que si \mathfrak{N} es una estructura finita de tipo L (un léxico finito), existe una sentencia ϕ sobre L tal que:

$$\mathfrak{I} \models \phi \Leftrightarrow \mathfrak{I} \simeq \mathfrak{N}$$

Dé tal sentencia para la estructura descrita por el grafo:



Capítulo 3

Deducción Formal

3.1 Axiomatización del Cálculo de Predicados

Para simplificar nuestro sistema suponemos que los únicos símbolos lógicos son \neg , \supset , \forall , $=$. Los demás se consideran abreviaciones, en la forma siguiente:

$\alpha \vee \beta$	abrevia	$\neg \alpha \supset \beta$
$\alpha \wedge \beta$	abrevia	$\neg (\alpha \supset \neg \beta)$
$\alpha \leftrightarrow \beta$	abrevia	$(\alpha \supset \beta) \vee (\beta \supset \alpha)$
$\exists x \phi$	abrevia	$\neg \forall x \neg \phi$

En adelante escribiremos $\phi(x)$ para enfatizar que la variable x puede estar libre en ϕ (y que ese es el caso interesante a considerar), pero puede suceder que x no esté realmente libre en ϕ y que ϕ contenga otras variables libres no exhibidas. En el contexto de $\phi(x)$ la expresión $\phi(t)$ denotará $(x_t)\phi$, es decir la sustitución de *todas* las ocurrencias libres de x en ϕ por t .

Definición 1. Un axioma del Cálculo de Predicados es cualquier fórmula θ que tenga una de las formas siguientes, o una tal fórmula θ que precedida de cuantificadores universales: $\forall v_1 \dots \forall v_n \theta$.

1. $\phi \supset (\psi \supset \phi)$
2. $(\phi \supset (\psi \supset \rho)) \supset ((\phi \supset \psi) \supset (\phi \supset \rho))$

3. $(\neg \phi \supset \neg \psi) \supset (\psi \supset \phi)$
4. $\forall x (\phi \supset \psi) \supset (\forall x \phi \supset \forall x \psi)$
5. $\forall x \phi(x) \supset \phi(t)$ si t libre para x en $\phi(x)$
6. $\phi \supset \forall x \phi$ si x no ocurre libre en ϕ
7. $t=t$
8. $x = t \supset (\phi(x) \supset \phi(t))$ si t libre para x en $\phi(x)$

Definición 2. Sea Γ un conjunto de fórmulas, una Γ -deducción es una sucesión de fórmulas ϕ_1, \dots, ϕ_n tal que cada ϕ_i es un axioma, es una fórmula de Γ , o viene por Modus Ponens (MP) de dos fórmulas anteriores. Si ϕ es la última fórmula de una Γ -deducción, escribimos $\Gamma \vdash \phi$. En lugar de $\emptyset \vdash \phi$, escribimos $\vdash \phi$.

Puesto que los axiomas 1, 2 y 3 junto con MP forman un sistema completo para el Cálculo de Proposiciones basado en \neg y \supset , tenemos que si $\phi_1, \dots, \phi_n \vdash \phi$ en el Cálculo de Predicados. Además, como la prueba del Teorema de la Deducción en Cap 2 § 2.3 sólo usa axiomas 1 y 2 y el hecho de que la única regla es MP, éste sigue valiendo en el nuevo sistema. Lo mismo puede decirse de la prueba por contradicción R.A. Cuando usemos reglas proposicionales lo justificaremos escribiendo "Prop".

Note que del Axioma 5 y MP se sigue la regla derivada:

T0. $\forall x \phi(x) \vdash \phi(t)$, Especificación Universal ($G \forall$), si t libre para x en $\phi(x)$.

En seguida desarrollamos otras reglas igualmente útiles.

T1. $\phi(t) \vdash \exists x \phi(x)$, Generalización Existencial ($G \exists$), si t libre para x en $\phi(x)$.

Demostración

- | | |
|--|---------|
| 1. $\forall x \neg \phi(x) \supset \neg \phi(t)$ | A5 |
| 2. $\phi(t) \supset \neg \forall x \neg \phi(x)$ | Prop, 1 |
| 3. $\phi(t)$ | P |
| 4. $\neg \forall x \neg \phi(x)$ | MP, 2,3 |

Tenemos por ejemplo el caso siempre correcto: $\phi(x) \vdash \exists x \phi(x)$.

La regla de Generalización existencial tiene una forma más compleja de quantificación parcial:

$$\phi(t,t) \vdash \exists x \phi(x,t)$$

si t es libre para x en $\phi(x,x)$ y x no es variable de t.

Para demostrarlo observe que el siguiente es un caso de A 5:

$$\forall x \neg \phi(x,t) \supset \neg \phi(t,t)$$

gracias a que x no ocurre en la segunda t, de los contrario el axioma sería más bien:

$$\forall x \neg \phi(x,t) \supset \phi(t,(\frac{x}{t})t)$$

resto de la prueba es como en T1. Por ejemplo, tenemos

$$x+y = x+y \vdash \exists z (z = x+y)$$

ro el siguiente sería un uso incorrecto de G \exists :

$$x+y = x+y \vdash \exists y (y = x+y)$$

$t_1=t_2 \quad \phi(t_1, t_1) \vdash \phi(t_2, t_1)$, Substitución de la Identidad (SI), si t_1 y t_2 son libres para x en $\phi(x,x)$.

nostración. Escoja una variable y que no ocurra en $\phi(x,x)$ ni en t_1 .

$$\forall y [x = t_2 \supset (\phi(x,y) \supset \phi(t_2,y))] \quad A8$$

$$\forall y [t_1=t_2 \supset (\phi(t_1,y) \supset \phi(t_2,y))] \quad E \forall (\text{pues } y \text{ no ocurre en } t_1)$$

$$t_1=t_2 \supset (\phi(t_1,t_1) \supset \phi(t_1,t_2)) \quad E \forall (\text{pues } t_1 \text{ libre para } x \text{ en } \phi(-, x))$$

$$t_1=t_2 \quad P$$

$$(t_1,t_1) \quad P$$

$$(t_2,t_1) \quad MP \text{ (dos veces) } 3,4,5$$

La regla dice que se puede substituir una o algunas ocurrencias de t_1 por t_2 , no necesariamente todas. Como aplicación tenemos:

$$T3. t_1=t_2 \mid \dots t_1=t_2$$

Demostración

$$1. t_1=t_2$$

P

$$2. t_1=t_1$$

A7



$$3. t_2=t_1$$

SI. 1,2 (primera ocurrencia)

$$T4. t_1=t_3 \quad t_2=t_3 \mid \dots t_1=t_3.$$

Demostración

$$1. t_2=t_3$$

P

$$2. t_1=t_2$$

P



$$3. t_1=t_3$$

SI. 1,2

Note que un caso especial de SI es la sustitución algebraica:

$$t_1=t_2 \mid \dots t(\dots t_1\dots) = t(\dots t_2\dots)$$

que resulta de:

$$1. t_1=t_2$$

P

$$2. t(\dots t_1\dots) = t(\dots t_1\dots)$$

A7



$$3. t(\dots t_1\dots) = t(\dots t_2\dots)$$

SI (segunda ocurrencia de t_1)

Para continuar desarrollando el sistema deductivo con mayor facilidad, demostramos la poderosa regla de *Generalización Universal (G V)*.

Teorema. Si $\Gamma \vdash \phi(x)$ y x no ocurre libre en ninguna fórmula de Γ entonces $\Gamma \vdash \forall x \phi(x)$.

Demostración. Sea ϕ_1, \dots, ϕ_n una Γ -deducción con ϕ_1 idéntica a $\phi(x)$. Demostremos por inducción en $k \leq n$ que $\Gamma \vdash \forall x \phi_k$.

I. Si $k=1$ hay dos casos

a) ϕ_1 es axioma, entonces $\forall x \phi_1$, también lo es y tenemos trivialmente $\Gamma \vdash \forall x \phi_1$.

b) $\phi_1 \in \Gamma$, entonces x no ocurre libre en ϕ_1 . Usando el axioma 6: $\phi_1 \supset \forall x \phi_1$ y por MP tenemos $\Gamma \vdash \forall x \phi_1$.

II. Suponemos que para toda $l < k$, $\Gamma \vdash \forall x \phi_l$. Si ϕ_k es axioma o pertenece a Γ , tenemos $\Gamma \vdash \forall x \phi_k$ como en (1). Si ϕ_k viene por MP de ϕ_i y $\phi_j = \phi_i \supset \phi_k$ con $i, j < k$, entonces $\Gamma \vdash \forall x \phi_i$ y $\Gamma \vdash \forall x (\phi_i \supset \phi_k)$ por hipótesis de inducción. Aplicando A4 tenemos $\Gamma \vdash \forall x (\phi_i \supset \phi_k) \vdash \forall x \phi_i \supset \forall x \phi_k$ y por MP; $\Gamma \vdash \forall x \phi_k$. ■

Corolario. Si $\vdash \phi$ entonces $\vdash \forall x_1, \dots, \forall x_n \phi$

Ejemplos

De T4, tenemos $x = y, y = z \vdash x = z$,

Pro TD: $x = y \vdash y = z \supset x = z$,

Pro G \forall : $x = y \vdash \forall z (y = z \supset x = z)$.

Sería incorrecto cuantificar x ó y y en la conclusión sin embargo, tenemos

Pro TD: $\vdash x = y \supset \forall z (y = z \supset x = z)$,

Pro G \forall : $\vdash \forall x \forall y (x = y \supset \forall z (y = z \supset x = z))$.

T5. $\forall x \phi(x) \vdash \forall y \phi(y)$, si y no ocurre libre en $\phi(x)$, y libre para x en $\phi(y)$

Demostración

1. $\forall x \phi(x)$

P

2. $\phi(y)$

E $\forall, 1.$ (A5 + MP)

3. $\forall y \phi(y)$

G \forall , 2

T6. $\exists x \phi(x) \vdash \neg \exists y \phi(y)$ y libre en $\phi(y)$

Demostración

1. $\forall y \neg \phi(y) \supset \forall x \neg \phi(x)$ T5 + TD
2. $\neg \forall x \neg \phi(x) \supset \neg \forall y \neg \phi(y)$ Prop,2.
3. $\neg \forall x \neg \phi(x)$ P
4. $\neg \forall y \neg \phi(y)$ MP,2,3.

T7. $\forall x (\phi \leftrightarrow \psi) \vdash \forall x \phi \leftrightarrow \forall x \psi$

Demostración

1. $\forall x (\phi \leftrightarrow \psi)$ P
2. $\phi \leftrightarrow \psi$ E \forall ,1.
3. $\phi \supset \psi$ Prop,2.
4. $\forall x (\phi \supset \psi)$ G \forall ,3.
5. $\forall x \phi \supset \forall x \psi$ A4 + MP,4.

Similarmente se demuestra $\forall x (\phi \leftrightarrow \psi) \vdash \forall x \psi \supset \forall x \phi$ y por Prop tenemos $\forall x (\phi \leftrightarrow \psi) \vdash \forall x \phi \leftrightarrow \forall x \psi$.

T8. $\forall x (\phi \wedge \psi) \vdash \forall x \phi \wedge \forall x \psi$

Demostración.

$$\forall x (\phi \wedge \psi) \vdash_{\text{EA}} \phi \wedge \psi \quad \vdash_{\text{Prop}} \left\{ \begin{array}{c} f \quad \overline{\Gamma} \\ \forall x \quad \phi \\ \psi \quad \overline{\Gamma} \\ \forall x \quad \psi \end{array} \right\} \rightarrow \vdash_{\text{Prop}} \forall x \phi \wedge \forall x \psi$$

Note que si se invierte el orden de la deducción, se tiene una deducción correcta del converso.

T9. $\forall x (\psi \supset \phi(x)) \vdash \psi \supset \forall x \phi(x)$, x no ocurre libre en ϕ .

Demostración

1. $\forall x (\psi \supset \phi(x))$ P

2. ψ	P
3. $\psi \supset \phi(x)$	E \forall ,1
4. $\phi(x)$	MP 2,3
5. $\forall x \phi(x)$	G \forall (x no ocurre libre en (1) y (2))

Por TD, concluimos: $\forall x (\psi \supset \phi(x)) \vdash \psi \supset \forall x \phi(x)$. La otra dirección:

1. $\psi \supset \forall x \phi(x)$	P
2. ψ	P
3. $\forall x \phi(x)$	MP,1,2
4. $\phi(x)$	E \forall

Por TD: $\psi \supset \forall x \phi(x) \vdash \psi \supset \phi(x)$. Por G \forall : $\psi \supset \forall x \phi(x) \vdash \forall x (\psi \supset \phi(x))$.

Damos un ejemplo de la práctica matemática. Suponga que Γ consiste en los siguientes axiomas de la teoría de grupos abelianos:

i. $\forall x \forall y (x+y = y+x)$

ii. $\forall x (x+e = x)$,

entonces $\Gamma \vdash \forall y [\forall x (x+y=x) \supset y=e]$.

Demostración

1. $\forall x \forall y (x+y = y+x)$	P $\in \Gamma$
2. Pt x $(x+e = x)$	P $\in \Gamma$
3. $\forall x (x+y = x)$	P
4. $e+y = e$	E \forall ,3
5. $y+e = y$	E \forall ,2
6. $\forall y (e+y = y+e)$	E \forall ,1
7. $e+y = y+e$	E \forall ,6
8. $e+y = y$	SI,7,5
9. $y = e$	SI,8,4

Por TD: $\Gamma \vdash \forall x (x+y=x) \supset y=e$

Por G \forall : $\Gamma \vdash \forall y [\forall x (x+y=x) \supset \wedge = e]$

Es claro pues que la asociatividad no se necesita en la demostración de la unicidad del neutro.

Ejercicios

1. Demuestre formalmente:

- $\forall x \forall y \phi(x,y) \vdash \forall y \forall x \phi(x,y), \exists x \exists y \phi(x,y) \vdash \exists y \exists x \phi(x,y)$
- $\forall x \forall y \phi(x,y) \vdash \forall x \phi(x,x)$
- $\vdash \forall x (P(x) \supset \exists y P(y))$
- $\vdash \forall x \exists y (y=f(x))$
- $\forall x (\phi \leftrightarrow \psi) \vdash \exists x \phi \leftrightarrow \exists x \psi$
- $\forall x \phi \vee \forall x \psi \vdash \forall x (\phi \vee \psi)$
- ${}^+ \exists x (\phi \vee \psi) \vdash \exists x \phi \vee \exists x \psi$
- ${}^+ \exists x (\psi \supset \phi(x)) \vdash \psi \supset \exists x \phi(x) \quad x \text{ no ocurre libre en } \psi$
- ${}^+ \forall x (\phi(x) \supset \psi) \vdash \forall x \phi(x) \supset \psi \quad x \text{ no ocurre libre en } \psi$
- ${}^+ \exists x (\phi(x) \supset \psi) \vdash \forall x \phi(x) \supset \psi \quad x \text{ no ocurre libre en } \psi$
- $\exists x \phi \supset \exists x \psi \vdash \exists x (\phi \supset \psi)$
- $\forall x (Q(x) \supset \forall y P(y)), Q(f(y,y)) \vdash \forall y P(f(y,y))$
- $\forall x (\phi(x) \cap \psi(x)) \vdash \forall x \phi(x) \cap \forall y \phi(y)$
 $\vdash \forall x (\phi(x) \cap \forall y \phi(y))$

2. Son correctas las siguientes aplicaciones de E \forall ?

- $\forall x \forall z [R(x,z) \supset \exists y R(f(w,y),y)] \vdash \forall z [R(f(w,y),z) \supset \exists y R(f(w,y),y)]$
- $\forall x \exists y R(x,y) \vdash \exists y R(y,y)$
- $\forall x \exists y R(x,y) \vdash \exists x R(x,x)$
- $\forall x. pt\ y\ (y=x) \vdash \forall y\ (y=y)$

3. Demuestre formalmente: $\Gamma \vdash c=0$, donde Γ consiste en

$$\forall x (x+0=x)$$

$$\forall x \forall y \forall z (x+y = x+z \supset y=z)$$

$$c + c = c$$

Se trata de demostrar que para una operación con neutro a derecha y con la propiedad cancelativa a la izquierda, el único idempotente es la identidad.

4. Dadas las fórmulas:

$$\phi_1: \forall x \forall y \forall z (x < y \vee y < z \supset x < z)$$

$$\phi_2: \forall x \forall y (x < y \supset \neg y < x)$$

$$\phi_3: \forall x \neg(x < x)$$

Demuestre: $\phi_1 \supset \phi_2 \Leftrightarrow \phi_3$.

- 5.* Sea D el sistema deductivo que consiste de los axiomas 1,2,3,5,7,8, más el axioma adicional:

$$\forall x (\psi \supset \phi^*x) \supset (\psi \supset \forall x \phi(x))$$

y las reglas MP y G \forall . Demuestre que se es equivalente al sistema del texto (Ayuda: muestre que TD vale en el nuevo sistema).

- 6.* Demuestre que si se considera el cuantificador \exists como un símbolo primitivo y no una abreviación, el sistema que resulta de añadir los nuevos axiomas:

$$9. \phi(t) \supset \exists x \phi(x) \text{ si } t \text{ libre para } x \text{ en } \phi(x)$$

$$10. \forall x (\phi(x) \supset \psi) \supset (\exists x \phi(x) \supset \psi) \text{ si } x \text{ no ocurre libre en } \psi$$

es equivalente al original.

- 7.* Sea x una variable que no ocurre libre en Γ , demuestre que si $\Gamma, \phi(x) \vdash \psi$ y x no ocurre libre en ψ , entonces $\Gamma, \exists x \phi(x) \vdash \psi$.

- 8.* Demuestre que si reemplaza el axioma

8: $t_1=t_2 \supset (\phi(t_1) \supset \phi(t_2))$

por el par de axiomas

$$8.1 x=y \supset (R(x_1, \dots, x_n) \supset R(x_1, \dots, y, \dots, x_n))$$

$$8.2 x=y \supset (f(x_1, \dots, x_n) = f(x_1, \dots, y, \dots, x_n))$$

por cada símbolo de relación R y función f , n -ádicos, el sistema resultante es deductivamente equivalente al original. (Ayuda: por inducción en términos muestre que $t_1=t_2 \vdash t(\dots t_1 \dots) = t(\dots t_2 \dots)$. Luego, por inducción en fórmulas demuestre que $t_1=t_2 \vdash \phi(\dots t_1 \dots) \supset \phi(\dots t_2 \dots)$.

- 9.* Demuestre que si reemplaza el axioma 5: $\forall x \phi(x) \supset \phi(t)$, por los tres axiomas

$$5.1 \forall x \phi(x) \supset \phi(y) \text{ y variable para } x \text{ en } \phi(y)$$

$$5.2 \forall x \phi(x) \supset \phi(c) \text{ } c \text{ constante}$$

$$5.3 \forall x \phi(x) \supset \phi(f(x_1, \dots, x_n)) \text{ } f(x_1, \dots, x_n) \text{ libre para } x \text{ e } \phi.$$

El sistema resultante es equivalente al original.

- 10.* Dé un algoritmo para listar el conunto de fórmulas sobre el léxico $I = \{R^2\}$ deducibles sin premisas.

(Ayuda: conviene dar primero un algoritmo que genere todas las deducciones formales).

- 11*. Si Γ es un conjunto infinito de fórmulas, dé un algoritmo para enumerar el conjunto $\{\phi \mid \Gamma \vdash \phi\}$

- 12*** Puede dar un algoritmo para decidir si dada una fórmula sobre el léxico $I = \{R^2\}$ se tiene $\vdash \phi$ o $\not\vdash \phi$?

3.2 Validez del sistema axiomático

Como ya hemos observado, los axiomas del Cálculo de Predicados son lógicamente válidos y por lo tanto todo lo que se deduzca en el sistema sigue siendo válido. Examinamos aquí con mas cuidado esta afirmación.

Definición. Sea Γ una colección de sentencias, una sentencia ϕ es Γ -válida si es verdadera en todos los modelos de Γ . Expresamos este hecho por $\Gamma \models \phi$.

En el caso $\Gamma = \emptyset$, toda estructura es modelo de Γ , por lo tanto una sentencia es \emptyset -válida si es lógicamente válida.

Por ejemplo, si Γ consiste de los axiomas de la teoría de grupos:

$$\forall x \forall y \forall z [x \cdot (y \cdot z) = (x \cdot y) \cdot z]$$

$$\forall x (x \exists = x \wedge e \cdot x = x)$$

$$\forall x \exists y [x \cdot y = e \wedge y \cdot x = e],$$

sus modelos serán evidentemente los grupos, por lo tanto las sentencias Γ -válidas serán aquellas verdaderas en todos los grupos. Así:

$$(1) \forall x (\forall y (y \cdot x = y \wedge x \cdot y = y) \supset x = e))$$

$$(2) \exists x_1 \exists x_2 \exists x_3 \exists x_4 \exists x_5 \forall x (\vee_{i=1}^5 x = x_i) \supset \forall x \forall y (x \cdot y = y \cdot x)$$

son Γ -válidas, como se muestra en los cursos de álgebra abstracta. Allí se demuestra que en cada grupo es cierto que:

(1) "La identidad es única"

(2) "Si el grupo tiene a lo sumo 5 elementos es conmutativo"

El sistema deductivo nos da un aparato infalible para producir sentencias Γ -válidas, por el siguiente teorema.

Teorema de Validez Si $\Gamma \vdash \neg \phi$ entonces $\Gamma \models \phi$.

Para demostrar el Teorema necesitamos primero:

Lema. Si ϕ es un axioma del C. de Predicados entonces $\models \phi$.

Demostración. Los tres primeros axiomas tiene la forma de una tautología, probablemente precedida de cuantificadores universales, y hemos visto en el Cap. 2, Sec. 2.1 que toda fórmula con forma de tautología es lógicamente

válida. La validez de los axiomas 5 y 8 ha sido discutida en la misma sección, y la de los demás es evidente. ■

(c)

Demostración del Teorema. Sea ϕ_1, \dots, ϕ_n una Γ -deducción de ϕ , y supongamos que todas las variables libres de las ϕ_i están incluidas en x_1, \dots, x_r . Sea \mathcal{R} un modelo arbitrario de Γ y $\bar{a} = (a_1, \dots, a_r)$ una lista de elementos de \mathcal{R} . Demostramos por inducción en $k \leq n$ que $\mathcal{R} \models \phi_k[\bar{a}]$. Como ϕ_n es ϕ una sentencia, tenemos $\mathcal{R} \models \phi[\bar{a}]$ y así $\mathcal{R} \models \phi$.

- I. a) ϕ_1 es un axioma, entonces $\mathcal{R} \models \phi_1[\bar{a}]$ por el Lema.
- b) $\phi_1 \in \Gamma$, entonces $\mathcal{R} \models \phi_1$ por hipótesis y trivialmente $\mathcal{R} \models \phi_1[\bar{a}]$.
- II. Suponga $\mathcal{R} \models \phi_l[\bar{a}]$ para I. Si ϕ_k es axioma o pertenece a Γ es igual que en I. Si ϕ_k viene por Modus Ponens de ϕ_i y $\phi_j = (\phi_i \supset \phi_k)[\bar{a}]$, por tanto $\mathcal{R} \models \phi_k[\bar{a}]$. ■

Corolario. Si $\vdash \phi$ entonces $\models \phi$.

Con lo anterior tenemos un método para demostrar resultados de no deducibilidad por medio de contraejemplos.

Ejemplos

(a) $\forall x (P(x) \vee Q(x)) \not\models \forall x P(x) \vee \forall x Q(x)$.

Pues la estructura $\langle \mathbb{N}, \underline{P}, \underline{Q} \rangle$ donde \underline{P} es el conjunto de los pares y \underline{Q} el de los impares es modelo de la premisa y no de la conclusión.

(b) Considere el silogismo:

$$\begin{array}{ccc} \text{Todo } P \text{ es } Q & & \exists x (P(x) \supset Q(x)) \\ \hline \text{algun } Q \text{ no es } S & & \exists x (Q(x) \wedge \neg S(x)) \\ \hline \text{algun } P \text{ no es } S & & \exists x (P(x) \wedge \neg S(x)) \end{array}$$

Su invalidez, se demuestra considerando la estructura $\mathcal{R} = \langle \{a, b, c, d\}, \underline{P}, \underline{Q}, \underline{S} \rangle$ donde $\underline{P} = \{a\}$, $\underline{Q} = \{a, b, c\}$ y $\underline{S} = \{a, b\}$. Entonces $P \subseteq Q$; que implica $\mathcal{R} \models \forall x (P(x) \supset Q(x))$; $\mathcal{R} \models Q[c] \wedge \neg S[c]$ que implica $\mathcal{R} \models \exists x (Q(x) \wedge \neg S(x))$. Sin embargo, $\mathcal{R} \not\models \exists x (P(x) \wedge \neg S(x))$.

(c) Los tres axiomas dados al comienzo de la sección para la Teoría de Grupos son *independientes*, es decir ninguno de ellos se puede deducir de los otros dos; por ejemplo la estructura $\{1,2\}, f_1$ donde la función binaria f está dada por la tabla

f	1	2
1	1	1
2	1	1

es modelo de los axiomas (i) y (iii) pero no del (ii), por lo tanto éste no es deducible de (i) y (iii).

Nuestro sistema deductivo produce sentencias Γ -válidas. ¿Es suficientemente fuerte para producir todas las Γ -válidas de un conjunto Γ ? ¿Es posible deducir de los axiomas de la Teoría de Grupos todas las sentencias verdaderas en todos los grupos? por ejemplo la sentencia (2) indicada al comienzo de la sección? La respuesta es que sí y este es uno de los hechos más famosos e importantes de la Lógica. Se podría decir que la Lógica moderna comienza con este resultado, la Tesis de Doctorado de K. Gödel.

Teorema de Completitud (Gödel, 1930). Si $\Gamma \models \phi$ entonces $\Gamma \vdash \phi$.

No daremos aquí la demostración del teorema, y evitaremos hacer uso del mismo por ahora, pero en las secciones siguientes continuaremos desarrollando el sistema para convencernos de su poder.

Ejercicios

1. Demuestre:

- $\forall x P(x) \supset \forall x Q(x) \not\models \neg \forall x (P(x) \supset Q(x))$
- $\exists x (P(x) \supset Q(x)) \not\models \exists x P(x) \supset \exists x Q(x)$
- $\exists x P(x) \Leftrightarrow \exists x Q(x) \not\models \exists x (P(x) \Leftrightarrow Q(x))$
- $\forall x P(x) \Leftrightarrow \forall x Q(x) \models \neg \forall x (P(x) \Leftrightarrow Q(x))$

- e. $\exists x P(x) \wedge \exists x Q(x) \not\vdash \neg \exists x (P(x) \wedge Q(x))$
- f. $\forall x \forall y \forall z [(R(x,y) \supset R(y,x)) \wedge (R(x,y) \wedge R(y,z) \supset R(x,z))] \not\vdash \forall x R(x,x)$
- 2.** Demuestre que si ϕ es una sentencia sin cuantificadores ni el símbolo $=$, y $\vdash \neg \phi$ entonces tiene la forma de una tautología.
3. Demuestre que ni la fórmula siguiente ni su negación son deducibles ($\not\vdash \phi, \not\vdash \neg \phi$)
- $$[\exists x P(x) \wedge (P(c) \supset \exists x Q(x))] \supset \exists x Q(x)$$
4. Demuestre:
- $P(0), \forall x (P(x) \supset P(x+1)) \not\vdash \forall x P(x)$
 - $P(0), P(1), P(1+1), P(1+1+1) \dots \not\vdash \forall x P(x)$
- 5.* Demuestre que $\Gamma \not\vdash \forall x (x \neq 0 \supset \exists y (y < x \wedge \forall z (z < x \supset z \leq y)))$, donde Γ consiste de
- $\forall x \forall y \forall z (x < y \wedge y < z \supset x < z)$
 - $\forall x \forall y (x < y \supset \neg (y < x))$
 - $\forall x \forall y (x \neq y \supset (x < y \vee y < x))$
 - $\forall x (x \neq 0 \supset 0 < x)$
 - $\forall x \exists y (x < y \wedge \forall z (x < z \supset (y = z \vee y < z)))$
- la teoría de conjuntos estrictamente ordenados con primer elemento 0, y donde cada elemento tiene un sucesor inmediato.
6. Explique por qué las afirmaciones siguientes son incorrectas:
- $\forall x \exists y R(x,y) \vdash \neg \exists y R(f(x,y),y)$
 - $\forall y (y(1,y)) \vdash \neg \exists x \forall y (y)$
 - $\exists x (R(x,y) \supset S(x,y,z)) \vdash \neg T(a,y) \supset S(a,y,z)$
 - Si $\phi(x,y) \vdash \exists y \phi(x,y)$, entonces $\phi(x,y) \vdash \forall x \exists y \phi(x,y)$

- e. Si $\Gamma \vdash \exists x (\phi(x) \vee \psi(x))$ entonces $\Gamma \vdash \exists x \phi(x) \vee \Gamma \vdash \exists x \psi(x)$
- f. $\forall x \forall y R(x,y) \not\vdash \forall y R(y,y)$
7. Demuestre que los axiomas (i) y (iii) de la Teoría de Grupos son independientes de los otros dos.

3.3 Especificación existencial

En la práctica se usa el siguiente modo de argumento. Dada una afirmación existencia $\exists x \phi(x)$, se dice "Sea c tal que $\phi(c)$ ", es decir se escoge un individuo con la propiedad ϕ y se continúa la deducción. Veremos que esto se puede realizar en el sistema formal siempre que se tenga cierto cuidado, pues éste no es un paso lógico e el sentido usual; consiste realmente en introducir una nueva premisa en la deducción, la cual posteriormente podrá ser eliminada para ciertas consecuencias.

Lema. (Generalización de constantes) *Si c es una constante que no aparece en Γ y $\Gamma \vdash \phi(c)$ entonces; $\Gamma \vdash \forall y \phi(y)$ para alguna variable y .*

Demostración. Si ϕ_1, \dots, ϕ_n es una Γ -deducción de $\phi(c)$, sea y una variable que no ocurre en ningún ϕ_i . Sea ϕ_i^* el resultado de substituir todas las ocurrencias de c en ϕ_i por y , entonces $\phi_i^* = \phi_i$ para $\phi_i \in \Gamma$. Por lo tanto $\phi_1^*, \dots, \phi_n^*$ es una Γ -deducción de $\phi_n \lambda \phi(y)$ por $G\forall$. ■

Corolario. *Si $\Gamma \vdash \exists x \phi(x)$ y $\Gamma, \phi(c) \vdash \psi$, donde c es una constante que no ocurre en Γ ni en ψ , entonces $\Gamma \vdash \psi$.*

Demostración. Si $\Gamma, \phi(c) \vdash \psi$ entonces $\Gamma \vdash \phi(c) \supset \psi \vdash \neg \psi \supset \neg \phi(c)$ por prop, entonces $\Gamma, \neg \psi \vdash \neg \phi(c)$. Por el lema anterior $\Gamma, \neg \psi \vdash \forall y \neg \phi(y)$ para alguna y ; pero $\forall y \neg \phi(y) \vdash \forall x \neg \phi(x)$, y tenemos finalmente por Prop: $\Gamma, \neg \forall x \phi(x) \vdash \psi$. Como $\Gamma \vdash \exists x \phi(x)$ tenemos $\Gamma \vdash \psi$. ■

El resultado anterior nos permite usar la siguiente regla que llamaremos *especificación existencial* (E \exists):

$\exists x \phi(x, \vec{y})$

$\phi(c, \vec{y})$ (P,c), c constante que no ha aparecido antes en la deducción.

La marca (P,c) indica que $\phi(c, \vec{y})$ es realmente una premisa, de la cual dependen los pasos posteriores de la deducción en *los que aparezca c*. Esto significa que en ninguna consecuencia en la que aparezca c se puede aplicar generalización universal con respecto a c misma. Por otra parte, los pasos posteriores de la deducción en los que *no aparece c* no dependen de la premisa $\phi(c, \vec{y})$, por el Corolario anterior, y se pueden considerar deducidos directamente de las premisas de las que se dedujo $\exists x \phi(x, \vec{y})$, por lo tanto para ellos no hay restricción en la aplicación de G \forall , excepto las que provengan de Γ .

Ejemplo 1

$$\exists x \forall y R(x,y) | — \forall y \exists x R(x,y)$$

Demostración

- | | |
|---------------------------------|--------------------------------|
| 1. $\exists x \forall y R(x,y)$ | P |
| 2. $\forall y R(c,y)$ | $E\exists (P,c)$ |
| 3. $R(c,y)$ | $E\forall$ |
| 4. $\exists x R(x,y)$ | $G\exists$ no depende de (P,c) |
| 5. $\forall y \exists x R(x,y)$ | $G\forall$ no depende de (P,c) |

En cambio la siguiente "deducción" es incorrecta

- | | |
|---------------------------------|--------------------|
| 1. $\exists x \forall y R(x,y)$ | P |
| 2. $\forall y R(x,y)$ | $E\forall$ |
| 3. $R(x,c)$ | $E\exists (P,c)$ |
| 4. $\forall x R(x,c)$ | $G\forall$ ← error |
| 5. $\exists y \forall x R(x,y)$ | $G\exists$ |

Como la línea 3 es realmente una premisa adicional en la que x ocurre libre, no se puede aplicar $G\forall$ con respecto a x en la línea siguiente en la que c aparece.

Ejemplo 2

$$\forall x \exists y \forall z R(x,y,z) \mid\!\!-\! \forall x \exists y R(x,y,y).$$

Demostración

- | | |
|---|---|
| 1. $\forall x \exists y \forall z R(x,y,z)$ | P |
| 2. $\exists y \forall z R(x,y,z)$ | $\text{E}\forall, 1$ |
| 3. $\forall z R(x,c,z)$ | $\text{E}\exists, 2 (P,c)$ |
| 4. $R(x,c,c)$ | $\text{E}\forall, 3$ |
| 5. $\exists y R(x,y,y)$ | $\text{G}\exists, 4 \text{ no depende de } (P,c)$ |
| 6. $\forall x \exists y R(x,y,y)$ | $\text{G}\forall, 5 \text{ no depende de } (P,c)$ |

Ejemplo 3

Considere el siguiente silogismo aristotélico:

$$\begin{array}{c}
 \text{Algún A es B} \\
 \text{Todo B es C} \\
 \hline
 \text{Algún A es C}
 \end{array}
 \qquad
 \begin{array}{c}
 \exists x (A(x) \wedge B(x)) \\
 \forall x (B(x) \supset C(x)) \\
 \hline
 \exists x (A(x) \wedge C(x))
 \end{array}$$

y su deducción formal:

- | | |
|------------------------------------|--|
| 1. $\exists x (A(x) \wedge B(x))$ | P |
| 2. $(A(c) \wedge B(c))$ | $\text{E}\exists, 1 (P,c)$ |
| 3. $B(c)$ | Prop, 2 |
| 4. $\forall x (B(x) \supset C(x))$ | P |
| 5. $B(c) \supset C(c)$ | $\text{E}\forall, 4.$ |
| 6. $C(c)$ | MP 3,5 |
| 7. $A(c) \wedge C(c)$ | Prop 2,6 |
| 8. $\exists x (A(x) \wedge B(x))$ | $\text{G}\exists \text{ no depende de } (P,c)$ |

Ejercicios

1. Demuestre

- $\exists x \forall y \phi(x,y) \vdash \neg \exists x \phi(x,x)$
- $\exists x (\phi \wedge \psi) \vdash \exists x \phi \wedge \exists x \psi$
- $\exists x (\phi \vee \psi) \vdash \neg \exists x \phi \vee \exists x \psi$
- $\forall x (\phi(x) \supset \psi) \vdash \neg \exists x \phi(x) \supset \psi$

2. Simbolice y demuestre formalmente la validez de los siguientes silogismos aristotélicos:

todo A es B	Algún A es B	Algún A es B
<u>Ningún C es B</u>	<u>Todo B es C</u>	<u>Ningún C es B</u>
Ningún A es C	Algún C es A	Algún A no es C

3. Demuestre que $\Gamma \vdash \forall x (D(x) \supset \neg Q(x))$ cuando Γ es

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

$$\exists x (P(x) \wedge \forall y (D(y) \supset R(x, y)))$$

4. Simbolice y deduzca formalmente.

a. 2 es primo. Para todo primo hay un primo mayor que él. Todo primo mayor que 2 es impar. Por lo tanto existen primos impares.

b. El Amazonas es un río. Todos los ríos son masas saladas o de agua dulce. En el Amazonas habitan tiburones verdes. Los tiburones verdes no pueden habitar en masas de agua salada. En conclusión, el Amazonas es una masa de agua dulce.

c. Quien tiene amigos tiene enemigos, luego quien no tiene enemigos tampoco tiene amigos.

5. Demuestre:

a. $P(t) \vdash \neg \exists z (t=z \wedge P(z))$

b. $\forall x \forall y (R(x,y) \wedge R(y,x) \supset x = y)$

$$\underline{\forall x \exists y (R(x,y) \wedge x \neq y)}$$

$$\neg \exists y \forall x R(x,y)$$

c. $\forall x \forall y (x \neq y \supset R(x,y) \vee R(y,x))$

$$\underline{\forall x \forall y \forall z (R(x,y) \wedge R(y,z) \supset R(x,z))}$$

$$\neg \exists x (R(x,x) \leftrightarrow \neg \exists x \exists y (R(x,y) \wedge R(y,x)))$$

d. $\forall x \forall y \exists z (x + z = y)$

$$\underline{x (\exists y (y+x=y) \supset x=0)}$$

$$\forall y (y+0=y)$$

6. Sea Γ una colección de sentencias. ¿Cuáles de las afirmaciones siguientes son correctas?

a. Si $\Gamma, \phi(x) \vdash \psi(x)$ entonces $\Gamma, \phi(x) \vdash \forall x \psi(x)$

b. Si $\Gamma, \phi(x) \vdash \psi(x)$ entonces $\Gamma, \forall x \phi(x) \vdash \forall x \psi(x)$

c. Si $\Gamma, \phi(x) \vdash \psi(x)$ entonces $\Gamma, \exists \phi(x) \vdash \psi(x)$

d. Si $\Gamma, \phi(x) \vdash \forall x \psi(x)$ entonces $\Gamma, \exists x \phi(x) \vdash \forall x \psi(x)$

3.4 Equivalencia, Forma Prenexa

Si ϕ es una fórmula con variables libres x_1, \dots, x_n , su clausura universal $\forall \forall \phi$, es la sentencia $\forall x_1, \dots, x_n \phi$. Sean γ y γ' fórmulas, y para cada fórmula sea $\phi(\gamma/\gamma')$ el resultado de substituir alguna ocurrencia de γ en ϕ , com subfórmula, por γ' .

Teorema 1. $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \phi \leftrightarrow \phi(\gamma/\gamma')$

Demostración. Primero hacemos notar que por el C de Proposiciones tenemos:

1. $\phi \leftrightarrow \psi \vdash \neg \phi \leftrightarrow \neg \psi$

$$2. \phi_1 \leftrightarrow \psi_1, \phi_2 \leftrightarrow \psi_2 \vdash \phi_2 \leftrightarrow \psi_1 \square \psi_2$$

Además, por T7, § 3.1,

$$3. \forall x (\phi \leftrightarrow \psi) \vdash \forall x \phi \leftrightarrow \forall x \psi$$

Probamos el teorema por inducción en ϕ , dejando fijas γ y γ' y llamando ϕ' a $\phi(\gamma/\gamma')$. Como suponemos que es subfórmula de ϕ , comenzamos la inducción con ϕ igual a γ ; en tal caso ϕ' es γ' y la afirmación se reduce a $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \gamma \leftrightarrow \gamma'$, que resulta por E \forall . Para el paso inductivo podemos suponer que γ es subfórmula propia de ϕ .

Suponga la afirmación cierta para ϕ_1 y ϕ_2 , es decir: $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \phi_1 \leftrightarrow \phi'_1, \phi_2 \leftrightarrow \phi'_2$. Entonces por (2), $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \phi_1 \square \phi_2 \leftrightarrow \phi'_1 \square \phi'_2$. Como $(\phi_1 \square \phi_2)' = \phi'_1 \square \phi'_2$, tenemos el paso inductivo para \square . Suponga que $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \phi \leftrightarrow \phi'$, entonces por (1) tenemos $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \neg \phi \leftrightarrow \neg \phi'$. Además, por (3) y G \forall que se puede aplicar a cualquier variable en este caso; $\forall \forall (\gamma \leftrightarrow \gamma') \vdash \forall x (\phi \leftrightarrow \phi') \vdash \forall x \leftrightarrow \forall x \phi'$, pero $\forall x \phi'$ es $(\forall x \phi)'$. Así hemos completado el paso de inducción para \neg y $\forall x$. La afirmación vale para toda fórmula ϕ , incluyendo las que contenga \exists pues ésta es una abreviación. ■

Igual que en el Cálculo de Proposiciones, definimos $\phi \text{ eq } \psi$ si y solo si $\vdash \phi \equiv \psi$ y $\psi \equiv \phi$. Obviamente, $\phi \text{ eq } \psi$ si y solo si $\vdash \phi \leftrightarrow \psi$. Por lo tanto tenemos el siguiente corolario.

Corolario 2. Si $\gamma \text{ eq } \gamma'$ entonces $\phi \text{ eq } \phi(\gamma/\gamma')$

Demostración. $\gamma \text{ eq } \gamma'$ implica $\vdash \gamma \leftrightarrow \gamma'$. Por G $\forall \vdash \forall \forall (\gamma \leftrightarrow \gamma')$ y por el anterior teorema $\vdash \phi \leftrightarrow \phi'$, es decir $\phi \text{ eq } \phi'$. ■

Este corolario nos permite producir equivalencias por sustitución de equivalencias. Por ejemplo, de la equivalencia ya demostrada:

$$\forall x (\phi \wedge \psi) \text{ eq } \forall x \phi \wedge \forall x \psi$$

podemos deducir:

$\exists x (\phi \vee \psi)$	eq	$\neg \forall x \neg (\phi \vee \psi)$	Definición
	eq	$\neg \forall x [\neg \phi \wedge \neg \psi]$	Substitución
	eq	$\neg [\forall x \neg \phi \wedge \forall x \neg \psi]$	Substitución

$$\text{eq } \neg \forall x \neg \phi \vee \neg \forall x \neg \psi \quad \text{Proposición}$$

$$\text{eq } \exists x \phi \vee \exists x \psi \quad \text{Definición}$$

Damos en seguida una lista de equivalencias básicas que nos permitirán "extraer" los cuantificadores de una fórmula. El *dual* del cuantificador \forall es \exists , es *dual* de \exists es \forall . Suponga que Q denota cualquiera de los cuantificadores, entonces \tilde{Q} denota su dual.

Lema 3 (Extracción de cuantificadores)

$$(a) \neg Qx \phi(x) \text{ eq } \tilde{Q} \neg \phi(x)$$

Suponga que x no ocurre libre en ψ , entonces

$$(b) \psi \supset Qx \phi(x) \text{ eq } Qx (\psi \supset \phi(x))$$

$$(c) Qx \phi(x) \supset \psi \text{ eq } \tilde{Q} (\phi(x) \supset \psi)$$

$$(d) \psi \wedge Qx \phi(x) \text{ eq } Qx (\psi \wedge \phi(x))$$

$$(e) \psi \vee Qx \phi(x) \text{ eq } Qx (\psi \vee \phi(x))$$

Demostración

$$(a) (i) \neg \forall x \phi \text{ eq } \neg \forall x \neg \neg \phi \text{ eq } \exists x \neg \phi$$

$$(ii) \neg \exists x \phi \text{ eq } \neg \neg \forall x \neg \phi \text{ eq } \forall x \neg \phi$$

$$(b) (i) \psi \supset \forall x \phi(x) \text{ eq } \forall x (\psi \supset \phi(x)), \text{ es T9}$$

$$(ii) \psi \supset \exists x \phi(x) \text{ eq } \exists x (\psi \supset \phi(x))$$

- " \Rightarrow "
1. $\psi \supset \exists x \phi(x)$ P
 2. $\neg \exists x (\psi \supset \phi(x))$ P
 3. $\forall x \neg (\psi \supset \phi(x))$ (a)
 4. $\neg (\psi \supset \phi(x))$ E \forall
 5. ψ Prop.
 6. $\neg \phi(x)$ Prop.
 7. $\forall x \neg \phi(x)$ G $\forall, 6$

8. $\neg \exists x \phi(x)$ (a)
 9. $\exists x \phi(x)$ Prop 1,5

El resultado se sigue por R.A.

- " \Leftarrow " 1. $\exists x (\psi \supset \phi(x))$ P
 2. $\psi \supset \phi(c)$ $E\exists (P,c)$
 3. ψ P
 4. $\phi(c)$ MP
 5. $\exists x \phi(x)$ $G\exists$ no depende de (P,c)

Por TD: $\exists x (\psi \supset \phi(x)) \vdash \neg \psi \supset \exists x \phi(x)$

- (c) $\forall x \phi(x) \supset \psi$ eq $\neg \psi \supset \neg \forall x \phi(x)$ Prop.
 eq $\neg \psi \supset \forall x \neg \phi(x)$ (a)
 eq $\forall x (\neg \psi \supset \neg \phi(x))$ (b)
 eq $\forall x (\phi(x) \supset \psi)$ Prop.

Note que en este caso \forall sale como \exists y \exists sale como \forall , es decir

$$\begin{aligned} \forall x \phi(x) \supset \psi &\text{ eq } \exists x (\phi(x) \supset \psi) \\ \exists x \phi(x) \supset \psi &\text{ eq } \forall x (\phi(x) \supset \psi) \end{aligned}$$

- (d) $\forall x \phi(x) \wedge \psi$ eq $\neg [\forall x \phi(x) \neg \psi]$ Prop.
 eq $\neg [\forall x \phi(x) \supset \neg \psi]$ (c)
 eq $\forall x \neg [\phi(x) \neg \psi]$ (a)
 eq $\forall x [\phi(x) \wedge \psi]$ Prop.

(e) Ejercicio, sale de (b) por equivalencias. ■

Las equivalencias del Lema 3 nos permiten correr un cuantificador hacia la izquierda de manera que aumente la longitud de la fórmula bajo su alcance, pero sólo en el caso de que la fórmula ψ no contenga la variable cuantificada x libre. Sin embargo esta última restricción se puede evitar, pues si x aparece libre en ψ podemos cambiar la variable cuantificada con la ayuda de la equivalencia:

$$Qx \phi(x) \text{ eq } Qy \phi(y)$$

que resulta de T6 y T7 si se utiliza una variable nueva y. Por ejemplo:

$$\begin{aligned} \psi(x) \vee \exists x \phi(x) &\text{ eq } \psi(x) \vee \exists y \phi(y) \text{ eq } \exists y (\psi(x) \vee \phi(y)) \\ \forall x \phi(x) \supset \psi(x) &\text{ eq } \forall y \phi(y) \supset \psi(x) \text{ eq } \exists y (\phi(y) \supset \psi(x)) \end{aligned}$$

Aplicando sistemáticamente estas equivalencias a una fórmula cualquiera podemos extraer todas sus ocurrencias de cuantificadores a la izquierda de manera que ningún cuantificador esté bajo el alcance de un conectivo, de ahí el siguiente teorema.

Teorema (Forma prenexa) *Toda fórmula del Cálculo de Predicados es deductivamente equivalente a una fórmula en forma prenexa, es decir de la forma $Q_1x_1 \dots Q_nx_n \psi(x_1, \dots, x_n)$, en donde cada Q_i es uno de los cuantificadores $\forall o \exists$, y $\psi(x_1, \dots, x_n)$ no tiene cuantificadores.*

Ejemplos

1. Hallar una forma prenexa de $\forall x P(x) \leftrightarrow \exists x Q(x)$.

$$\begin{aligned} \forall x P(x) \leftrightarrow \exists x Q(x) &\text{ eq } [\forall x P(x) \supset \exists x Q(x)] \wedge [\exists x Q(x) \supset \forall x P(x)] \\ &\text{ eq } \exists x [P(x) \supset \exists x Q(x)] \wedge \forall x [Q(x) \supset \forall x P(x)] \\ &\text{ eq } \exists x [P(x) \supset \exists y Q(y)] \wedge \forall x [Q(x) \supset \forall y P(y)] \\ &\text{ eq } \exists x \exists y [P(x) \supset Q(y)] \wedge \forall x \forall y [Q(x) \supset P(y)] \\ &\text{ eq } \exists x \{\exists y [P(x) \supset Q(y)]\} \wedge \forall x \forall y [Q(x) \supset P(y)] \\ &\text{ eq } \exists x \exists y \{[P(x) \supset Q(y)] \wedge \forall x \forall y [Q(x) \supset P(y)]\} \\ &\text{ eq } \exists x \exists y \{[P(x) \supset Q(y)] \wedge \forall z \forall w [Q(z) \supset P(w)]\} \\ &\text{ eq } \exists x \exists y \forall z \forall w \{[P(x) \supset Q(y)] \wedge [Q(z) \supset P(w)]\} \end{aligned}$$

2. Hallar la forma prenexa de $\phi: \neg \exists x [\forall x R(x,y) \supset \neg \exists y \forall w S(y,w,x)]$

$$\begin{aligned} \phi &\text{ eq } \neg \exists x \exists y [R(x,y) \supset \forall w \neg S(y,w,x)] \\ &\text{ eq } \neg \exists x \exists y [R(x,y) \supset \forall z \exists w \neg S(z,w,x)] \end{aligned}$$

$$\text{eq } \neg \exists x \exists y \forall z \exists w [R(x,y) \supset \neg S(z,w,x)]$$

$$\text{eq } \forall x \forall y \exists z \forall w \neg [R(x,y) \supset \neg S(z,w,x)]$$

$$\text{eq } \forall x \forall y \exists z \forall w [R(x,y) \wedge S(z,w,x)]$$

5.

3. La forma prenexa no es única, puede variar, dependiendo del orden de extracción de los cuantificadores. Por ejemplo:

$$\exists x P(x) \supset \exists x Q(x) \text{ eq } \forall x (P(x) \supset \exists y Q(y)) \text{ eq } \forall x \exists y (P(x) \supset Q(y))$$

$$\exists x P(x) \supset \exists x Q(x) \text{ eq } \exists x (\exists x P(x) \supset Q(x)) \text{ eq } \exists x \forall y (P(y) \supset Q(x))$$

6.

Ejercicios

1. Demuestre que $\text{Qx } \phi(x) \Leftrightarrow \psi \rightarrow \text{eq } \text{Qx } (\phi(x) \Leftrightarrow \psi)$, aún si x no ocurre libre en ψ .

8.

2. Demuestre parte (e) del lema 3.

3. Determine para qué combinaciones del cuantificador Q y el conectivo \square se tiene $\text{Qx } \phi(x) \square \text{Qx } \psi(x) \text{ eq } \text{Qx } (\phi(x) \square \psi(x))$

4. Halle una forma prenexa de las fórmulas siguientes:

a. $\forall x R(x,y) \vee \forall x Q(x,x)$

b. $\forall x P(x) \wedge \exists x Q(x)$

c. $\exists x \forall y R(x,y) \supset \neg \forall w \forall y R(w,y)$

d. $\forall x P(x) \supset (\forall x Q(x) \supset \forall x S(x))$

e. $\forall x \{2 \wedge \forall y \forall z [y.z = x \supset (y=x \vee y=1)]\}, \text{imp } \exists y ((y+y)+1 = x)\}$

Esta fórmula "dice" que todo primo mayor que 2 es impar.

f. $((\forall x P(x) \supset \forall y Q(y)) \supset \forall y S(y)) \supset \forall y T(y)$.

g. $\forall x ((\exists y Q(y,x)) \supset \exists w (Q(w,x) \wedge \forall z (Q(z,w) \supset \neg Q(z,x))))$

7.

5. Con la ayuda de equivalencias conocidas o la forma prenexa demuestre:
- $\vdash \exists x (P(x) \supset \forall x P(x))$
 - $\vdash \neg \forall x \exists y (P(x) \supset P(y))$
 - $\vdash \neg \exists y \forall x (P(x) \supset P(y))$
 - $\vdash \neg "Ningún invitado llegó sin regalo, luego todos los invitados llegaron con regalo" (\text{Simbolice antes}).$
6. Trate de determinar cual es el máximo número de pasos necesarios para poner una fórmula en forma prenexa.
7. Demuestre que toda fórmula es equivalente a una fórmula en la forma

$$Q_1 x_1 \dots Q_n x_n \left(\bigvee_{i=1}^n \left(\bigwedge_{j=1}^m \phi_j^i \right) \right)$$

en donde cada ϕ_j^i es una fórmula atómica o su negación.

- 8.* Sea $\mathcal{N} = \langle \mathbb{N}, <, +, \cdot, 0, 1 \rangle$, entonces la clausura universal de las siguientes equivalencias es verdadera en \mathcal{N} :

- (1) $x < y \Leftrightarrow \exists z (y = (z+1) + x)$
- (2) $x \neq y \Leftrightarrow (x < y \vee y < x)$
- (3) $(x=y \vee z=w) \Leftrightarrow (xz + yw = xw + yz)$
- (4) $(x=y \wedge z=w) \Leftrightarrow (x^2 + y^2 + z^2 + w^2 = xy + yx + zw + zw)$

- Explique por qué valen las equivalencias anteriores.
- Demuestre por inducción en fórmulas que para toda fórmula ϕ del lenguaje de \mathcal{N} existe un fórmula ϕ^* de la forma:

$$Q_1 x_1 \dots Q_n x_n (t_1(x_1, \dots, x_n) = t_2(x_1, \dots, x_n))$$

en donde t_1 y t_2 son términos en $+$ y en \cdot (polinomios), tal que $\mathcal{N} \models W(\phi \Leftrightarrow \phi^*)$.

- halle ϕ^* para la fórmula $\phi(x)$ que dice "x es primo".

9. Formule y demuestre resultados análogos a los del problema 8 para las estructuras $Z = \langle \mathbb{Z}, +, -, \cdot, 0, 1 \rangle$ y $\mathcal{R} = \langle \mathbb{R}, <, +, -, \cdot, 0, 1 \rangle$. Note que las equivalencias de 8 no valen necesariamente en estas estructuras.

Capítulo 4

Resolución en el cálculo de predicados

En este capítulo mostramos como reducir el problema de la validez en el Cálculo de Predicados, por medio del Teorema de Herbrand, al caso de fórmulas sin cuantificadores, y al Cálculo puramente proposicional. Tales ideas son la base teórica para los métodos de demostración automática de teoremas, algunos de los cuales han llegado a ser incorporados en los modernos lenguajes de programación (v.gr. PROLOG). Los resultados de que dependen se remontan, sin embargo a Skolem (1928) y Herbrand (1930), mucho antes del advenimiento de los computadores.

4.1 Forma Normal de Skolem

Consideremos una fórmula de la forma:

$$\forall x_1 \dots \forall x_n \exists y \phi(x_1, \dots, x_n, y) \quad (1)$$

Si esta fórmula es verdadera en alguna estructura $\mathfrak{N} = (A, \dots)$, es claro que para cualquier n-tupla de elementos $a_1, \dots, a_n \in A$ podemos escoger $b \in A$ tal que $\phi[a_1, \dots, a_n, b]$ es verdadera en \mathfrak{N} . Por supuesto, b depende de (a_1, \dots, a_n) .

Esto significa que hay una función de n -variables: $F : A^n \rightarrow A$ tal que para todo $a_1, \dots, a_n \in A$:

$$\mathfrak{N} \models \phi[a_1, \dots, a_n, F(a_1, \dots, a_n)]$$

F se llama una *función de Skolem*. * Añadiendo un *nuevo* símbolo de función f al lenguaje, y tomando F como interpretación de f , lo anterior se puede expresar:

$$(A, \dots, F) \models \forall x_1 \dots \forall x_n \phi(x_1, \dots, x_n, f(x_1, \dots, x_n)) \quad (2)$$

El símbolo de función f debe ser nuevo, porque los símbolos de función que ocurren en $\phi(x_1, \dots, x_n, y)$ tienen ya una interpretación asignada en \mathfrak{N} .

En el caso en que la lista de cuantificadores universales sea vacía, es decir si tenemos

$$\mathfrak{N} \models \exists y \phi(y) \quad (1')$$

podemos añadir un símbolo nuevo de constante c , y entonces para algún individuo $a \in A$ tenemos:

$$(A, \dots, a) \models \phi(c) \quad (2'')$$

A pesar de lo anterior, note que

$$\forall x_1 \dots \forall x_n \exists y \phi(x_1, \dots, x_n, y) \not\models \forall x_1 \dots \forall x_n \phi(x_1, \dots, x_n, f(x_1, \dots, x_n)).$$

Esto se debe a que la fórmula de la derecha es verdadera en \mathfrak{N} para la función específica $F : A^n \rightarrow A$ pero no para cualquier función $F' : A^n \rightarrow A$.

Sin embargo las dos fórmulas son equivalentes en un sentido más débil. Recuerde que una fórmula es *satisfactible* si tiene por lo menos un modelo.

Teorema 1 $\forall x_1 \dots \forall x_n \exists y \phi(x_1, \dots, x_n, y)$ es satisfactible si y solamente si $\forall x_1 \dots \forall x_n \phi(x_1, \dots, x_n, f(x_1, \dots, x_n))$ es satisfactible

* Por el matemático noruego T. Skolem.

Demstración. " \Rightarrow " es lo que acabamos de observar anteriormente. " \Rightarrow izq"

Se puede demostrar fácilmente que

$$\forall x_1 \dots \forall x_n \phi(x_1, \dots, x_n, f(x_1, \dots, x_n)) \vdash \neg \forall x_1 \dots \forall x_n \exists y \phi(x_1, \dots, x_n, y)$$

(ejercicio), por lo tanto, todo modelo de la fórmula de la izquierda es modelo de la derecha. ■

Note que lo mismo sucede con las fórmulas $\exists y \phi(y)$ y $\phi(c)$. Aplicando sistemáticamente el resultado del teorema podemos eliminar todos los cuantificadores existenciales de una fórmula en forma prenexa y producir una fórmula *equivalente para satisfactibilidad*, éste se llama una *forma normal de Skolem*.

Ejemplo 1

Si tenemos:

$$\exists x \forall y \forall z \exists w \forall u \exists t \exists s \forall r \phi(x, y, z, w, u, t, s, r)$$

podemos transformarla sucesivamente en:

$$\forall y \forall z \exists w \forall u \exists t \exists s \forall r \phi(c, y, z, w, u, t, s, r)$$

$$\forall y \forall z \forall u \exists t \exists s \forall r \phi(c, y, z, f(y, z), u, t, s, r)$$

$$\forall y \forall z \forall u \exists s \forall r \phi(c, y, z, f(y, z), u, g(y, z, u), s, r)$$

$$\forall y \forall z \forall u \forall r \phi(c, y, z, f(y, z), u, g(y, z, u), h(y, z, u), r)$$

La última fórmula es una forma normal de skolem de la primera, y es satisfactible si y solamente si la primera lo es.

Ejemplo 2

$$\exists x \forall y R(x, y) \supset \forall w \exists y R(w, y) \text{ eq } \forall x \exists y \forall w \exists z [(R(x, y) \supset R(w, z))]$$

Su forma normal de Skolem sería:

$$\forall x \forall w [R(x, f(x)) \supset R(w, g(x, w))]$$

Ejercicios

1. Termine la demostración del teorema 1.
2. Demuestre que si f es un símbolo de función que no ocurre en $\phi(x_1, \dots, x_n, y)$ entonces:

$$\models \exists x_1, \dots, \exists x_n \forall y \phi(x_1, \dots, x_n, y)$$

si y solamente si

$$\models \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n, f(x_1, \dots, x_n))$$

3. Halle la forma normal de Skolem de :

 - a. $\forall x [P(x) \supset \exists z (\neg \forall y (Q(x, y) \supset P(f(z))) \wedge \forall y (Q(x, y) \supset P(x)))]$
 - b. $(\forall x \exists y R(x, y) \wedge \forall y \exists z Q(y, z)) \supset \forall x \exists y \exists z (P(x, y) \supset Q(y, z))$
 - c. $\forall x ((P(x) \wedge \forall y (R(x, y) \supset \exists z (f(x, z) = y))) \supset \forall y R(x, y))$.

4. Sea ϕ^S la forma normal de Skolem de una fórmula ϕ , y sea ψ una fórmula en donde no ocurren los símbolos de función de Skolem de ϕ^S . Demuestre que :

$$\phi \vdash \phi \Leftrightarrow \phi^S \vdash \psi$$

5. Sea $\phi_1, \phi_2, \dots, \phi_n$ ϕ fórmulas. Muestre que $\phi_1, \dots, \phi_n \vdash \phi$ si y solamente si $\phi_1^S, \phi_2^S, \dots, \phi_n^S, (\neg \phi)^S$ es insatisfacible.
- 6.*** Del ejercicio 2 se desprende, usando el Teorema de Completitud de Gödel que

$$\vdash \exists x \forall y \phi(x, y) \Leftrightarrow \vdash \exists x \phi(x, f(x))$$

donde f es un símbolo de función que no ocurre en ϕ . Demuestre lo anterior sin usar el Teorema de Completitud, es decir de una manera puramente sintáctica.

4.2 Teorema de Herbrand

Damos ahora un teorema de gran interés teórico y que nos permitirá extender al Cálculo de Predicados el método de Resolución basado en las reglas de *corte y simplificación*, desarrollado para el Cálculo de Proposiciones (Parte I, Cap. 5). Considere una fórmula *puramente existencial*, es decir de la forma

$$\exists x_1 \dots \exists x_n \phi(x_1, \dots, x_n) \quad (1)$$

donde $\phi(x_1, \dots, x_n)$ es *puramente proposicional* (no tiene cuantificadores). Llamamos *universo de Herbrand* de la fórmula (1) al conjunto de los términos *cerrados* (sin variables) que se pueden formar con los símbolos de función y constante que ocurren en $\phi(x_1, \dots, x_n)$.

Ejemplo

Dada

$$\exists x \exists y \exists z (R(x, f(y, a)) \wedge h(f(x, a)) = z \supset (x = b \vee h(y) = z))$$

su universo de Herbrand sería:

$$\{a, b, h(z), h(b), f(a, a), f(a, b), \dots, f(a, h(a)), \dots, f(f(a, b), a), \dots\}$$

Si la fórmula no tiene constantes, se añade una al lenguaje para formar el universo de Herbrand.

Ejemplo

Dada

$$\exists x \exists y (g(g(x)) = y \supset R(g(x), y))$$

tenemos, añadiendo una constante c:

$$\{c, g(c), g(g(c)), \dots\}$$

Consideremos ahora términos:

$$t_1^1, \dots, t_n^1; t_1^2, \dots, t_n^2; t_1^k, \dots, t_n^k$$

entonces es fácil ver que

$$\phi(t_1^1, \dots, t_n^1) \vee \phi(t_1^2, \dots, t_n^2) \vee \dots \vee \phi(t_1^k, \dots, t_n^k) \vdash_{\text{Prop}} \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n)$$

aplicando las reglas del Cálculo de Proposiciones y GÁ (ejercicio). El converso no es necesariamente cierto (¿por qué?). Sin embargo, tenemos el siguiente teorema donde \vdash_{Prop} significa deducibilidad usando *sólo* los axiomas y reglas del Cálculo de Proposiciones.

Teorema 2 (Herbrand) *Sea $\phi(x_1, \dots, x_n)$ una fórmula sin cuantificadores que no contiene el símbolo de igualdad, entonces*

$$\vdash_{\text{Prop}} \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n) \quad (2)$$

si y solamente si existen términos del universo de Herbrand de la fórmula:

$$t_1^1, \dots, t_n^1; t_1^2, \dots, t_n^2; t_1^k, \dots, t_n^k \text{ (para algún } k \in \mathbb{N}^+ \text{)}$$

tales que:

$$\vdash_{\text{Prop}} \phi(t_1^1, \dots, t_n^1) \vee \phi(t_1^2, \dots, t_n^2) \vee \dots \vee \phi(t_1^k, \dots, t_n^k) \quad | \quad (3)$$

Demostración. " \Rightarrow izq" Obvio, pues la fórmula (2) se deduce de la fórmula (3) en el Cálculo de Predicados, como se observó anteriormente. " \Rightarrow " Esta dirección es mucho más complicada pues requiere utilizar el "teorema de compacidad" del Cálculo de Proposiciones (Parte I, Cap. 6). Se supone que no vale (3) y se muestra que no vale (2). Dejamos la prueba para la sección 4.4 ■

El siguiente corolario nos dará nuestro método deductivo, que es la base de la llamada demostración automática de teoremas.

Colorario 3 *Una fórmula en forma normal de Skolem que no contiene el símbolo =,*

$$\forall x_1, \dots, \forall x_n \phi(x_1, \dots, x_n) \quad (4)$$

es inconsistente si y solamente si existen términos del universo de Herbrand

$$t_1^1, \dots, t_n^1 \vee t_1^2, \dots, t_n^2 \vee \dots \vee t_1^k, \dots, t_n^k$$

tales que

$$\{\phi(t_1^1, \dots, t_n^1) \vee \phi(t_1^2, \dots, t_n^2) \vee \dots \vee \phi(t_1^k, \dots, t_n^k)\}$$

es inconsistente en el Cálculo de Proposiciones.

Demostración. (4) es inconsistente si y solamente si

$$\vdash \neg \forall x_1, \dots, \forall x_n \phi(x_1, \dots, x_n)$$

$$\Leftrightarrow \vdash \exists x_1, \dots, \exists x_n \neg \phi(x_1, \dots, x_n)$$

\Leftrightarrow Existen t_1^i, \dots, t_n^i , $i=1, \dots, k$, tales que

$$\Leftrightarrow \vdash_{\text{prop}} \bigvee_{i=1}^k \neg \phi(t_1^i, \dots, t_n^i) \text{ (por Teorema de Herbrand)}$$

$$\Leftrightarrow \vdash \neg (\bigvee_{i=1}^k \phi(t_1^i, \dots, t_n^i))$$

$\Leftrightarrow \{\phi(t_1^1, \dots, t_n^1), \dots, \phi(t_1^k, \dots, t_n^k)\}$ inconsistente en Cálculo de

Proposiciones. ■

Suponga ahora que estamos interesados en demostrar que cierta sentencia σ es válida. como σ es válida si y solamente si $\neg\sigma$ es insatisfacible, podemos tomar la forma normal de Skolem de $\neg\sigma$, digamos:

$$\forall x_1, \dots, \forall x_n \phi(x_1, \dots, x_n) \tag{5}$$

y tenemos por el teorema 1 que σ es válida si y solamente si (5) es insatisfacible. Por el teorema de Completitud de Gödel esto equivale a decir que (5) es inconsistente, y por el corolario 3, para probarlo basta encontrar

términos t_1^i, \dots, t_n^i , $i = 1, \dots, k$ tales que $\{\phi(t_1^i, \dots, t_n^i) \mid i = 1, \dots, k\}$ sea inconsistente en el Cálculo de Proposiciones. Como

$$\phi(x_1, \dots, x_n) \text{ eq. } \{D_1(x_1, \dots, x_n), \dots, D_s(x_1, \dots, x_n)\} \text{ donde los } D_i(x_1, \dots, x_n)\}$$

son disyunciones elementales de fórmulas atómicas y sus negaciones, podemos entonces pensar en el siguiente método: -genere *todas* las n-tuplas del universo de Herbrand de

$$\phi(x_1, \dots, x_n): (t_1^1, \dots, t_n^1), (t_1^2, \dots, t_n^2) \dots$$

-aplique sucesivamente las reglas de *corte* y *simplificación* a los conjuntos:

$$1- D_1(t_1^1, \dots, t_n^1), \dots, D_s(t_1^1, \dots, t_n^1)$$

$$2- \boxed{D_1(t_1^1, \dots, t_n^1), \dots, D_s(t_1^1, \dots, t_n^1)} \\ \quad \quad \quad \boxed{D_1(t_1^2, \dots, t_n^2), \dots, D_s(t_1^2, \dots, t_n^2)}$$

$$3- \boxed{D_1(t_1^1, \dots, t_n^1), \dots, D_s(t_1^1, \dots, t_n^1)} \\ \quad \quad \quad \boxed{D_1(t_1^2, \dots, t_n^2), \dots, D_s(t_1^2, \dots, t_n^2)} \\ \quad \quad \quad \boxed{D_1(t_1^3, \dots, t_n^3), \dots, D_s(t_1^3, \dots, t_n^3)}$$

etc. Eventualmente (si (5) es inconsistente) se encontrará algún i:

$$\boxed{D_1(t_1^1, \dots, t_n^1), \dots, D_s(t_1^1, \dots, t_n^1)} \\ \boxed{D_1(t_1^2, \dots, t_n^2), \dots, D_s(t_1^2, \dots, t_n^2)} \\ \quad \quad \quad \vdots \\ \boxed{D_1(t_1^i, \dots, t_n^i), \dots, D_s(t_1^i, \dots, t_n^i)} \quad \mid_{\overline{R}} f$$

Ejemplo 1

Supóngase que queremos mostrar

$$\mid \neg \exists x \forall y R(x,y) \supset \forall y \exists x R(x,y)$$

Basta mostrar que su negación

$$\neg(\exists x \forall y R(x,y) \supset \forall y \exists x R(x,y))$$

es inconsistente. Una forma prenexa de la negación es:

$$\exists x \forall y \exists z \forall w [R(x,y) \supset R(w,z)]$$

Su forma normal de Skolem:

$$\forall y \forall w [R(c,y) \wedge \neg R(w,f(y))]$$

El universo de Herbrand:

$$\{c, f(c), f(f(c)), \dots\}$$

Los pares de términos:

$$(c,c), (c,f(c)), (f(c),c), (f(c),f(c)), \dots$$

Substituyendo sucesivamente las variables (y,w) por estos pares en las disyunciones elementales de la matriz de la fórmula tenemos:

- | | |
|--|-------------------|
| 1. $R(c,c), \neg R(c,f(c))$
2. $R(c,c), \neg R(f(c),f(c))$
3. $R(c,f(c)), \neg R(c,f(f(c)))$ | $\mid_{\neg R} f$ |
|--|-------------------|

Se logra una contradicción con las tres primeras sustituciones.

Ejercicios

1. Demuestre

$$\bigvee_{i=1}^n \phi(t_1^i, \dots, t_n^i) \vdash \exists x_1 \dots \exists x_n \phi(x_1, \dots, x_n)$$

2. Sea $\phi(x_1, \dots, x_n)$ una fórmula que no contiene el símbolo $=$ ni símbolos de función o constante. Demuestre utilizando el teorema de Herbrand, que las siguientes afirmaciones son equivalentes:

- (i) $\exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n)$
- (ii) $\vdash \neg \phi(x, x, \dots, x)$
- (iii) $\phi(x, x, \dots, x)$ tiene la forma de una tautología.

3. Sean $\phi(x_1, \dots, x_n)$ y $\psi(x_1, \dots, x_n)$ fórmulas sin cuantificadores. Muestre que $\forall x_1, \dots, \forall x_n \phi(x_1, \dots, x_n) \vdash \exists x_1, \dots, \exists x_n \psi(x_1, \dots, x_n)$ si y solamente si existen términos t_1^i, \dots, t_n^i , $i=1, \dots, k$, tales que

$$\bigvee_{i=1}^n \phi(t_1^i, \dots, t_n^i) \vdash_{\text{Prop}} \bigvee_{i=1}^n \psi(t_1^i, \dots, t_n^i).$$

4.3 Método de Resolución

Es claro que al hacer sustituciones como en el último ejemplo de la sección anterior se generan muchas fórmulas irrelevantes que no contribuyen a la contradicción. Un método más eficiente consiste en substituir en cada una de las disyunciones $R(c, y)$, $\neg R(w, f(y))$ de manera que estas se unifiquen:

En $R(c, y)$, $y \rightarrow f(c)$ dá $R(c, f(c))$

En $\neg R(w, f(y))$, $w \rightarrow c$, $y \rightarrow c$, dá $\neg R(c, f(c))$

y se obtiene directamente la contradicción.

El siguiente sistema, llamado *método de resolución*, recoge esta idea. A las reglas de *corte* y *simplificación* se añade la siguiente regla. Llamaremos *literales* de una disyunción elemental $D(x_1, \dots, x_n)$ a las fórmulas atómicas que aparecen en $D(x_1, \dots, x_n)$ afirmadas o negadas.

Unificación. Sean $D(x_1, \dots, x_n)$ y $D'(x_1, \dots, x_n)$ disyunciones elementales y $t_1, \dots, t_n, r_1, \dots, r_n$ términos (que pueden tener variables libres) tales que $D(t_1, \dots, t_n)$ y $D'(r_1, \dots, r_n)$ contienen un literal común, afirmado en una fórmula y negado en la otra, entonces

$$\frac{D(x_1, \dots, x_n), D'(x_1, \dots, x_n)}{D(t_1, \dots, t_n), D'(r_1, \dots, r_n)}$$

Ejemplo 2

Queremos demostrar

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

Una forma prenexa de su negación es:

$$\forall x \exists y \exists z \forall w \forall u ((P(x) \vee R(x, y)) \wedge \neg R(z, w) \wedge \neg P(u))$$

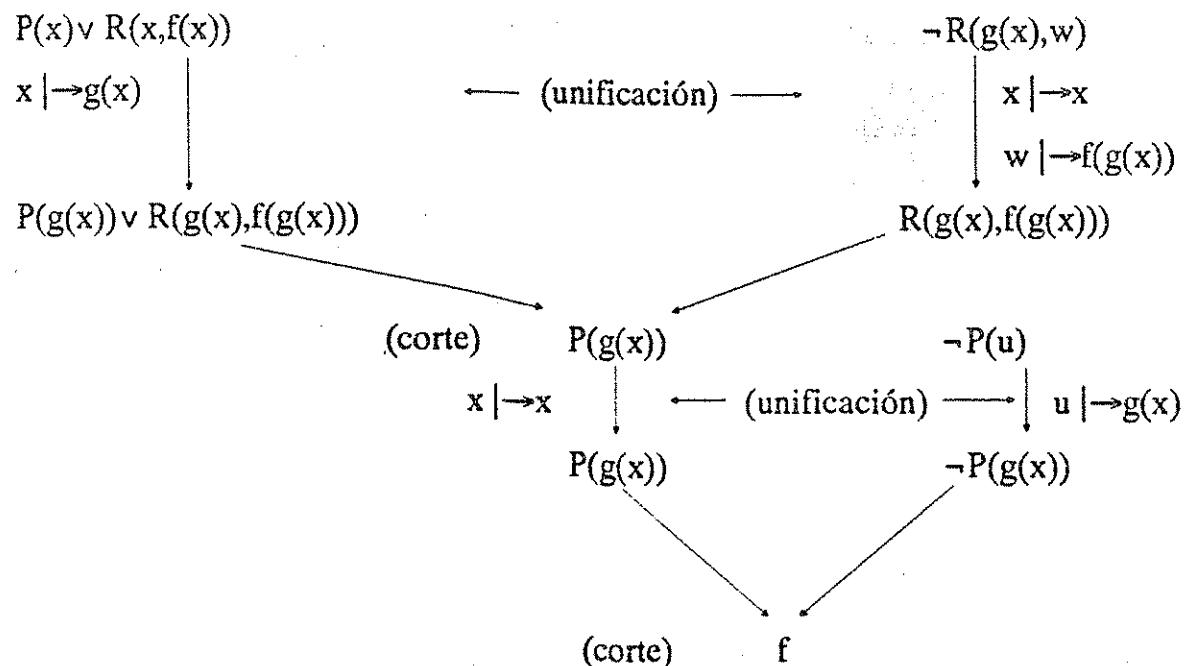
y su forma normal de Skolem:

$$\forall x \forall w \forall u ((P(x) \vee R(x, f(x))) \wedge \neg R(g(x), w) \wedge \neg P(u))$$

Disyunciones fundamentales:

$$P(x) \vee R(x, f(x)) \quad R(g(x), w) \quad P(u)$$

Tenemos:



Note que al unificar sustituimos términos con variables libres para dejar abierta la puerta a futuras substituciones. El método admite muchos refinamientos que no discutiremos aquí.

Ejercicios

1. Aplique el método de resolución para demostrar la inconsistencia de las siguientes fórmulas:
 - $[\forall x \exists y R(x, y) \wedge \forall y \exists z Q(y, z)] \supset \forall x, \exists y, \exists z (R(x, y), y Q(y, z))$
 - $\neg \exists y \forall z [R(z, y) \leftrightarrow \neg \exists x (R(z, x) \wedge R(x, z))]$
 - $\forall z \forall x \{ [\neg R(z, c) \vee \neg R(z, x) \vee \neg R(x, z)] \wedge [R(x, f(z)) \vee R(z, c)] \wedge [R(f(z), z) \vee R(z, a)] \}$
 - $\forall x \forall y \{ [R(x, y) \wedge (\neg R(y, f(x, y)) \vee \neg R(f(x, y), f(x, y)))] \vee [R(x, y) \wedge Q(x, y) \wedge (\neg Q(x, f(x, y)) \vee \neg Q(f(x, y), f(x, y)))] \}$

$$e. \forall x, \forall y \{ R(x,y) \wedge [\neg R(y,f(x,y)) \vee \neg R(f(y),f(x,y)) \vee \neg R((f,y),f(x,y)) \\ Q(x,y)] \wedge [\neg R(y,f(x,y)) \vee \neg R(f(x,y),f(x,y)) \vee \neg Q(x,f(x,y))] \}$$

2. Use resolución para demostrar:

$$\neg \exists x \forall y R(x,y), \forall x \forall y (R(x,y) \vee R(y,x)) \vdash \forall x \exists y R(x,y)$$

(se puede tomar la forma normal de Skolem para cada fórmula por separado).

4.4 Demostración del Teorema de Herbrand

Sea L un léxico enumerable de primer orden. Si no tiene constantes, añadimos la constante adicional c .

Llamamos IG al conjunto de los axiomas de la igualdad especificados en los términos cerrados del lenguaje, es decir el conjunto de fórmulas sin cuantificadores:

$$\begin{aligned} t &= t \\ t = t' &\supset (P(\dots t \dots) \supset P(\dots t' \dots)) \\ t = t' &\supset (f(\dots t \dots) = f(\dots t' \dots)) \end{aligned}$$

donde t y t' recorren todos los términos cerrados del lenguaje, P recorre los símbolos de predicado y f los de función.

Usaremos \vdash para denotar deducción en el Cálculo de predicados y \vdash_{subP} para la deducción en el Cálculo de Proposiciones.

Lema 1 De IG se puede deducir en el Cálculo de Proposiciones:

- (a) $t = t; t = t' \supset t' = t; t = t' \wedge t' = t'' \supset t = t''$
- (b) $t_1 = t'_1 \wedge \dots \wedge t_n = t'_n \supset (P(t_1, \dots, t_n) \Leftrightarrow P(t'_1, \dots, t'_n))$
- (c) $t_1 = t'_1 \wedge \dots \wedge t_n = t'_n \supset (f(t_1, \dots, t_n) = f(t'_1, \dots, t'_n))$

Demostración. Se deja al lector. ■

Sea $\mathbb{P}(L)$ el lenguaje proposicional construido con las fórmulas atómicas:

$$R(t_1, \dots, t_n) \quad t = t'$$

donde t_1, \dots, t_n, t y t' recorren los términos cerrados y R recorre los símbolos de predicado de L .

Dada una valuación $v: pos(L) \rightarrow \{V, F\}$ tal que $\bar{v}(IG) = V$ definimos una relación en el conjunto T de los términos así:

$$t \sim_{\bar{v}} t' \Leftrightarrow V(t = t') = V$$

Es fácil ver que $\sim_{\bar{v}}$ es una relación de equivalencia en T , gracias a que $\bar{v}(IG) = V$ y al Lema 1(a).

Sea $[t]$ la clase de equivalencia de t . Definimos ahora la estructura:

$$\mathfrak{N}_v = \{[t] \mid t \in \}, R^{\mathfrak{N}}, \dots, f^{\mathfrak{N}}, \dots, c^{\mathfrak{N}} \dots\}$$

donde

$$([t_1], \dots, [t_n]) \in R^{\mathfrak{N}} \Leftrightarrow \bar{v}(R(t_1, \dots, t_n)) = V$$

$$f^{\mathfrak{N}}([t_1], \dots, [t_n]) = [f(t_1, \dots, t_n)], c^{\mathfrak{N}} = [c]$$

Lema 2 \mathfrak{N}_v está bien definida y para toda fórmula proposicional ϕ de $\mathbb{P}(L)$ se tiene:

$$\mathfrak{N}_v \models \bar{v}(\phi(t_1, \dots, t_n)) = V$$

Demostración. La buena definición de \mathfrak{N}_v resulta del lema 1 (b) y (c). La equivalencia es cierta para atómicas por definición y la inducción en los conectivos proposicionales funciona trivialmente. ■

Colorario 3 Sea ϕ una sentencia sin cuantificadores entonces

$$\vdash \phi \Leftrightarrow IG \dashv \phi$$

Demostración. La implicación de derecha a izquierda es trivial. Suponga ahora que $\models \phi$ y $IG \not\models_{\text{Prop}} \phi$, entonces existe una valuación v de las fórmulas atómicas de $\mathbb{P}(L)$ tal que $\bar{v}(IG) = V$ y $\bar{v}(\phi) = F$. Por el lema anterior, $\mathfrak{N} \not\models \phi$ contradiciendo que $\models \phi$. ■

Teorema 4 (Teorema de Herbrand) *Sea $\phi(x_1, \dots, x_n)$ una fórmula sin cuantificadores tal que $\models \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n)$, entonces existen términos cerrados t_1^j, \dots, t_n^j , $j = 1, \dots, k$, tales que*

$$IG \models \bigvee_{i=1}^k v \phi(t_1^i, \dots, t_n^i)$$

Demostración. Sea $\vec{t}^1, \vec{t}^2, \dots$ una enumeración de todas las n -tuplas de términos cerrados del lenguaje, $\vec{t}^j = (t_1^j, \dots, t_n^j)$ y suponga que para toda k se tiene:

$$IG \models \bigvee_{i=1}^k v \phi(\vec{t}^i)$$

entonces

$$IG \cup \{\neg(\bigvee_{i=1}^k \phi(\vec{t}^i))\}$$

y por lo tanto el conjunto $IG \cup \{\neg \phi(\vec{t}^1), \dots, \neg \phi(\vec{t}^k)\}$ sería proposicionalmente consistente para todo k .

Por compactidad del Cálculo de Proposiciones $T = IG \cup \{\phi(\vec{t}^1), \neg \phi(\vec{t}^2), \dots\}$ sería proposicionalmente consistente. Sea V una valuación de las fórmulas atómicas de $\mathbb{P}(L)$ tal que $\bar{v}(T) = V$. Entonces $\bar{v}(\phi(\vec{t}^j)) = F$ y por el Lema 2

$$\mathfrak{N}_v \not\models \phi[t_1^j, \dots, t_n^j]$$

para todo $\vec{t}^j = (t_1^j, \dots, t_n^j)$. Por definición del universo de \mathfrak{N}_v tendríamos:

$$\mathfrak{N}_v \not\models \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n)$$

contradicciendo que $\models \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n)$. ■

Es fácil ver que en el Teorema anterior los términos t_i y los axiomas de IG se pueden tomar de manera que solo involucren los símbolos de función constante y predicado que aparecen en ϕ , más una constante que debe añadirse si ϕ no tiene constantes. Esta constante c puede substituirse por una variable especial x , distinta de todas las variables que aparecen en ϕ . Si (x_1, \dots, x_n) no contiene el símbolo $=$, los axiomas de IG no serán utilizados y obtenemos el Teorema de Herbrand en la forma en que lo enunciamos en 4.2.

Ejercicios

1. Generalice el Colorario a fórmulas con variables libres.
2. Generalice el Teorema de Herbrand a fórmulas $\phi(\vec{x}, \vec{y})$ tales que $\vdash \exists \vec{x} \phi(\vec{x}, \vec{y})$.
3. Sea Γ un conjunto de sentencias universales, es decir de la forma $\forall z_1 \dots \forall z_k \theta$ donde θ no tiene cuantificadores. Demuestre que si $\phi(x_1, \dots, x_n)$ no tiene cuantificadores y $\Gamma \vdash \exists x_1, \dots, \exists x_n \phi(x_1, \dots, x_n)$, entonces existen términos $t_1^j, \dots, t_n^j, j=1 \dots, k$ tales que

$$\tilde{\Gamma} + \text{IG} \vdash \bigvee_{i=1}^k \phi(t_1^i, \dots, t_n^i)$$

donde $\tilde{\Gamma}$ resulta de especificar las fórmulas de Γ a los términos del lenguaje.

4. Sea $\{\forall \vec{x}_1 \phi_1(\vec{x}_1), \dots, \forall \vec{x}_n \phi_n(\vec{x}_n)\}$ un conjunto de sentencias universales, inconsistente en el Cálculo de Predicados. Demuestre que existen tuplas de términos $\vec{t}_1^j, \dots, \vec{t}_n^j, j=1, \dots, k$ tales que el conjunto IG unido a

$$\{\phi_1(\vec{t}_1^1), \dots, \phi_1(\vec{t}_1^k), \dots, \phi_n(\vec{t}_n^1), \dots, \phi_n(\vec{t}_n^k)\}$$

es inconsistente en el Cálculo de Proposiciones.

Tercera parte

Calculabilidad



Enumerabilidad efectiva, decidibilidad

En el estudio del Cálculo de Proposiciones y el de Predicados hemos destacado sus aspectos algorítmicos. Por ejemplo, tenemos algoritmos para decidir si una cadena de símbolos es una fórmula bien formada, o una tautología, o un axioma del Cálculo de Predicados. La deducción formal nos proporciona otra gran cantidad de algoritmos. En este capítulo iniciamos el estudio de esos procesos "calculables", en un contexto más general.

Al comienzo trabajaremos basándonos en la noción intuitiva de *algoritmo*. Un algoritmo es un método cuya ejecución consiste en la aplicación, paso a paso, de ciertas reglas de transformación de símbolos especificados a priori, las cuales deben determinar el resultado final completamente. Los métodos clásicos para sumar, multiplicar, dividir, sacar raíz cuadrada en notación decimal, son algoritmos: a partir de las expresiones numéricas dadas, sigue un proceso en el cual repetitivamente se marcan, tachan, borran o escriben nuevos símbolos. Un programa de computador es el prototipo por excelencia de un algoritmo. La palabra algoritmo viene del nombre del matemático árabe Al-Kwarizmi (siglo IX) en cuyos escritos aparecen muchos de los métodos que aún usamos hoy día en álgebra y aritmética (el mismo origen de la palabra "guarismo" que usamos como sinónimo de "cifra").

Podemos concebir algoritmos que manipulen todo tipo de estructuras finitas, por ejemplo árboles, matrices, grafos, etc. Sin embargo, no se pierde

generalidad si nos restringimos a algoritmos que trabajen sobre sucesiones de símbolos en un alfabeto dado, ya que las estructuras finitas se pueden "linealizar" por medio de códigos adecuados, por ejemplo las fbf del Cálculo de Proposiciones se podrían considerar como códigos de ciertos árboles. Si Σ es un conjunto de símbolos, recuérdese que Σ^* denota el conjunto de todas las cadenas finitas de símbolos de Σ , incluyendo la cadena vacía Λ :

$$\Sigma^* = \{ \Lambda \} \cup \{ s_1 \dots s_n \mid s_i \in \Sigma, n \in \mathbb{N} \}$$

A los elementos de Σ^* se les llama también *palabras* sobre Σ .

En la sección siguiente repasamos las nociones clásicas, debidas a Cantor (1845 - 1918), de enumerabilidad y no enumerabilidad efectiva y decibilidad.

1.1 Conjuntos enumerables y no enumerables

Un conjunto A es *enumerable* si $A = \emptyset$ o existe una función sobreyectiva $f: \mathbb{N} \rightarrow A$. En tal caso se dice que f es una *enumeración* de A , y se puede visualizar como una sucesión infinita:

$$f(0), f(1), f(2), \dots$$

en la cual aparecen todos los elementos de A , con posibles repeticiones, y sólo elementos de A .

Ejemplo 1

El conjunto de los números enteros \mathbb{Z} es enumerable, como lo indica la enumeración siguiente:

$$0 \ -1 \ 1 \ -2 \ 2 \ -3 \ \dots$$

dada por la función

$$f(n) = \begin{cases} \frac{n}{2} & \text{si } n \text{ es par} \\ \frac{-(n+1)}{2} & \text{si } n \text{ es impar} \end{cases}$$

Ejemplo 2

El conjunto de los números racionales positivos $Q^* = \{n/m \mid n, m \in \mathbb{N}, m \neq 0\}$ es enumerable, como lo muestra la siguiente sucesión finita:

$$\frac{1}{1} \frac{1}{2} \frac{2}{1} \frac{1}{3} \frac{2}{2} \frac{3}{1} \frac{1}{4} \frac{2}{3} \frac{3}{2} \frac{4}{1} \frac{1}{5} \frac{2}{4} \frac{3}{3} \frac{4}{2} \frac{5}{1} \dots$$

donde hemos enumerado los fracciones según lo que sumen el numerador y el denominador, primero los de suma 2, luego los de suma 3, los de suma 4, etc. Aunque no podamos dar una "fórmula" para la función, ésta queda bien definida: $f(0) = 1/1$, $f(1) = 1/2$, $f(2) = 2/1$, $f(3) = 1/3$, $f(4) = 2/2$, etc. El conjunto de todos los racionales $Q = \{n/m \mid n \in \mathbb{Z}, m \neq 0\}$ también resulta enumerable:

$$0 \frac{1}{1} - \frac{1}{1} \frac{1}{2} - \frac{1}{2} \frac{2}{1} - \frac{2}{1} \frac{1}{3} - \frac{1}{3} \frac{2}{2} - \frac{2}{2} \frac{3}{1} - \frac{3}{1} \frac{1}{4} - \frac{1}{4} \frac{2}{3} - \frac{2}{3} \dots$$

Ejemplo 3

Todo conjunto finito es enumerable. Si $A = \{a_1, \dots, a_M\}$ basta definir $f: \mathbb{N} \rightarrow A$ por:

$$f(n) = \begin{cases} a_n & \text{si } n < 1 \\ a_M & \text{si } n \geq M \end{cases}$$

Ejemplo 4

$\{a\}^* = \{\Lambda, a, aa, aaa, \dots\}$ es enumerable.

En general, si $\Sigma = \{a_1, a_2, \dots, a_M\}$ es un conjunto finito de símbolos entonces Σ^* es enumerable; ya que podemos listar las palabras de acuerdo a su longitud, primero la palabra vacía, luego las de longitud 1 (hay M), luego las de longitud 2 (hay M^2), entonces las de longitud 3 (hay M^3), etc.

$$\Lambda, a_1, a_2, \dots, a_M, a_1a_1, a_1a_2, a_1a_M, a_2a_1, a_2a_2, \dots, a_2a_M, \dots, a_Ma_1, a_Ma_2, \dots, a_Ma_M, a_1a_1a_2, \dots$$

El siguiente teorema nos proporciona una gran cantidad de ejemplos enumerables:

Teorema 1. *Si A es enumerable y $B \subseteq A$, entonces B es enumerable.*

Demostración. Si $B = \emptyset$ no hay nada que demostrar. Suponga $B \neq \emptyset$ y escoja $b_0 \in B$. Sea $f: \mathbb{N} \rightarrow A$ una enumeración de A , entonces definimos $g: \mathbb{N} \rightarrow B$ por

$$g(n) = \begin{cases} f(n) & \text{si } f(n) \in B \\ b_0 & \text{si } f(n) \notin B \end{cases}$$

evidentemente g es sobreyectiva. ■

Corolario. *Si A es enumerable y existe una función inyectiva $h: B \rightarrow A$ entonces B es enumerable.*

Demostración. Suponga $B \neq \emptyset$ y sea $B' = \{h(b) \mid b \in B\}$. Como $B' \subseteq A$ entonces es enumerable, además $h: B \rightarrow B'$ es una biyección y por lo tanto tiene inverso $h^{-1}: B' \rightarrow B$. Ahora, si $f: \mathbb{N} \rightarrow B'$ es una enumeración de B' . ■

Del teorema 1 se desprende que todo subconjunto de \mathbb{N} es enumerable, inclusive conjuntos para los cuales nadie sabe como dar "explícitamente" una enumeración, por ejemplo

$$\{n \in \mathbb{N} \mid \text{la ecuación } x^n + y^n = z^n \text{ tiene soluciones enteras}\}$$

Otra interesante aplicación es la siguiente:

Ejemplo 5.

El conjunto $F(p_1, \dots, p_n)$ de las fbf del Cálculo de Proposiciones que sólo contienen las letras p_1, \dots, p_n , es enumerable, pues es un subconjunto de Σ^* , que es enumerable por el ejemplo 4, donde $\Sigma = \{ \neg, \wedge, \vee, \supset, (,), p_1, \dots, p_n \}$.

El siguiente resultado nos proporciona otros métodos para construir conjuntos enumerables.

Teorema 2. *Sea A_1, A_2, A_3, \dots una familia enumerable de conjuntos enumerables, entonces $\bigcup_{n=1}^{\infty} A_n$ es enumerable.*

Demostración. Tome una enumeración para cada A_i , con ellas podemos formar una matriz infinita:

$$A_1: a_0^2 \ a_1^1 \ a_2^1 \ a_3^1 \dots$$

$$A_2: a_0^2 \ a_1^2 \ a_2^2 \ a_3^2 \dots$$

$$A_3: a_0^3 \ a_1^3 \ a_2^3 \ a_3^3 \dots$$

$$A_4: a_0^4 \ a_1^4 \ a_2^4 \ a_3^4 \dots$$

$$\vdots \quad \vdots \quad \vdots \quad \vdots$$

en donde aparecen todos los elementos de la unión, y la cual podemos convertir en una sola lista recorriéndola por el método diagonal de Cantor indicado en el diagrama:

$$a_0^1 \ a_0^2 \ a_1^1 \ a_0^3 \ a_1^2 \ a_2^1 \ a_0^4 \ \dots$$

Corolario. *Si A y B son enumerables entonces $A \cup B$ y $A \times B$ son enumerables.*

Demostración. $A \cup B$ se puede ver como la unión infinita de la sucesión enumerable de conjuntos: A, B, A, A, A, A, \dots Por el Teorema 2 debe ser

enumerable. Por otra parte se a_1, a_2, \dots es una enumeración de A, es claro que para cada n, $\{a_n\} \times B$ se puede poner en biyección con B y por lo tanto es enumerable, entonces

$$A \times B = \bigcup_{i=1}^{\infty} \{a_i\} \times B$$

resulta enumerable por el Teorema 2. ■

Ejemplo 6

Sea p_1, p_2, p_3, \dots una lista definida de letras proposicionales, entonces $F(p_1, p_2, \dots)$, el conjunto de las fbf del Cálculo de Proposiciones que se pueden formar con dichas letras es enumerable pues

$$F(p_1, p_2, p_3, \dots) = \bigcup_{i=1}^{\infty} F(p_1, \dots, p_n)$$

y por el ejemplo 5 cada $F(p_1, \dots, p_n)$ es enumerable.

Ejemplo 7

Por el corolario, los conjuntos $\mathbb{N} \times \mathbb{N}$, $\mathbb{N} \times \mathbb{N} \times \mathbb{N}$, etc. son enumerables.

Terminamos demostrando la no enumerabilidad del conjunto de subconjuntos de \mathbb{N} , *partes de \mathbb{N}* .

Teorema 3. *El conjunto $P(\mathbb{N}) = \{s \mid S \subseteq \mathbb{N}\}$ no es enumerable.*

Demostración. Suponga que $P(\mathbb{N})$ si es enumerable y sea

$$S_0, S_1, S_2, \dots$$

una enumeración de $P(\mathbb{N})$. Defina $A = \{n \mid n \notin S_n\}$ entonces $A \in P(\mathbb{N})$ y así $A = S_M$ para algún $M \in \mathbb{N}$. Ahora, si $M \in A$, tenemos por definición de A que $M \notin S_M$, es decir $M \neg \in A$, una contradicción. Pero si $M \notin A$

también llegamos a una contradicción, pues por definición de A, esto implica $M \in S_M = A$. En todo caso obtenemos una contradicción. ■

Corolario. Si Σ es un alfabeto finito no vacío entonces $p(\Sigma^*)$ es no enumerable.

Demostración. Como hay una biyección entre Σ^* y \mathbb{N} entonces se puede establecer una biyección entre $p(\mathbb{N})$ y $P(\Sigma^*)$. ■

Ejercicios

1. Demuestre que A es infinito enumerable si y solamente si existe una biyección $f: \mathbb{N} \rightarrow A$. (Esta es la definición usual de conjunto enumerable).
2. Demuestre la siguiente caracterización "dual" de los conjuntos enumerables: A es enumerable si y solamente si existe una función inyectiva $h: A \rightarrow \mathbb{N}$.
3. Demuestre que el conjunto de todas las matrices con coeficientes racionales es enumerable.
4. Demuestre que si Σ es un conjunto finito enumerable de símbolos entonces Σ^* es enumerable.
5. Demuestre que el conjunto $\mathbb{N}^{\mathbb{N}} = \{ f \mid f: \mathbb{N} \rightarrow \mathbb{N} \}$ no es enumerable.
6. Muestre que si A es finito, $P(A)$ no es enumerable.
7. ¿Son enumerables los siguientes conjuntos?
 - a. El conjunto de todas las sucesiones finitas de números racionales.
 - b. El conjunto de todas las sucesiones infinitas de números naturales.
 - c. El conjunto de los números reales.

- d. El conjunto de las fórmulas del Cálculo de predicados sobre el léxico $L = \{ R^2, P^1, f^2, g^3 \}$ que son deducibles de la premisa $\forall x (R(g(x, f(x, x), x), x) \supset P(x))$.
8. a. Muestre que el conjunto de los subconjuntos finitos de \mathbb{N} es enumerable.
- b. Muestre que el conjunto de los subconjuntos infinitos de \mathbb{N} no es enumerable.
- 9.* Dos conjuntos A y B se dicen *equipotentes*, escrito $A \sim B$ si existe una biyección entre A y B . Por ejemplo, A es infinito enumerable si y sólo si es equipotente a \mathbb{N} .
- Demuestre que \sim es una relación de equivalencia.
 - Demuestre que los conjuntos siguientes son todos equipotentes entre sí:

$$\mathbb{R}, (0,1), [0,1], [0,1], \mathbb{R} \times \mathbb{R}, P(\mathbb{N}), \mathbb{N}^\mathbb{N}$$

1.2 Conjuntos efectivamente enumerables

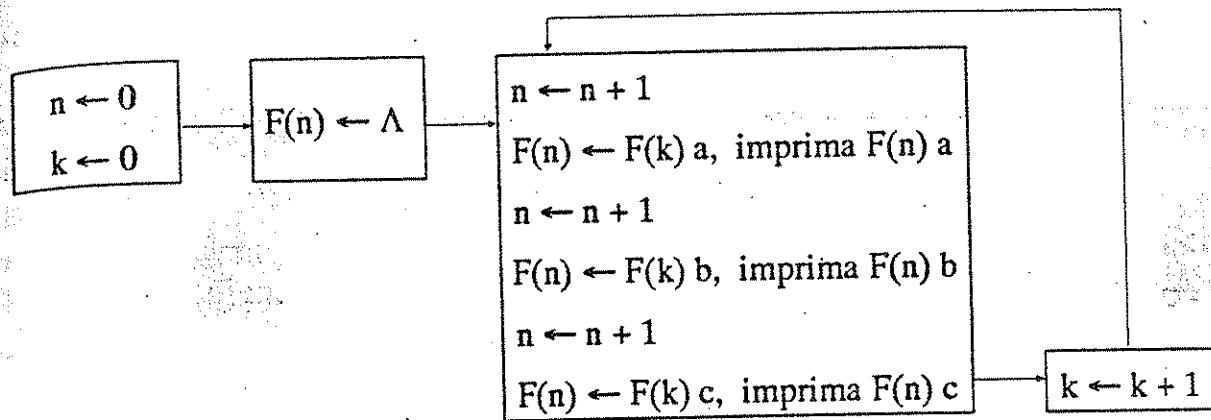
Sea Σ un alfabeto infinito y $A \subseteq \Sigma^*$. Se dice que A es *efectivamente* si existe un algoritmo que genera una sucesión de elementos de A en la cual aparece eventualmente todo elemento de A . En otras palabras existe una enumeración $f: \mathbb{N} \rightarrow A$ *calculable por medio de un algoritmo*.

Ejemplo 1

Σ^* mismo es efectivamente enumerable, pues la enumeración que dimos en la sección anterior (ejemplo 4):

Λ , palabras de longitud 1, palabras de longitud 2, ...

puede generarse fácilmente por un programa de computador. Si $\Sigma = \{a, b, c\}$, un diagrama de flujo para el algoritmo podría ser:



evidentemente, si se activa este algoritmo, imprime sucesivamente las palabras de Σ^* en orden lexicográfico.

Ejemplo 2

Si p, q, r son letras proposicionales, el conjunto de fórmulas bien formadas $F(p, q, r) \subseteq \{ p, q, r, \neg, \wedge, \vee, \supset, () \}^*$ es efectivamente enumerable, pues hay un algoritmo para enumerar primero las palabras de longitud 1: p, q, r luego las de longitud 4: $\neg(p), \neg(q), \neg(r)$, entonces las de longitud 7: $\neg(\neg(p)), \neg(\neg(q)), \neg(\neg(r)), (p) \wedge (p), (p) \wedge (q), (p) \wedge (r), (q) \wedge (p), \dots$, etc., cuyos detalles dejamos al lector.

Veamos ahora el caso de los conjuntos numéricos. Los números naturales pueden representarse de muchas maneras, por ejemplo, como palabras en $\{/ \}^*$ (notación monádica), o como palabras en $\{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \}^*$ (notación decimal). Un subconjunto de \mathbb{N} se dice efectivamente enumerable si lo es en representación monádica, o decimal, o en cualquiera de las representaciones usuales. Esta noción resulta independiente de la notación, pues hay algoritmos claros para pasar de una notación a otra. De esta manera \mathbb{N} mismo resulta efectivamente enumerable, trivialmente.

Utilizando la notación decimal podemos representar los números racionales como palabras sobre el alfabeto $\{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, -, / \}$ y es fácil ver que de esta manera los conjuntos \mathbb{Z} y \mathbb{Q} mismos resultan efectivamente enumerables, pues las enumeraciones indicadas en los ejemplos 1, 2 Sec. anterior, son generables algorítmicamente.

el conjunto de los números reales, \mathbb{R} , no puede ser efectivamente enumerable porque ni siquiera es enumerable.

Si hay una fórmula para enumerar un conjunto de números, éste será efectivamente enumerable, pues los algoritmos aritméticos que aprendemos desde niños (por ejemplo en notación decimal) nos permiten utilizar la fórmula como un programa para generar los números, por ejemplo el conjunto de cuadrados perfectos estará generado efectivamente por $f(n)=n^2$.

Sin embargo, la "fórmula" no es una condición necesaria para la enumerabilidad efectiva.

Ejemplo 3

El conjunto de los números primos es efectivamente enumerable. Podríamos definir la enumeración de la forma siguiente:

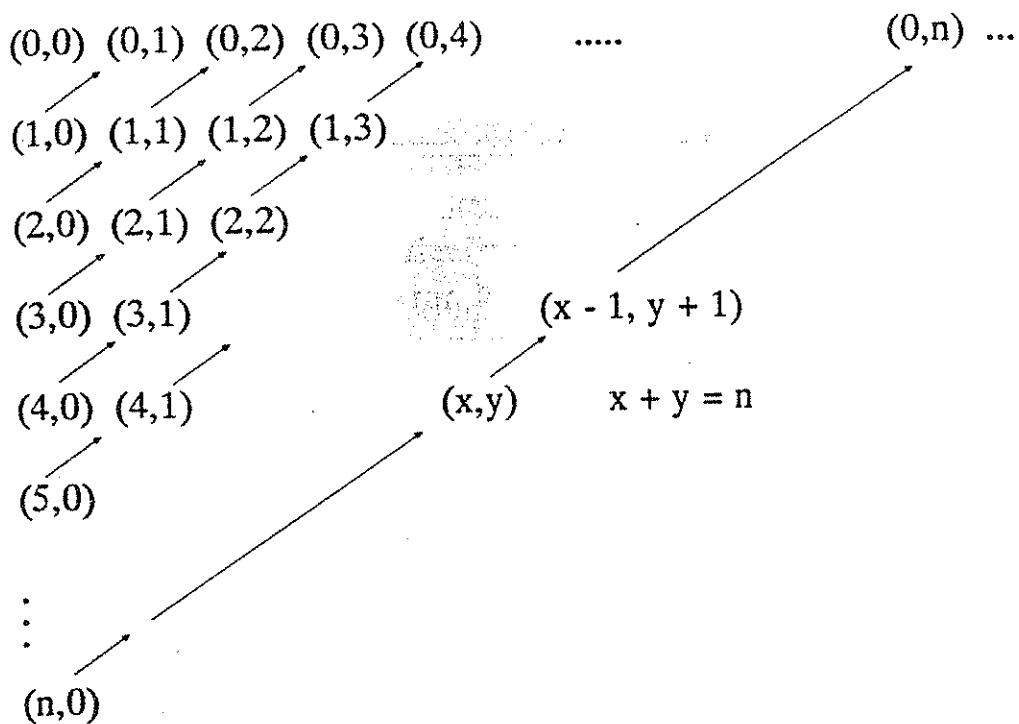
$$p(0) = 2$$

$$p(n + 1) = \min x \in \mathbb{N} (x > p(n), y p(0), p(1), \dots, p(n) \text{ no dividen } x)$$

Es claro que esta función se puede calcular paso a paso. Suponiendo que se han calculado $p(0), \dots, p(n)$, entonces $p(n + 1)$ se calcula de la manera siguiente: se toma $p(n) + 1$ y se divide sucesivamente por $p(0), \dots, p(n)$. Si resulta divisible por alguno de ellos se repite el proceso con $p(n) + 2, p(n) + 3$, etc., hasta hallar $p(n) + k$ que no sea divisible por ninguno, entonces se hace $p(n + 1) = p(n) + k$. Todos los procesos indicados se pueden realizar en un número finito de pasos, y el Teorema de Euclides sobre la infinitud de los números primos nos garantiza que encontraremos el número $p(n) + k$ buscado.

Ejemplo 4

$\mathbb{N} \times \mathbb{N}$ y todos sus subconjuntos pueden considerarse como palabras sobre el alfabeto $\{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, (,), '\}$. $\mathbb{N} \times \mathbb{N}$ es efectivamente enumerable, pues el método de Cantor de recorrerlo en diagonal es algorítmicamente calculable:



es decir $f(0) = (0,0)$, $f(1) = (1,0)$, $f(2) = (0,1)$, $f(3) = (2,0)$, $f(4) = (1,1)$, etc. Si $f(n) = (x_n, y_n)$ tenemos el siguiente algoritmo *recursivo* para calcular la pareja (x_n, y_n) :

$$f(0) = ((x_0, y_0)) = (0,0)$$

$$f(n+1) = (x_{n+1}, y_{n+1}) = \begin{cases} (x_n - 1, y_n + 1) & \text{si } x_n \geq 1 \\ (y_{n+1}, 0) & \text{si } x_n = 0 \end{cases}$$

Es decir, si (x_n, y_n) es la n -ésima pareja y $x_n \geq 1$, (x_{n+1}, y_{n+1}) se obtiene moviéndose diagonalmente arriba y a la derecha. Si $x_n = 0$, se ha llegado al extremo superior de la diagonal y (x_{n+1}, y_{n+1}) debe ser el extremo inferior de la siguiente diagonal.

¿Es posible dar fórmulas cerradas que den x_n y y_n directamente en función de n , $x_n = J(n)$, $y_n = K(n)$? Esta es una pregunta difícil que dejamos al lector ponderar.

Del ejemplo anterior tenemos el siguiente corolario:

Teorema 1. $\mathbb{N}^n = \mathbb{N} \times \mathbb{N} \times \dots \times \mathbb{N}$ (n veces) es efectivamente enumerable.

Demostración. Por inducción en n podemos construir funciones calculables $f_n: \mathbb{N} \rightarrow \mathbb{N}^n$ ($n \geq 2$) así: $f_2 = f: \mathbb{N} \rightarrow \mathbb{N} \times \mathbb{N}$. Suponiendo que hemos definido $f_k: \mathbb{N} \rightarrow \mathbb{N}^k$ biyección calculable, definimos $f_{k+1}: \mathbb{N} \rightarrow \mathbb{N}^{k+1} = \mathbb{N}^k \times \mathbb{N}$ por

$$n \mapsto f(n) = (x_n, y_n) \mapsto (f_k(x_n), y_n)$$

Esta función es *sobreyectiva*, pues si $(n_1, \dots, n_k, n_{k+1}) \in \mathbb{N}^{k+1}$ entonces $(n_1, \dots, n_k) = f_k(x)$ para algún $x \in \mathbb{N}$. Sea $n \in \mathbb{N}$ tal que $f(n) = (x, n_{k+1})$ entonces

$$f_{k+1}(n) = (f_k(x), n_{k+1}) = (n_1, \dots, n_k, n_{k+1})$$

Es *inyectiva* (ejercicio) y es obviamente *calculable* pues f y f_k lo son. ■

Ejemplo

El conjunto $\{(x, y, z) \in \mathbb{N}^3 \mid x^3 + y^4 = z^5\}$ es efectivamente enumerable. Para ver esto, utilizamos un algoritmo que genere una lista exhaustiva de la triples \mathbb{N}^3 :

$$(x_0, y_0, z_0), (x_1, y_1, z_1), (x_2, y_2, z_2), \dots$$

Ahora, cada vez que el algoritmo genere una tripla, digamos la n -ésima, (x_n, y_n, z_n) , calculamos $x_n^3 + y_n^4$ y z_n^5 ; si estos dos números resultan iguales ponemos la tripla en la lista de salida, de lo contrario, ponemos $(0, 0, 0)$ en la lista de salida, generamos la tripla siguiente $(x_{n+1}, y_{n+1}, z_{n+1})$, y repetimos el proceso. De esta manera generamos una lista (con repeticiones) del conjunto de soluciones de $x^3 + y^4 = z^5$:

$$(x_0^3, y_0^3, z_0^3), (x_1^3, y_1^3, z_1^3), (x_2^3, y_2^3, z_2^3), \dots$$

Si este conjunto es infinito (?) podemos modificar el algoritmo de manera que la lista no tenga repeticiones.

Ejercicios

1. Describa un algoritmo para generar el conjunto $F(p,q,r)$ (dé el diagrama de flujo).
2. ¿Son efectivamente enumerables los conjuntos siguientes?
 - a. El conjunto de las fórmulas del Cálculo de Predicados sobre el léxico $L = \{ R^2, P^1, f^2, g^3 \}$.
 - b. El conjunto de las matrices 3×3 con coeficientes en \mathbb{N} .
 - c. El conjunto de los grafos infinitos coloreables con cuatro colores.
 - d. $\{ n \mid x^n + y^n = z^n \text{ tiene solución no-trivial} \}$.
 - e. El conjunto de los primos de la forma $2^n + 1$.
 - f. $\{ (n, m) \mid n/m < \sqrt{2} \}$
 - g.* $\{ (n, m) \mid n/m < \pi \}$
 - h. $\{ (x, y, z) \mid \exists n (x^n + y^n = z^n) \}$
- 3.*** Determine si es efectivamente enumerable el conjunto
 $\{ n \mid x^n + y^n = z^n \text{ no tiene soluciones enteras} \}$
 (si el último teorema de Fermat es cierto, este conjunto es efectivamente enumerable ¿por qué?)
4. ¿Hay algún algoritmo más rápido que el del texto que permita calcular la enumeración creciente de los primos $p(0), p(1), \dots$?
5. Halle fórmulas algebraicas $g_3(x_1, x_2, x_3)$ y $g_4(x_1, x_2, x_3, x_4)$ que definan biyecciones $g_3: \mathbb{N}^3 \rightarrow \mathbb{N}$ y $g_4: \mathbb{N}^4 \rightarrow \mathbb{N}$.
6. Demuestre que la función f_{k+1} , definida en la demostración del Teorema 1 es inyectiva.
7. Demuestre que si A y B son efectivamente enumerables entonces $A \cup B$, $A \cap B$ y $A \times B$ son efectivamente enumerables.

8. Demuestre que si A es infinito y efectivamente enumerable entonces existe una *biyección* calculable $h: \mathbb{N} \rightarrow A$.
9. Demuestre que la función inversa $g = f^{-1}: \mathbb{N} \times \mathbb{N} \rightarrow \mathbb{N}$ de la enumeración f del Ejemplo 4 tiene una fórmula polinomial:

$$g(x, y) = \frac{(x+y)(x+y+1)}{2} + y$$

Ayuda: si (x, y) está en la n -ésima diagonal ($0 \leq n$) entonces $x + y = n$; además $g(n, 0) = 1 + 2 + \dots + n$.

- 10.** Demuestre que si $f(n) = (x_n, y_n)$ es la enumeración del Ejemplo 4 entonces:

$$x_n = \frac{\left(\frac{\sqrt{1+8n}-1}{2}\right)\left(\frac{\sqrt{1+8n}+5}{2}\right)}{2} - n$$

$$y_n = n - \frac{\left(\frac{\sqrt{1+8n}-1}{2}\right)\left(\frac{\sqrt{1+8n}+1}{2}\right)}{2}$$

donde $[a]$ denota la parte entera de a .

1.3 Existencia de conjuntos no efectivamente enumerables

No es cierto que si A es efectivamente enumerable y $B \subseteq A$ entonces B debe ser efectivamente enumerable. Para mostrar ésto es suficiente demostrar que hay subconjuntos de \mathbb{N} que no son efectivamente enumerables. Para ello necesitamos primero "contar" las funciones calculables.

Lema. *El conjunto $C = \{f: \mathbb{N} \rightarrow \mathbb{N} \mid f \text{ es calculable}\}$ es enumerable.*

Demostración. (Informal) Todo algoritmo debe ser descriptible por una lista de instrucciones en algún lenguaje por ejemplo español o Fortran. De esta

manera cada algoritmo se puede ver como una sucesión finita de símbolos de algún alfabeto finito. Si es el Español este será:

$$\Sigma = \{ a, b, c, \dots, x, y, z, ', *, :, ;, \square \} \quad (\square \text{ denota blanco})$$

Si es Fortran:

$$\Sigma = \{ 0, 1, \dots, 9, A, B, \dots, Z, +, *, -, /, (), \square \}$$

Esta sucesión finita será una palabra de Σ^* que da las "instrucciones" del algoritmo. Como Σ^* es enumerable entonces el conjunto

$$\text{Alg} = \{ \alpha \in \Sigma^* \mid \text{describir un algoritmo (programa)} \}$$

es enumerable. Ahora, para cada función calculable f podemos escoger una descripción $\alpha_f \in \Sigma^*$ de un algoritmo que calcule a f . La función $F: C \rightarrow \text{Alg}$ dada por $F(f) = \alpha_f$ es inyectiva pues dos funciones distintas no pueden ser calculadas por el mismo algoritmo, por lo tanto C es enumerable (corolario a Teorema 1, Sección 1.1).

Corolario. *El conjunto $E = \{ S \subseteq \mathbb{N} \mid S \text{ es efectivamente enumerable} \}$ es enumerable.*

Demostración. Para cada $S \in E$, escoja una función calculable f_S que enumere efectivamente a S . La función $H(S) = f_S$ de E en ?? es obviamente inyectiva y por lo tanto E es enumerable. ■

Teorema. *Existen subconjuntos de \mathbb{N} que no son efectivamente enumerables.*

Demostración. $E \subseteq P(\mathbb{N})$; si tuviéramos $E = P(\mathbb{N})$ entonces E sería no enumerable por el teorema de Cantor, contradiciendo el corolario anterior. Por lo tanto E es subconjunto propio de $P(\mathbb{N})$. Tome $S \in P(\mathbb{N})$ tal que $S \notin E$, entonces S no es efectivamente enumerable. ■

El teorema anterior es altamente no constructivo en el sentido de que asegura la existencia de un conjunto no efectivamente enumerable pero no nos dá la más mínima idea de como puede ser un tal conjunto, un ejemplo

explícito. El siguiente resultado nos dà una gran cantidad de ejemplos un poco mas explícitos:

Teorema. *Sea E_1, E_2, E_3, \dots cualquier enumeración exhaustiva de los subconjuntos efectivamente enumerables de \mathbb{N} (existe por el Corolario) entonces el conjunto $\{ n \in \mathbb{N} \mid n \notin E_n \}$ no es efectivamente enumerable.*

Demostración. Si este conjunto fuera efectivamente enumerable tendríamos $\{ n \in \mathbb{N} \mid n \notin E_n \} = E_i$ para algún i . Para tal i sería cierto entonces $i \in E_i$ si y solo si $i \notin E_i$, una contradicción. ■

Mencionamos sin demostración que el conjunto

$$\{ \phi \mid (\mathbb{N}, +, \cdot, 0, i) \models \phi \}$$

no es efectivamente enumerable. Esto es consecuencia del famoso Teorema de Incompletitud de Gödel.

Es fácil ver que si A y B son efectivamente enumerable entonces $A \cup B$ es efectivamente enumerable, pues si $f: \mathbb{N} \rightarrow A$ y $g: \mathbb{N} \rightarrow B$ son enumeraciones calculables, la función:

$$h(n) = \begin{cases} f\left(\frac{n}{2}\right) & \text{si } n \text{ es par} \\ g\left(\frac{n+1}{2}\right) & \text{si } n \text{ es impar} \end{cases}$$

es claramente calculable y enumera a $A \cup B$. Pero la anterior propiedad no puede generalizarse a familias infinitas, unión enumerable de conjuntos efectivamente enumerables no es necesariamente efectivamente enumerable.

Ejemplo

Sea S un subconjunto de \mathbb{N} que no sea efectivamente enumerable y sea:

$$n_1, n_2, n_3, \dots$$

una enumeración de dicho conjunto (que existe pero evidentemente no se puede dar por un algoritmo), y defina la familia de conjuntos $A_1 = \{ n_1 \}$,

$A_2 = \{ n_2 \}$, $A_3 = \{ n_3 \}$, ... Evidentemente cada A_i es efectivamente enumerable, pero $\bigcup_{i=1}^{\infty} A_i = S$ no lo es.

Ejercicios

1. Muestre que hay una cantidad no enumerable de subconjuntos de \mathbb{N} que no son efectivamente enumerables.
2. Demuestre que existen funciones $f: \mathbb{N} \rightarrow \mathbb{N}$ que no son calculables.
3. Demuestre que si Σ es un alfabeto finito, $\Sigma \neq \emptyset$, entonces Σ^* contiene subconjuntos que no son efectivamente enumerables.
4. ¿Es cierto que si A no es efectivamente enumerable y $B \subseteq A$ entonces B tampoco es efectivamente enumerable?
5. Sea f_1, f_2, f_3, \dots una enumeración de todas las funciones calculables de \mathbb{N} en \mathbb{N} . Demuestre que la función $h(n) = f_n(n)$ no es calculable.
6. Demuestre que el conjunto $\bigcup_{n=1}^{\infty} \mathbb{N}^n$ si es efectivamente enumerable. Dé una biyección explícita $f: \mathbb{N} \rightarrow \bigcup_{n=1}^{\infty} \mathbb{N}^n$

1.4 Conjuntos decidibles

Dado un alfabeto finito Σ , un subconjunto de Σ^* es *decidable* si existe un algoritmo que permite averiguar de una palabra arbitraria $\alpha \in \Sigma^*$ si $\alpha \in A$ o $\alpha \notin A$.

Ejemplo 1

$\{ n \in \mathbb{N} \mid n \text{ es primo} \}$ es decidable. Dado $n \in \mathbb{N}$ (que aquí podemos identificar con $\{ / \}^*$) basta dividir a n por todos los números naturales cuyo cuadrado es menor que $n + 1$ y verificar si dà residuo cero en todos los casos o no. Para ello sólo es necesario aplicar el algoritmo de la división un número finito de veces.

Ejemplo 2

El conjunto de las fórmulas del Cálculo de Predicados es realmente infinito (aún para un léxico finito) por la presencia de la infinidad de variables x_1, x_2, \dots . Pero si identificamos la variable x_n con la palabra x_n donde n está dado en notación decimal, y hacemos algo parecido con los símbolos de relación, función y constante, podemos suponer que los términos y fórmulas del Cálculo de Predicados forman un subconjunto de Σ^* donde

$$\Sigma = \{ \neg, \wedge, \vee, \supset, \forall, \exists, =, (,), ', x, R, f, c, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \}$$

En este contexto lo siguientes conjuntos son decidibles:

$$\{ \phi \mid \phi \text{ es fórmula del C. de Predicados} \}$$

$$\{ \phi \mid \phi \text{ es axioma del C. de Predicados} \}$$

$$\{ (\phi_1, \dots, \phi_n) \mid \phi_1, \dots, \phi_n \text{ es una deducción formal en el C. de Predicados} \}$$

Teorema 1. Si $S \subseteq \Sigma^*$ es decidable entonces S es efectivamente enumerable.

Demostración. Si $S = \emptyset$ el resultado es trivial. Suponga $S \neq \emptyset$. Por el Ejemplo 1 de la sección 1.2, Σ^* es efectivamente enumerable. Sea $f: \mathbb{N} \rightarrow \Sigma^*$

una enumeración de Σ^* para producir una enumeración efectiva de S , calcule sucesivamente $f(0), f(1), \dots$ y en cada caso use el algoritmo de decisión de S para determinar si $f(n) \in S$ o $f(n) \notin S$. Continúe hasta hallar el primer k_0 tal que $f(k_0) \in S$ entonces defina:

$$g(0) = f(k_0)$$

$$g(n) = \begin{cases} f(n) & \text{si } f(n) \in S \\ g(0) & \text{si } f(n) \notin S \end{cases} \quad n > 0$$

que es una enumeración efectiva (con posibles repeticiones de $g(0)$) de S . ■

Todos los conjuntos de los ejemplos 1, 2 y 3 son efectivamente enumerables gracias al Teorema 1.

Corolario. *El conjunto de las fórmulas lógicamente válidas del Cálculo de Predicados es efectivamente enumerable.*

Demostración. Sabemos por el Teorema de Completitud de Gödel que las fórmulas lógicamente válidas coinciden con las fórmulas deducibles formalmente en el Cálculo de Predicados Axiomático. Como se anotó en el Ejemplo 3, el conjunto de las deducciones formales en dicho cálculo es efectivamente enumerable: se D_1, D_2, D_3, \dots una enumeración calculable de todas las deducciones formales correctas, entonces la función:

$$f(n) = \text{última fórmula de } D_n$$

es obviamente calculable y dà una enumeración efectiva de todos los teoremas del Cálculo de Predicados. ■

Es posible, pues, programar un computador para que genere todas las fórmulas lógicamente válidas, las leyes lógicas de primer orden.

Como hay subconjuntos de \mathbb{N} (o de Σ^*) que no son efectivamente enumerables, estos mismos conjuntos no podrán ser decidibles, por el Teorema 1. Peor aún, hay subconjuntos de \mathbb{N} o de Σ^* , que sí son efectivamente enumerables pero no son decidibles, es decir el converso del Teorema 1 no vale. Demostrar esta afirmación es mucho más difícil y exige un análisis más profundo del concepto de algoritmo. Mencionamos, sin embargo, el

siguiente ejemplo, que es en realidad uno de los hechos más importantes del Cálculo de Predicados, sin demostración.

Teorema de Church. *El conjunto de fórmulas lógicamente válidas del Cálculo de Predicados no es decidible.*

Es pues imposible diseñar un algoritmo al cual si se le dá una fórmula del Cálculo de Predicados, nos informe si ésta es lógicamente válida o no. Sin embargo el conjunto es efectivamente enumerable como vimos en el Corolario. (En cierto sentido éste hecho justifica la existencia de los matemáticos).

Más adelante daremos ejemplos de conjuntos de números que son efectivamente enumerables pero no son decidibles.

Terminamos con un teorema que garantiza en ciertos casos que un conjunto efectivamente enumerable es decidible.

Teorema 2. *Sea $A \subseteq \Sigma^*$, si A y $\Sigma^* \setminus A$ son ambos efectivamente enumerable entonces A es decidible.*

Demostración. Sea $f: \mathbb{N} \rightarrow A$ y $g: \mathbb{N} \rightarrow \Sigma^* \setminus A$ enumeraciones calculables. Dado $\alpha \in \Sigma^*$, calcule sucesivamente:

$$f(0), g(0), f(1), g(1), f(2), g(2), \dots$$

hasta que aparezca α , si aparece como $f(n)$ está en A , si aparece como $g(n)$ está en B . El punto es que α debe aparecer en algún momento. ■

Ejercicios

1. Describa en líneas generales algoritmos para decidir pertenencia a los conjuntos del ejemplo 3.
2. Demuestre que son decidibles los conjuntos del ejercicio 2 (b), (c), (e), (f), (g), sección 1.2. Explique dónde puede haber dificultades para la decidibilidad de los conjuntos del mismo ejercicio, partes (d) y (h).

3. Sean $\sigma_1, \sigma_2, \dots, \sigma_n$ fórmulas del Cálculo de Predicados, muestre que el conjunto $\{ \phi \mid \sigma_1, \sigma_2, \dots, \sigma_n \vdash \phi \}$ es efectivamente enumerable.
4. Suponga que S es infinito en el Teorema 1, y modifique la definición de la función g en la prueba de manera que resulte monótona en longitud, es decir $m < n$ implica $|g(n)| \leq |g(m)|$, donde $|\alpha|$ denota la longitud de la palabra α .
5. Demuestre que si $S \subseteq \mathbb{N}$ y $f: \mathbb{N} \rightarrow S$ es una función sobreyectiva calculable monótona entonces S es decidible. (Considere el caso finito e infinito separadamente.)
6. Demuestre que si $f: \mathbb{N} \rightarrow \mathbb{N}$ es una biyección calculable entonces para todo $S \subseteq \mathbb{N}$,
 - a. S es efectivamente enumerable sii $f(S)$ es e.e.
 - b. S es decidible sii $f(S)$ es decidible.
7. Sea $\Gamma = \{ \sigma_1, \dots, \sigma_n \}$ un conjunto de sentencias del Cálculo de Predicados sobre un léxico L , con la siguiente propiedad: para toda sentencia sobre L se tiene $\Gamma \vdash \phi$ o $\Gamma \vdash \neg \phi$ (es decir, Γ es una teoría completa). Demuestre que entonces el conjunto $\{ \phi \text{ sobre } L \mid \Gamma \vdash \phi \}$ es decidible.
8. Muestre que el conjunto

$$\{ \phi \mid \phi \text{ fórmula del Cálculo de Predicados y } \not\vdash \phi \}$$
 no es efectivamente enumerable (use el Teorema de Church)
- 9.* Sea Γ un conjunto infinito, efectivamente enumerable, de fórmulas del Cálculo de Predicados. Muestre que el conjunto $\{ \phi \mid \Gamma \vdash \phi \}$ es efectivamente enumerable.

10.** El problema de la parada

Sea Σ el alfabeto de un lenguaje de programación suficientemente poderoso para permitir programar en él toda clase de algoritmos (por ejemplo Fortran). Demuestre, o al menos convénzase de las siguientes afirmaciones.

- a. El conjunto de los programas es decidable y por lo tanto efectivamente enumerable.
- b. Dado un programa P y un número natural n , si hacemos correr el programa con el número n como dato de entrada (en notación adecuada) puede suceder una de dos cosas: el programa se detiene después de cierto número de pasos, dando un resultado que podemos considerar siempre numérico, o la ejecución del programa nunca se detiene, es decir entra en "loop". Sea P_1, P_2, P_3, \dots una enumeración efectiva de todos los programas, demuestre que el siguiente conjunto es efectivamente enumerable:

$$\mathcal{P} = \{ (n, k) \in \mathbb{N} \times \mathbb{N} \mid P_n \text{ con el dato } k \text{ se detiene} \}$$

- c. Demuestre que el conjunto

$$D = \{ n \in \mathbb{N} \mid P_n \text{ con el dato } n \text{ se detiene} \}$$

es efectivamente enumerable pero no es decidable. Para mostrar que no es decidable, suponga por contradicción que sí lo es y demuestre que la siguiente función es calculable:

$$f(n) = \begin{cases} (\text{resultado de } P_n \text{ con dato } n) + 1, & \text{si } n \in D \\ 0 & \text{si } n \notin D \end{cases}$$

Sea P_i un programa que calcula a f , entonces al tratar de calcular $f(i)$ se llega a una contradicción.

- d. Muestre que el conjunto \mathcal{P} no es decidable. (Esta es la famosa insolubilidad del problema de la parada.)

- e. Muestre que el conjunto

$$\{ n \in \mathbb{N} \mid P_n \text{ con el dato } n \text{ no se detiene} \}$$

no es efectivamente enumerable.

- f. Indique en dónde los anteriores argumentos pueden dejar de ser totalmente rigurosos.

Capítulo 2

Funciones recursivas

En las secciones anteriores hemos utilizado la noción de "función calculable por medio de un algoritmo", surge la pregunta natural de ¿cuáles son las funciones calculables? ¿cómo caracterizarlas intrínsecamente? En esta sección damos una respuesta a esta pregunta, debida a Gödel (1934), para el caso de las funciones entre números naturales. Será evidente que las funciones recursivas aquí introducidas son calculables, y demostraremos en las secciones siguientes que toda función es calculable por una *máquina de Turing* (es decir un algoritmo, en un sentido muy preciso) y por lo tanto toda función calculable por medio de un programa de computador, por ejemplo, es recursiva. La afirmación de que las funciones calculables entre números naturales son precisamente las funciones recursivas que aquí introducimos se llama la *Tesis de Church* (Alonso Church, 1936).

2.1 Funciones recursivas primitivas

Comenzamos definiendo una subclase de la clase de todas las funciones recursivas, que contiene la mayor parte de las funciones numéricas que aparecen en la práctica.

Definición. Una función $h: \mathbb{N}^k \rightarrow \mathbb{N}$ se dice *recursiva primitiva* (pr) si puede obtenerse por una sucesión de aplicaciones de las reglas siguientes.

I. Las siguientes funciones, llamadas *básicas* son recursivas primitivas

$$Z(x) = 0 \quad \text{función constante } 0$$

$$S(x) = x+1 \quad \text{función sucesor}$$

$$P_k^n(x_1, \dots, x_n) = x_k \quad \begin{matrix} \text{k-ésima proyección en } n \text{ variables,} \\ \text{para cada } n, k \in \mathbb{N}^+ \text{ con } k \leq n. \end{matrix}$$

II. Composición

Si $f(y_1, \dots, y_n)$ y $g_1(x_1, \dots, x_k), \dots, g_n(x_1, \dots, x_k)$ son pr entonces $h(x_1, \dots, x_k) = f(g_1(x_1, \dots, x_k), \dots, g_n(x_1, \dots, x_k))$.

III. Recurrencia Primitiva

Si $f(x_1, \dots, x_k)$ y $g(x_1, \dots, x_k, y, z)$ son pr entonces la función $h(x_1, \dots, x_k, y)$ definida por:

$$\begin{cases} h(x_1, \dots, x_k, 0) = f(x_1, \dots, x_k) \\ h(x_1, \dots, x_k, S(y)) = g(x_1, \dots, x_k, y, h(x_1, \dots, x_k, y)) \end{cases}$$

es pr.

En la composición, las funciones g_1, \dots, g_n no necesariamente dependen de todas las variables x_1, \dots, x_k , y en el esquema de recurrencia primitiva tampoco es necesario que todas las variables x_1, \dots, x_k y aparezcan, más aún, puede ser que no aparezca ninguna; por lo tanto la regla III contiene los dos esquemas especiales de recurrencia primitiva siguientes:

$$\begin{cases} h(0) = n_0 \in \mathbb{N} \\ h(S(y)) = g(y, h(y)) \end{cases}$$

$$\begin{cases} h(\vec{x}, 0) = f(\vec{x}) \\ h(\vec{x}, S(y)) = g(h(\vec{x}, y), y) \end{cases}$$

Ejemplos

(a) La función identidad $I(x) = x = P_1^1(x)$ es pr por definición.

(b) La función constante $C_n(x) = n$ para todo x es pr, pues $C_0(x) = Z(x)$ y $C_n(x) = S(S \dots S(Z(x)))$ para $n \geq 1$ (regla II).

(c) $h(x, y) = x + y$ es pr. Sea $f(x) = P_1^1(x) = x$, $g(x, y, z) = S(P_3^3(x, y, z)) = z$, entonces:

$$x + 0 = f(x) = x$$

$$x + S(y) = g(x, y, x+y) = S(x+y)$$

(d) $h(x, y) = x \cdot y$ es pr. Sea $f(x) = Z(x)$, $g(x, y, z) = x + z$, entonces :

$$x \cdot 0 = Z(x) = 0$$

$$x \cdot S(y) = g(x, y, x \cdot y) = x + x \cdot y$$

(c) $h(x, y) = x^y$ es pr. Sea $f(x) = S(Z(x))$, $g(x, y, z) = x \cdot z$,

$$x^0 = S(z(x)) = 1$$

$$x^{y+1} = g(x, y, x^y) = x \cdot x^y$$

(f) La *resta truncada* se define

$$x \dashv y = \begin{cases} x - y & \text{si } x \geq y \\ 0 & \text{si } x < y \end{cases}$$

y es pr. Primero se muestra que $h(x) = x \dashv 1$ es pr: $h(0) = Z(0) = 0$, $h(S(x)) = x$. Ahora definimos :

$$x \dashv 0 = x$$

$$x \dashv S(y) = h(x \dashv y) = (x \dashv y) \dashv 1.$$

(g) La función $|x - y| = (x \dashv y) + (y \dashv x)$ es pr.

(h) Las funciones

$$Sg(x) = \begin{cases} 0 & \text{si } x = 0 \\ 1 & \text{si } x \geq 1 \end{cases} \quad Sg(x) = 1 \dashv Sg(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x \geq 1 \end{cases}$$

son pr pues

$$Sg(0) = Z(0) = 0$$

$$Sg(S(x)) = S(Z(x)) = 1$$

(i) Sea C_p la función característica de los pares; C_p es pr pues

$$C_p(0) = S(Z(0)) = 1$$

$$C_p(x+1) = 1 \pm C_p(x)$$

(j) Si $p(x_1, \dots, x_k)$ es un polinomio con coeficientes en \mathbb{N} la función $P: \mathbb{N}^k \rightarrow \mathbb{N}$ definida por p es pr por ser composición de proyecciones $P_i^k(x_1, \dots, x_k)$ y constantes $C_n(x_i)$ por medio de sumas y multiplicaciones.

Lema 1. *Sea $f(\vec{x}, y)$ pr entonces las funciones*

$$g(\vec{x}, z) = \sum_{i=0}^z f(\vec{x}, i), \quad h(\vec{x}, z) = \prod_{i=0}^z f(\vec{x}, i) \text{ son pr.}$$

Demostración. $g(\vec{x}, 0) = f(\vec{x}, 0)$

$$g(\vec{x}, S(y)) = f(\vec{x}, S(y)) + g(\vec{x}, y)$$

$$h(\vec{x}, 0) = f(\vec{x}, 0)$$

$$h(\vec{x}, S(y)) = f(\vec{x}, S(y)) \cdot h(\vec{x}, y). \blacksquare$$

Como aplicación del lema anterior tenemos que las siguientes funciones son pr:

$$f(n, m) = \begin{cases} 1 & \text{si } n \text{ divide a } m \\ 0 & \text{si } n \text{ no divide a } m \end{cases}$$

pues $f(n, m) = 1 \pm \prod_{i=1}^n Sg | n - mi |$.

$$g(n) = \text{número de divisores de } n$$

pues $g(n) = \sum_{i=0}^n f(i, n)$.

Finalmente,

$$h(n) = \begin{cases} 1 & \text{si } n \text{ es primo} \\ 0 & \text{de lo contrario} \end{cases}$$

pues $h(n) = 1 \pm |g(n) - SS(Z(n))|$, ya que h es primo si y sólo si tiene exactamente dos divisiones. Usando la misma idea podemos mostrar que la función de Euler:

$$\phi(n) = \text{Card} \{ m \mid m \leq n, m \text{ relativamente primo a } n \}$$

es pr. Primero definimos

$$d(n,m) = \sum_{i=0}^{n+m} f(i,n) \cdot f(i,m)$$

que da el número de divisores comunes de n y m . Entonces i es relativamente primo a n si y solo si $d(i,n) = 1$, por lo tanto

$$\phi(n) = \sum_{i=0}^n (1 \pm |d(i,n) - 1|)$$

pues el i -ésimo sumando es 1 cuando $d(i,n) = 1$, y 0 de lo contrario. Prácticamente todas las funciones elementales de la Teoría de Números son **primitivamente recursivas**.

Como último ejemplo considere la biyección $g : \mathbb{N}^2 \rightarrow \mathbb{N}$,

$$g(x,y) = (x+y)(x+y+1)/2 + y$$

Esta es primitivamente recursiva pues

$$g(x,y) = \left(\sum_{i=0}^{x+y} i \right) + y$$

Por inducción en k resulta que las biyecciones canónicas $g_k : \mathbb{N}^k \rightarrow \mathbb{N}$ definidas por

$$g_2(x_1, x_2) = g(x_1, x_2)$$

$$g_{k+1}(x_1, \dots, x_k, x_{k+1}) = g_2(g_k(x_1, \dots, x_k), x_{k+1})$$

son primitivamente recursivas.

No toda función de \mathbb{N}^k en \mathbb{N} es recursiva primitiva como lo muestra el lema siguiente.

Lema 2. *El conjunto de las funciones recursivas primitivas es enumerable, por lo tanto existen funciones de \mathbb{N}^k en \mathbb{N} que no son recursivas primitivas para cada $k \in \mathbb{N}^k$.*

Demostración. Defina

$$\mathcal{P}_0 = \{ f \mid f \text{ es función básica} \}$$

$$\mathcal{P}_{n+1} = \mathcal{P}_n \cup C(\mathcal{P}_n) \cup RP(\mathcal{P}_n)$$

donde

$$C(\mathcal{P}_n) = \{ f(g_1(\vec{x}), \dots, g_m(\vec{x})) \mid f, g_i \in \mathcal{P}_n, m \in \mathbb{N}^+ \}$$

$$RP(\mathcal{P}_n) = \{ h \mid h \text{ es definida por recurrencia primitiva de } f, g \in \mathcal{P}_n \}$$

Es fácil mostrar que \mathcal{P}_0 es enumerable. Además, si \mathcal{P}_n es enumerable, el conjunto de sucesiones finitas de funciones en \mathcal{P}_n es enumerable; por lo tanto $C(\mathcal{P}_n)$ es enumerable. Similarmente, los pares de funciones en \mathcal{P}_n son enumerables, y $RP(\mathcal{P}_n)$ es enumerable. De esta manera queda demostrado que \mathcal{P}_{n+1} es enumerable, y por inducción todo \mathcal{P}_n es enumerable. Evidentemente, la clase de las funciones recursivas primitivas coincide con

$$\mathcal{P} = \bigcup_{i=1}^{\infty} \mathcal{P}_n$$

y por lo tanto debe ser enumerable. Ahora, sabemos que el conjunto $\{f \mid f: \mathbb{N}^k \rightarrow \mathbb{N}\}$ es no enumerable, por lo tanto $\{f \mid f: \mathbb{N}^k \rightarrow \mathbb{N}\} - \mathcal{P}$ sigue siendo no enumerable. ■

El lema anterior no es constructivo, en el sentido de que no nos dà ejemplos explícitos de funciones que no sean pr. Más adelante daremos tales ejemplos (vea ejercicio 5).

Ejercicios

1. Sea $f: \mathbb{N} \rightarrow \mathbb{N}$ pr, demuestre que $h(n,x) = f(f(\dots f(x)\dots))$, n veces, es pr.
2. Demuestre que las funciones siguientes son pr.
 - a. $f(n) = n!$
 - b. $\min(m,n) =$ mínimo entre m y n
 - c. $\max(m,n) =$ máximo entre m y n
 - d. $f(n) = n(n+1)/2$
 - e. $f(n) = [n/2]$, mayor entero menor o igual que $n/2$
 - f. $f(n) = n^n$
 - g. $f(n,k) = (n^{k+1} - 1)/(n-1)$, si $n \neq 1$, $f(1,k) = k+1$
3. Demuestre que si $f(x,y)$ es pr, lo mismo es cierto de $g(x) = f(x,x)$, $h(x,y) = f(y,x)$.
- 4.* Demuestre que las funciones siguientes son pr.
 - a. $r(m,n) =$ residuo de la división $m \div n$, ó 0 si $n=0$
 - b. $q(m,n) =$ cociente de la división $m \div n$, ó 0 si $n=0$
 - c. $\text{mcd}(m,n) =$ máximo común divisor de m,n
 - d. $p(n) =$ n -ésimo primo
 - e. $[\sqrt{n}] =$ máximo número entero menor o igual que \sqrt{n}
 - *f. $f(n) =$ n -ésimo dígito en la expansión decimal de $\pi/4$
5. Sea f_1, f_2, \dots una enumeración de todas las funciones pr de \mathbb{N} en \mathbb{N} , demuestre que $g(x) = f_x(x)$, $h(x,y) = f_x(y)$ no son pr.
6. Demuestre que la función $\pi(x) =$ número de primos menores o iguales que x es pr.

2.2 Funciones recursivas

Una función $f(x_1, \dots, x_k, y)$ es *regular para y* si para todo n_1, \dots, n_k existe m tal que $f(n_1, \dots, n_k, m) = 0$, en tal caso puede definirse una nueva función: $h(f_n) = \mu y (f(x_1, \dots, x_k, y) = 0)$, donde $\mu y \dots$ es el "mínimo y tal que ..."; h es la *minimización* de f con respecto a y . Si a la definición de funciones primitivamente recursivas añadimos la operación de minimización obtenemos las funciones recursivas.

Definición. Una función f es *recursiva* si es una de las funciones básicas o puede obtenerse de las funciones básicas por un número finito de aplicaciones de composición, recurrencia primitiva y la nueva regla:

IV. *Minimización* (de funciones regulares)

Si $f(\vec{x}, y)$ es recursiva y regular para y entonces $h(\vec{x}) = \mu y (f(\vec{x}, y) = 0)$ es recursiva.

Toda función primitivamente recursiva es recursiva. Como existe un método claro para calcular $\mu y (f(\vec{x}, y) = 0)$ en el caso en que $f(\vec{x}, y)$ sea calculable y regular (calcule sucesivamente $f(\vec{x}, 0), f(\vec{x}, 1), \dots$ hasta obtener el primer 0), resulta que las funciones recursivas son efectivamente calculables. Veremos que hay argumentos muy fuertes para afirmar que la clase de las funciones recursivas incluye todas las funciones efectivamente calculables. Tal afirmación es la *Tesis de Church* y equivale a aceptar que la definición de función recursiva es la formulación matemática correcta de la noción (intuitiva y por lo tanto imprecisa) de función calculable. Sin embargo nosotros no usaremos en ningún momento dicha Tesis.

Como en el caso de las funciones primitivamente recursivas, puede demostrarse que la clase \mathcal{R} de las funciones recursivas es enumerable y por lo tanto para cada k existen funciones de los \mathbb{N}^k en \mathbb{N} que no son recursivas. Se puede mostrar además que el operador de minimización permite definir funciones que no son pr, es decir, \mathcal{R} es una extensión propia de \mathcal{P} (véase ejemplo 2).

Ejemplo 1

La función $p(n) = n\text{-ésimo primo}$ es recursiva. Recuerde que la función

$$f(x) = \begin{cases} 1 & \text{si } x \text{ es primo} \\ 0 & \text{si } x \text{ no es primo} \end{cases}$$

es primitivamente recursiva y por tanto recursiva (sec 1.1). Como hay una infinidad de primos, la función

$$h(x, y) = \overline{\text{Sg}}(f(y) * (y - x)) = \begin{cases} 0 & \text{si } y \text{ primo, } y > x \\ 1 & \text{de lo contrario} \end{cases}$$

es regular. Por lo tanto la función

$$g(x) = \mu y (h(x, y) = 0)$$

es recursiva y podemos definir por recurrencia

$$p(0) = 2$$

$$\begin{aligned} p(n+1) &= g(p(n)) = \mu y (h(p(n), y) = 0) \\ &= \mu y (y \text{ es primo, } y > p(n)) \end{aligned}$$

Veremos que esta función es en efecto primitivamente recursiva.

Ejemplo 2

(Función de Ackermann). La siguiente función es recursiva pero no es pr, para la demostración de estos hechos vea ejercicios. Defina $A(x, y)$ por el siguiente esquema:

$$A(0, y) = y + 1$$

$$A(x+1, 0) = A(x, 1)$$

$$A(x+1, y+1) = A(x, A(x+1, y))$$

Como ilustración considere el cálculo de $A(2, 1)$

$$\begin{aligned}
 A(2,1) &= A(1, A(2,0)) \\
 &= A(1, A(1,1)) \\
 &= A(1, A(0, A(1,0))) \\
 &= A(1, A(0, A(0,1))) \\
 &= A(1, A(0,2)) \\
 &= A(1,3) \\
 &= A(0, A(1,2)) \\
 &= A(0, A(0, A(1,1))) \\
 &= A(0, A(0, A(0, A(1,0)))) \\
 &= A(0, A(0, A(0, A(0,1)))) \\
 &= A(0, A(0, A(0,2))) \\
 &= A(0, A(0,3)) \\
 &= A(0,4) \\
 &= 5
 \end{aligned}$$

Ejercicios

1. Sea $A(x,y)$ la función de Ackermann y defina $f_n(x) = A(n,x)$ para cada $n \in \mathbb{N}$. Demuestre que cada una de las funciones f_0, f_1, \dots es pr.
2. Sea $f: \mathbb{N} \rightarrow \mathbb{N}$ recursiva y sobreyectiva, muestre que existe $g: \mathbb{N} \rightarrow \mathbb{N}$ recursiva tal que $f \circ g = I$. Sea $f: \mathbb{N} \rightarrow \mathbb{N}$ una biyección, entonces f es recursivo si y solo si f^{-1} es recursiva.
- 3.* Demuestre las siguientes propiedades de la función de Ackermann, donde $f_n(y) = A(n,y)$.
 - a. $y < f_n(y)$ (inducción en n , luego en y)
 - b. $f_n(y) < f_{n+1}(y)$ (inducción en n)
 - c. $f_n(y+1) \leq f_{n+1}(y)$ (inducción en y , n fijo)

- d. $f_n(y) < f_{n+1}(y)$ (de (b) y (c))
e. $f_n(y) + f_m \leq 2f_M(y) \leq f_{M+3}(y)$ para algún M
f. $\forall n_1, \dots, n_k \exists m \text{ tal que } \sum_{i=1}^k f_{n_i}(y) < f_m(y)$
g. $f_n(f_m(y)) \leq f_{n+m+1}(y)$
- 4.* Demuestre por inducción en la construcción de $\vec{g(x)}$ que para toda función pr $g(x_1, \dots, x_k)$ existe M tal que

$$g(x_1, \dots, x_k) < f_M(x_1 + \dots + x_k)$$
- 5.* Usando (4*) demuestre que $A(x,y)$ no es pr. (Ayuda: si $A(x,y)$ es pr entonces $g(x) = A(x,x)$ es pr).
- 6.** Demuestre que $A(x,y)$ es recursiva (véase Capítulo 5).
7. Sea f_0, f_1, f_2, \dots , una enumeración de todas las funciones recursivas de una variable. Demuestre que $h(n,m) = f_n(m)$ no es recursiva.

2.3 Conjuntos recursivos

Un conjunto $S \subseteq \mathbb{N}^k$ es *recursivo* si su función característica

$$C_S(x_1, \dots, x_k) = \begin{cases} 1 & \text{si } (x_1, \dots, x_k) \in S \\ 0 & \text{si } (x_1, \dots, x_k) \notin S \end{cases}$$

es recursiva. Una relación k -ádica $R(x_1, \dots, x_k)$, es *recursiva* si su función característica

$$C_R(x_1, \dots, x_k) = \begin{cases} 1 & \text{si } R(x_1, \dots, x_k) \\ 0 & \text{si } \neg R(x_1, \dots, x_k) \end{cases}$$

es recursiva. Obviamente esto equivale a decir que el conjunto $\{(n_1, \dots, n_k) \in \mathbb{N}^k \mid R(n_1, \dots, n_k)\}$ es recursivo, además S es recursivo si y

sólo si la relación " $\vec{x} \in S$ " es recursiva. Sin embargo, algunos resultados se expresan más convenientemente en términos de relaciones. Esta definición corresponde a la noción intuitiva de conjunto o relación decidable. Si la función característica es primitivamente recursiva, decimos que el conjunto o relación es primitivamente recursivo.

Ejemplos

- (a) Todo conjunto finito es (primitivamente) recursivo. Si $S = \{n_1, \dots, n_k\}$ entonces $C_S(x) = \overline{\text{Sg}} | x - n_1 | + \dots + \overline{\text{Sg}} | x - n_k |$.
- (b) Las relaciones $x = y$, $x < y$ son (primitivamente) recursivas pues $C = (x, y) = \overline{\text{Sg}} | x - y |$ $C < (x, y) = \text{Sg} (y - x)$.
- (c) De los ejemplos de la sección 1.1 resulta que las siguientes relaciones y conjuntos son (primitivamente) recursivos :

$$\{x \mid x \text{ es primo}\}$$

$$R(x, y) \Leftrightarrow x \text{ divide a } y$$

$$R(x, y) \Leftrightarrow x, y \text{ son primos relativos}$$

- (d) Si $R(x_1, \dots, x_k)$ es (primitivamente) recursiva y $f_1(\vec{y}), \dots, f_k(\vec{y})$ son funciones (primitivamente) recursivas entonces la relación

$$T(\vec{y}) \Leftrightarrow R(f_1(\vec{y}), \dots, f_k(\vec{y}))$$

es (primitivamente) recursiva. En particular,

$$f_1(\vec{x}) = f_2(\vec{x}), f_1(\vec{x}) < f_2(\vec{x})$$

son relaciones (primitivamente) recursivas.

Si $R(\vec{x}, y, z)$ es una relación las expresiones

$$\forall y \leq z R(\vec{x}, y, z) \quad \exists y \leq z R(\vec{x}, y, z)$$

abrevian respectivamente:

$$\forall y [y \leq z \supset R(\vec{x}, y, z)] \quad \exists y [y \leq z \wedge R(\vec{x}, y, z)]$$

Llamamos *cuantificadores acotados* a los simbolos " $\forall y \leq z$ ", " $\exists y \leq z$ ". Si $R_1(\vec{x})$ y $R_2(\vec{y})$ son relaciones las relaciones $\neg R_1(\vec{x})$, $R_1(\vec{x}) \wedge R_2(\vec{y})$, $R_1(\vec{x}) \vee R_2(\vec{y})$, $R_1(\vec{x}) \supset R_2(\vec{y})$ tienen el significado obvio.

Lema 1. *Las relaciones recursivas (respectivamente, p r) son cerradas bajo \neg , \wedge , \vee , \rightarrow y cuantificación acotada.*

Demostración. Basta mostrarlo para \neg , \wedge y " $\forall y \leq z$ " pues los dos primeros forman un conjunto completo de conectivos y $\exists y \leq z R \Leftrightarrow \neg \forall y \leq z \neg R$. Sea C_R la función característica de la relación $R(\vec{x})$ entonces

$$C_{\neg R}(\vec{x}) = 1 \pm C_R(\vec{x}) \quad (1)$$

Sean C_{R_1} , C_{R_2} las funciones características de $R_1(\vec{x}, \vec{y})$ y $R_2(\vec{y}, \vec{z})$ entonces

$$C_{R_1 \wedge R_2}(\vec{x}, \vec{y}, \vec{z}) = C_{R_1}(\vec{x}, \vec{y}) \cdot C_{R_2}(\vec{y}, \vec{z}) \quad (2)$$

Sea $R(\vec{x}, \vec{y}, \vec{z})$ una relación y defina $T(\vec{x}, \vec{z}) \Leftrightarrow \forall y \leq z R(\vec{x}, \vec{y}, \vec{z})$ entonces

$$C_T(\vec{x}, \vec{z}) = \prod_{i=0}^z C_R(\vec{x}, i, \vec{z}). \quad (3)$$

Las operaciones (1) (2) (3) transforman funciones recursivas (ó p r) en funciones recursivas (ó p r), lo cual demuestra el lema. ■

Colorario 2 *La clase de los conjuntos recursivos (respectivamente p r) es cerrada bajo unión, intersección y complementación.*

Llamamos lenguaje de la aritmética al lenguaje de primer orden construido de los símbolos de función binaria $+$, \cdot , de relación $<$, $=$ y de constante $0, 1$. En este lenguaje cada número natural n tiene un nombre $\underline{n} = 1+1+\dots+1$, n -veces. Una relación $R(x_1, \dots, x_k)$ de \mathbb{N}^k es *definible por una fórmula* $\phi(x_1, \dots, x_k)$ del lenguaje de la aritmética si $R(n_1, \dots, n_k) \Leftrightarrow \mathcal{N} \models \phi[n_1, \dots, n_k]$, para todo n_1, \dots, n_k en \mathbb{N} , donde $\mathcal{N} = \langle \mathbb{N}, +, \cdot, <, 0, 1 \rangle$.

Colorario 3 *Si la relación $R(x_1, \dots, x_k)$ es definible por una fórmula de la aritmética con todos sus cuantificadores acotados entonces es primitivamente recursiva.*

Demostración. Por inducción en la complejidad de la fórmula $\phi(x_1, \dots, x_k)$ que define a R. Si ϕ es átomica debe tener una de las formas

$$t_1(w_1, \dots, w_k) = t_2(w_1, \dots, w_k) \quad (1)$$

$$t_1(w_1, \dots, w_k) < t_2(w_1, \dots, w_k) \quad (2)$$

en donde t_1, t_2 son términos construidos de 0, 1, +, *. Entonces $t_1^{\mathcal{N}}, t_2^{\mathcal{N}}$ (su interpretación en \mathbb{N}) son funciones primitivamente recursivas, y las relaciones definidas por (1) y (2) son precisamente

$$t_1^{\mathcal{N}}(w) = t_2^{\mathcal{N}}(w)$$

$$t_1^{\mathcal{N}}(w) < t_2^{\mathcal{N}}(w)$$

que son recursivas. Los pasos inductivos de la prueba están dados por el lema 1. ■

Ejemplos

- (a) Una nueva demostración de que la propiedad "x es primo" es p.r. resulta de que está definida por la fórmula con cuantificadores acotados $\forall y \leq x \forall z \leq x (x = y * z \rightarrow (x = v \vee x = z))$. Más aun, analizando la prueba del lema 1, puede hallarse una "fórmula" que dé la función característica:

$$C_P(x) = \prod_{y=0}^x \prod_{z=0}^x Sg (Sg |x-yz| + \overline{Sg} |x-y| + \overline{Sg} |x-z|)$$

- (b) La propiedad "x es un número perfecto" (es decir, x es la suma de sus divisores propios) es primitivamente recursiva. Sea $f(x, y)$ la función característica de "x divide a y" entonces la propiedad de ser perfecto es definible por la ecuación entre funciones p.r.:

$$\sum_{i=0}^x i \cdot f(i, x) = 2 \cdot x$$

Ejercicios

1. Demuestre que la clase de subconjuntos recursivos de \mathbb{N}^k es enumerable.
2. Sea $f : \mathbb{N}^k \rightarrow \mathbb{N}$ recursiva, muestre que si $S \subseteq \mathbb{N}$ es recursivo entonces $f^{-1}(S)$ es recursivo. (El converso no vale aun si f es sobreyectiva.)
3. Si S, S' son recursivos entonces $S \times S'$ es recursivo.
4. $f : \mathbb{N}^k \rightarrow \mathbb{N}$ es recursiva si y solo si el gráfico de f , $G(f) = \{(\vec{x}, y) \mid y = f(\vec{x})\}$, es recursivo.
5. Demuestre que las relaciones siguientes son (primitivamente) recursivas.
 - a. $x \equiv y \pmod{z}$
 - b. "x es el máximo primo que divide a y".
- 6.* Sea $f(x)$ recursiva, $g(x)$ pr, y $f(x) \leq g(x)$ entonces $f(x)$ es pr.
- 7.* f es pr si y solamente si f es recursiva y existe M tal que $f(x) \leq A(M; x)$ para toda x .

2.4 Funciones definidas de relaciones

Sea S un subconjunto recursivo de \mathbb{N}^k y sean $f, g : \mathbb{N}^k \rightarrow \mathbb{N}$ funciones recursivas entonces la función

$$h(\vec{x}) = \begin{cases} f(\vec{x}) & \text{si } \vec{x} \in S \\ g(\text{roman } \vec{x}) & \text{si } \vec{x} \notin S \end{cases}$$

es recursiva pues si C_S es la función característica de S ,

$$h(\vec{x}) = C_S(\vec{x}) f(\vec{x}) + (1 - C_S(\vec{x})) g(\vec{x})$$

El resultado anterior se puede generalizar por inducción:

Lema 1 (Definición por casos). Sean S_1, \dots, S_l subconjuntos recursivos disyuntos de \mathbb{N}^k y sean $f_i: \mathbb{N}^k \rightarrow \mathbb{N}$, $i = 1, \dots, l+1$ funciones recursivas entonces la función:

$$h(\vec{x}) = \begin{cases} f_1(\text{roman } \vec{x}) & \text{si } \vec{x} \in S_1 \\ \vdots \\ f_l(\vec{x}) & \text{si } \vec{x} \in S_l \\ f_{l+1}(\vec{x}) & \text{si } \vec{x} \notin S_1 \cup \dots \cup S_l \end{cases}$$

es recursiva.

El resultado anterior vale también para conjuntos y funciones primitivamente recursivas. Además el Lema 1 vale para relaciones $R_1 \dots R_l$ mutuamente exclusivas, en lugar de conjuntos. Sea ahora $R(\vec{x}, y)$ una relación recursiva *regular*, es decir $\forall x_1 \dots x_k \exists y R(x_1 \dots x_k, y)$ entonces la función

$$f(\vec{x}) = \mu y R(\vec{x}, y) = \mu y (|C_R(\vec{x}, y) - 1| = 0)$$

es recursiva. El resultado análogo para relaciones primitivamente recursivas no vale, minimización de relaciones p.r. no es necesariamente p.r. Sin embargo, minimización *acotada* de relaciones p.r. es p.r. Defina

$$\mu y \leq z R(\vec{x}, y, z) = \begin{cases} R(\vec{x}, y, z) & \text{si } \exists y \leq z R(\vec{x}, y, z) \\ 0 & \text{de lo contrario} \end{cases}$$

Lema 2 (Minimización acotada). Si $R(\vec{x}, y)$ es una relación p.r. entonces la función $h(\vec{x}, z) = \mu y \leq z R(\vec{x}, y)$ es p.r.

Demostración

$$h(\vec{x}, 0) = 0$$

$$h(\vec{x}, z+1) = \begin{cases} h(\vec{x}, z) & \text{si } \exists y \leq z R(\vec{x}, y) \\ z + 1 & \text{si } \forall y \leq z \neg R(\vec{x}, y) \wedge R(\vec{x}, z+1) \\ 0 & \text{de lo contrario} \end{cases}$$

Cada una de las relaciones a la derecha es p.r. por el lema 1 de la sección 2.3. La definición anterior es una combinación de definición por casos p.r. y recursividad primitiva, por tanto h es p.r. ■

Del lema anterior resulta que si $R(\vec{x}, z, y)$ es p.r. entonces la función $f(\vec{x}, z, w) = \mu y \leq w R(\vec{x}, z, y)$ es p.r. Identificando w con z en f tenemos que

$$h(\vec{x}, z) = \mu y \leq z R(\vec{x}, z, y)$$

es p.r. En general, si $r(\vec{x})$ es una función p.r., reemplazando w por $r(\vec{x})$ en f y suponiendo que z forma parte de la lista \vec{x} , tenemos que

$$h(\vec{x}) = \mu y \leq r(\vec{x}) R(\vec{x}, y)$$

también es p.r.

Ejemplo 1

Mostramos que $p(n) = "n\text{-ésimo primo}"$ es p.r. Sea $P(y)$ la relación " y es primo" que es p.r. entonces la función $g(x) = \mu y \leq (x! + 1)[P(y) \wedge y > x]$ es p.r. por las observaciones anteriores. Como se sabe que entre un primo p y $p! + 1$ existe un primo distinto de p podemos definir :

$$p(0) = 2$$

$$p(n+1) = g(p(n)) = \mu y \leq (p(n)! + 1) [P(y) \wedge y > p(n)]$$

que es p.r.

Ejemplo 2

Las funciones $q(m, n)$, $r(m, n)$, que dan el cociente y el residuo, respectivamente, de la división $m \div n$, ó dan 0 cuando $n = 0$, son recursivas pues

$$r(m, n) = \mu r \leq m [\exists q \leq m (m = nq + r)]$$

$$q(m, n) = \mu q \leq m [m = nq + r(m, n)]$$

Ejemplo 3

Sea $g : \mathbb{N}^2 \rightarrow \mathbb{N}$ la biyección $g(x,y) = (x+y)(x+y+1)/2 + y$ que es p.r. Sea $f: \mathbb{N} \rightarrow \mathbb{N}^2$, $f(n) = (J(n), K(n))$ la función inversa entonces J y K son p.r. pues

$$J(n) = \mu x \leq n [\exists y \leq n (g(x,y) = n)]$$

$$K(n) = \mu y \leq n [\exists y \leq n (g(x,y) = n)]$$

El último ejemplo muestra que podemos generalizar la noción de función recursiva, $f: \mathbb{N}^k \rightarrow \mathbb{N}^l$, $f(x) = (f_1(x), \dots, f_l(x))$ es recursiva si y solo si cada $f_i: \mathbb{N}^k \rightarrow \mathbb{N}$ $i = 1, \dots, l$ es recursiva. De esta manera la función $f: \mathbb{N} \rightarrow \mathbb{N}^2$ del ejemplo anterior es (primitivamente) recursiva. De la misma manera puede demostrarse que si $g_k: \mathbb{N}^k \rightarrow \mathbb{N}$ es la biyección canónica donde $g_2 = g$ y $g_{k+1}(X_1, \dots, X_k, X_{k+1}) = g_k(X_1, \dots, X_k), X_{k+1}$ son p.r. entonces las inversas $f^k: \mathbb{N} \rightarrow \mathbb{N}^k$ son p.r.

Ejercicios

1. Demuestre que las siguientes funciones son p.r.
 - a. $\text{mcd}(m, n) =$ máximo común divisor de m, n
 - b. $f(x) =$ máximo primo que divide a x
 - c. $f(n, y) =$ exponente del primo $p(n)$ en la descomposición prima de y .
2. Muestre que si $R(\vec{x}, y, z)$ es p.r. entonces $\max y \leq z \quad R(\vec{x}, y, z)$ es una función p.r.
3. Sean $\{0, 1, \dots, n-1\}$ los residuos módulo n y sea $x(\text{mod } n)$ el residuo de x , demuestre que las funciones siguientes son p.r. $f(x) = x(\text{mod } n)$
 $f(x, y) = x+y \pmod{n}$ $f(x, y) = x \cdot y \pmod{n}$
- 4.* Sea $g: \mathbb{N}^k \rightarrow \mathbb{N}$ una biyección recursiva y sea $f(n) = (f_{1(n)}, \dots, f_{k(n)})$ la función inversa, muestre que $f(n)$ es recursiva.

- 5.* Sea $r_1: \mathbb{N}^1 \rightarrow \mathbb{N}$ una biyección fija, demuestre que $f: \mathbb{N}^k \rightarrow \mathbb{N}^l$ es recursiva si y solo si $r_1 \circ f: \mathbb{N}^k \rightarrow \mathbb{N}$ es recursiva, si y solo si $r_1 \circ f \circ r_k^{-1}: \mathbb{N} \rightarrow \mathbb{N}$ es recursiva. (Esto muestra que en la discusión de funciones recursivas bastará considerar funciones de una variable).
6. Demuestre que las funciones $r_k: \mathbb{N} \rightarrow \mathbb{N}^k$ son p.r. (Ayuda: inducción en k).
- 7.* Sean a_0, a_1, \dots, a_n números naturales arbitrarios, demuestre que las siguientes relaciones son (primitivamente) recursivas:
- $Q(a_0, \dots, a_n) \Leftrightarrow$ la ecuación $a_0 + a_1x_1 + \dots + a_nx_n$.
 - $L(a_0, \dots, a_n) \Leftrightarrow$ la ecuación $a_0 + a_1x_1 + \dots + a_nx^n = 0$ tiene solución x en \mathbb{Z}
- 8.* Halle una fórmula para la función $p(n) = n\text{-ésimo primo}$ que no utilice minimización.

2.5 Conjuntos recursivamente enumerables

Un subconjunto S de \mathbb{N} es *recursivamente enumerable* (re) si $S = \emptyset$ o existe una función recursiva sobreyectiva $f: \mathbb{N} \rightarrow S$, es decir una *enumeración recursiva* de $S: f(0), f(1), \dots$ (con posibles repeticiones). Estos conjuntos corresponden a los conjuntos efectivamente enumerables en el sentido intuitivo.

Lema 1. *Todo subconjunto recursivo de \mathbb{N} es re.*

Demostración. $S = \emptyset$ es re por definición. Si $S \neq \emptyset$ sea $n_0 \in S$ y defina

$$f(n) = \begin{cases} n & \text{si } n \in S \\ n_0 & \text{si } n \notin S \end{cases}$$

esta función es recursiva y enumera a S por el Lema 1 de la sección anterior. ■

Veremos más adelante que el converso del lema anterior no es válido. Hay subconjuntos re de \mathbb{N} que no son recursivos. Sin embargo, bajo ciertas circunstancias, S es recursivo si y solo si es r.e.

Lema 2. *Un subconjunto S de \mathbb{N} es recursivo si y sólo si S y S^c son ambos re.*

Demostración. Si S es recursivo, S^c también y ambos son re por el lema 1. Suponga ahora que S y S^c son re y sean f, g enumeraciones recursivas de S y S^c , respectivamente, (dejamos el caso $S = \emptyset$ o $S = \mathbb{N}$ al lector). Defina

$$h(n) = \begin{cases} f\left(\left[\frac{n}{2}\right]\right) & \text{si } n \text{ es par} \\ g\left(\left[\frac{n}{2}\right]\right) & \text{si } n \text{ es impar} \end{cases}$$

que es recursiva. Como $\mathbb{N} = S \cup S^c = \{f(0), g(0), f(1), g(1), \dots\} \cup \{h(0), h(1), h(2), \dots\}$ la función $r(n, x) = |h(n) - x|$ es regular para n. Sea C_P la función características de los pares, entonces la característica de S está dada por

$$C_S(x) = C_P(\mu n (|h(n) - x| = 0)) \blacksquare$$

La demostración anterior muestra que el algoritmo que consiste en buscar x en la lista $f(0), g(0), f(1), g(1), \dots, f(n), g(n), \dots$ y determinar si está en la imagen de f ó en la de g es calculable por una función recursiva.

Lema 3. *Sea S un subconjunto infinito de \mathbb{N} , entonces S es recursivo si y solo si existe una enumeración recursiva estrictamente creciente de S.*

Demostración. Sea C_S la función característica de S. Como S es infinito la función

$$h(x, y) = \overline{\text{Sg}}(C_S(y)(y \dashv x)) = \begin{cases} 0 & \text{si } y \in S \wedge y > x \\ 1 & \text{de lo contrario} \end{cases}$$

es regular, por lo tanto podemos definir $\mu y(h(x, y) = 0)$ y la función

$$f(0) = \mu y (|C_S(y) - 1| = 0) = \mu y (y \in S)$$

$$f(n+1) = \mu y (h(f(n), y) = 0) = \mu y (y \in S \wedge y > f(n))$$

es recursiva y enumera a S.

Para el converso, suponga que f es una enumeración recursiva estrictamente creciente de S, entonces $g(x) = \mu n (f(n) \geq x)$ es recursiva y la función característica de S está dada por :

$$C_S(x) = \overline{\Sigma g} | f(\mu n (f(n) \geq x)) - x|$$

ya que si $f(\mu n (f(n) \geq x)) = x$, obviamente $x \in S$, y si $f(\mu n (f(n) \geq x)) \neq x$ entonces tenemos $f(0), \dots, f(n-1) < x$ y $f(m) \geq f(n) > x$ para $m \geq n$, por lo tanto $x \notin S$. ■

Miremos ahora las propiedades de clausura de los conjuntos recursivamente enumerables.

Lema 4. Sean $S, S' \subseteq \mathbb{N}$ conjuntos re entonces $S \cap S'$ y $S \cup S'$ son re.

Demostración. Sean f, g enumeraciones recursivas de S y S' respectivamente, entonces

$$h(n) = \begin{cases} f([\frac{n}{2}]) & \text{si } n \text{ es par} \\ g([\frac{n}{2}]) & \text{si } n \text{ es impar} \end{cases}$$

enumera a $S \cup S'$. Sea $r: \mathbb{N} \rightarrow \mathbb{N} \times \mathbb{N}$ una biyección recursiva, $r(n) = (r_1(n), r_2(n))$. Si $S \cap S' = \emptyset$ no hay nada que probar, si $S \cap S' \neq \emptyset$ tome $n_0 \in S \cap S'$ y defina

$$h(n) = \begin{cases} r_1(n) & \text{si } f(r_1(n)) = g(r_2(n)) \\ n_0 & \text{de lo contrario} \end{cases}$$

Lo que esta función hace es enumerar todas las posibles parejas $f(x_1), g(x_1)$, obviamente $y \in S \cap S'$ si y solo si existe n tal que $f(r_1(n)) = g(r_2(n)) = y$ por lo tanto h enumera recursivamente a $S \cap S'$. ■

En contraste con los conjuntos recursivos, los conjuntos re no son cerrados bajo complementación. De serlo, todo conjunto re sería recursivo por el lema 2.

Por otra parte, los conjuntos re son cerrados bajo *proyecciones* de conjuntos recursivos.

Lema 5. *Sea $S \subseteq \mathbb{N}^k$, $k \geq 2$, un conjunto recursivo entonces*

$$P_1^k(S) = \{x_1 \mid \exists x_1 \dots \exists x_k ((x_1 \dots x_k) \in S)\} \text{ es re.}$$

Demostración. Sea $r : \mathbb{N} \rightarrow \mathbb{N}^k$, $r(n) = (r_1(n), r_2(n), \dots, r_k(n))$ una biyección recursiva, es decir cada r_i es recursiva. Si $P_1^k(S) = \emptyset$ no hay nada que probar, de lo contrario sea $n_0 \in P_1^k(S)$ y defina

$$h(n) = \begin{cases} r_1(n) & \text{si } (r_1(n), r_2(n), \dots, r_k(n)) \in S \\ n_0 & \text{de lo contrario} \end{cases}$$

que enumera recursivamente a $P_1^k(S)$. ■

Ejemplo

(Conjuntos diofantinos). Sean $p(x_1, \dots, x_k)$, $q(x_1, \dots, x_k)$ polinomios con coeficientes en \mathbb{N} , la relación

$$p(x_1, \dots, x_k) = q(x_1, \dots, x_k)$$

se llama una *ecuación diofantina*. Su conjunto solución es primitivamente recursivo. Un conjunto de la forma

$$\{n \mid \exists x_1 \dots \exists x_k (p(n, x_1, \dots, x_k) = q(n, x_1, \dots, x_k))\}$$

Se llama un *conjunto diofantino*. Del lema anterior se sigue todo conjunto diofantino es recursivamente enumerable.

Una famosa conjetura, demostrada por Matijasevich en 1970, en conexión con la solución del 10º problema de Hilbert afirma que todo re es diofantino!

Podemos llamar a un conjunto $S \subseteq \mathbb{N}^k$ *recursivamente enumerable* en caso de que exista una función recursiva $f: \mathbb{N} \rightarrow \mathbb{N}^k$, es decir $f(n) = (f_1(n), \dots, f_k(n))$ donde cada f_i es recursiva, tal que S sea la imagen de f . Por ejemplo cada \mathbb{N}^k sería re. Con esta definición los lemas 1, 2 y 4 siguen valiendo para subconjuntos de \mathbb{N}^k . Una relación k -ádica $R(x_1 \dots x_k)$ es re si el conjunto $\{(n_1, \dots, n_k) \mid R(n_1, \dots, n_k)\}$ es re. De esta manera el lema 5 tiene la siguiente generalización.

Lema 6. Si $K(x_1, \dots, x_k, x_{k+1}, \dots, x_l)$ es una relación re, entonces $T(x_1, \dots, x_k) \Leftrightarrow \exists x_{k+1} \dots \exists x_l K(x_1, \dots, x_k, x_{k+1}, \dots, x_l)$.

Demostración. Sea $f(n) = (f_1(n), \dots, f_k(n), f_{k+1}(n), \dots, f_l(n))$ una enumeración recursiva del conjunto de verdad de R , entonces $h(n) = (f_1(n), \dots, f_k(n))$ enumera recursivamente al conjunto de verdad de T . ■

Ejercicios

1. Sea $S \subseteq \mathbb{N}$ re y $f: \mathbb{N} \rightarrow \mathbb{N}$ una función recursiva, demuestre que $f(S)$ y $f^{-1}(S)$ son re.
2. Demuestre que si S es re infinito, existe una enumeración recursiva de S sin repeticiones.
3. Demuestre que el conjunto siguiente es re: $\{n \mid \text{la ecuación } x^n + y^n = z^n \text{ tiene soluciones mayores que 1}\}$.
4. Sea $.gj (x_1, \dots, x_k, x_{k+1}, \dots, x_l)$ una fórmula de la aritmética con todos sus cuantificadores acotados demuestre que la relación definida por $Q_1 x_1 \dots Q_k x_k .gj (x_1 \dots x_l)$, donde $Q_i x_i$ es $\exists x_i$ ó $\forall x_i \leq y_i$ es re.
- 5.* Demuestre que los lemas 1, 2, 4 valen para subconjuntos re de \mathbb{N}^k .
- 6.* Demuestre que si $K(\bar{x}, \bar{y})$ es una relación re, lo mismo es cierto de $\forall y \leq z R(\bar{x}, y)$.

- 7.* Sea $f: \mathbb{N}^k \rightarrow \mathbb{N}$ arbitraria muestre que $G(f)$ es recursivo si y sólo si $G(f)$ es re.
- 8.* Sea $R(\vec{x})$ una relación. Si existen relaciones recursivas $R_1(\vec{x}, y_1 \dots y_k)$, $R_2(\vec{x}, z_1 \dots z_l)$ tales que $R(\vec{x}) \Leftrightarrow \exists y_1 \dots \exists y_k R_1(\vec{x}, y_1 \dots y_k)$ y $R(\vec{x}) \Leftrightarrow \forall z_1 \dots \forall z_l R_2(\vec{x}, z_1 \dots z_l)$ entonces $R(\vec{x})$ es recursiva

2.6 Funciones recursivas parciales

Sea $f(\vec{x}, y)$ una función recursiva no necesariamente regular para y , entonces la función

$$h(\vec{x}) = \mu y (f(\vec{x}, y) = 0)$$

no está definida para x en caso de que no exista n tal que $f(\vec{x}, n) = 0$. Sin embargo, h es una función parcial calculable en el siguiente sentido : si tenemos un algoritmo para calcular a $f(\vec{x}, y)$, entonces el algoritmo natural que consiste en calcular sucesivamente :

$$f(\vec{x}, 1), f(\vec{x}, 2), f(\vec{x}, 3), \dots$$

hasta obtener el primer 0, permite calcular $\mu n (f(\vec{x}, n) = 0)$ si tal n existe, y continúa sin término si no existe. La anterior observación nos lleva a extender la noción de función recursiva de la siguiente manera.

Definición. Una función es *recursiva parcial* si puede obtenerse a partir de las funciones básicas por un número finito de aplicaciones de composición, recurrencia primitiva y minimización de funciones totales.

Note que las funciones recursivas parciales incluyen a las totales, a pesar de su nombre. Composición de funciones *estRICTAMENTE PARCIALES*, es decir que no son totales, es en general estrictamente parcial, porque $h(\vec{x}) =$

$f(g_1(\vec{x}), \dots, g_k(\vec{x}))$ está definida en \vec{n} si y solo si $g_1(\vec{n}), \dots, g_k(\vec{n})$ están definidas y esta k -tupla pertenece al dominio de h . Pero puede suceder perfectamente que f no sea total y $f(g_1(\vec{x}), \dots, g_k(\vec{x}))$ sí lo sea. Igualmente, una función $h(\vec{x}, y)$ definida por recurrencia de funciones parciales $f(\vec{x})$ y $g(\vec{x}, y, z)$ está definida en (\vec{x}, n) si y solo si $h(\vec{x}, 0) = f(\vec{x})$, $h(\vec{x}, 1), \dots, h(\vec{x}, n-1)$ están todas definidas; y puede darse el caso de que h sea total aunque g no lo sea.

La minimización se restringe solamente a funciones totales pues si la situación es por ejemplo

$$f(\vec{x}, 0) = 1$$

$$f(\vec{x}, 1) = 3$$

$$f(\vec{x}, 2) \text{ indefinida}$$

$$f(\vec{x}, 3) = 0$$

entonces μ y $(f(x, y) = 0) = 3$ pero el procedimiento descrito arriba para calcular este número no tiene término, pues se queda empantanado en el cálculo de $f(\vec{x}, 2)$. Podemos extender la minimización a funciones parciales si acordamos que en casos como éste μ y $(f(x, y) = 0)$ queda indefinida; ésto equivale a definir:

$$\mu^* y (f(\vec{x}, y) = 0) = \mu y (\prod_{i=0}^y f(\vec{x}, i) = 0)$$

Cuando escribimos de dos funciones recursivas parciales $f(x) = g(x)$ debe entenderse que f y g ambas están definidas en x , y $f(x) = g(x)$, o tanto f como g están indefinidas en x .

Veremos más adelante que las funciones recursivas parciales no son simplemente restricciones de recursivas, pues hay funciones recursivas estrictamente parciales que no pueden extenderse a una función recursiva total. Por otra parte el dominio de una función recursiva parcial no puede ser cualquier conjunto pues también demostramos que debe ser recursivamente

enumerable (Cap. 5, Sec. 5.4). Por ahora nos contentamos con la siguiente observación.

Lema. *Todo conjunto r.e. es el dominio de una función recursiva parcial.*

Demostración. Sea $E \subseteq \mathbb{N}^k$ un conjunto r.e. enumerado por una función recursiva $f : \mathbb{N} \rightarrow E$, digamos $f(n) = (f_1(n), \dots, f_k(n))$.

Sea $h : \mathbb{N}^k \rightarrow \mathbb{N}$ la función

$$h(x_1, \dots, x_k) = \mu y (|f_1(y) - x_1| \dots |f_k(y) - x_k|) = 0$$

entonces $(x_1, \dots, x_k) \in E$ si y solamente si $(x_1, \dots, x_k) \in \text{Dom}(h)$. ■

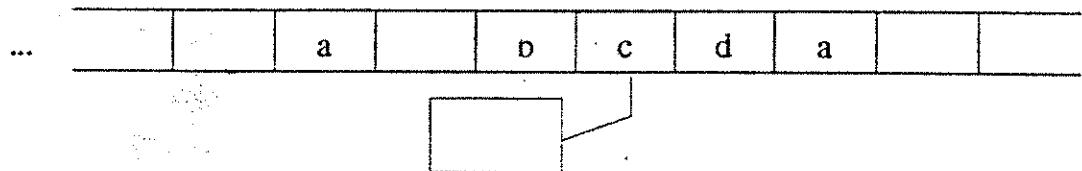
Máquinas de Turing

3.1 Máquinas de Turing

Hemos afirmado en la sección anterior que toda función recursiva es calculable por medio de un algoritmo. En general, se puede afirmar que ciertos resultados se pueden realizar algorítmicamente. Pero ¿qué es un algoritmo? Hasta ahora hemos usado una noción informal explicada por medio de ejemplos y de ciertos métodos generales que aceptamos como algorítmicos. Esta noción es suficiente para obtener resultados positivos de calculabilidad. Si se quiere mostrar que algún proceso es calculable basta describir un algoritmo concreto (por ejemplo, un programa, un diagrama de flujo, etc.) que lo calcule. Sin embargo, para poder demostrar resultados de *insolubilidad algorítmica*, es decir la no calculabilidad por medio de algoritmos de ciertas funciones o procesos, necesitamos una definición precisa de algoritmo que abarque todos los casos imaginables y pueda ser analizado matemáticamente. En 1936, el matemático inglés Alan Turing propuso tal definición, en la forma de autómatas capaces de calcular sobre expresiones simbólicas. Estas son las hoy llamadas *máquinas de Turing*, que pueden considerarse como versiones idealizadas de las computadoras digitales modernas, cuando éstas se cargan con un programa fijo. Sin embargo, Turing concibió su idea antes de que se desarrollaran las computadoras.

Sea Σ un alfabeto finito y considere una *cinta* dividida en celdas que pueden estar en blanco o contener un símbolo de Σ . La cinta es potencialmente

infinita, es decir, aunque no sea infinitamente larga, es posible añadirle celdas cada vez que sea necesario:



Una máquina de Turing es un autómata que "calcula" sobre la cinta leyendo y alternando el contenido de las celdas (una a la vez) moviendo su cabeza lectora de una a otra celda de acuerdo con ciertas instrucciones prefijadas. Si en un momento dado la cabeza se sale del extremo derecho o izquierdo de la cinta, se supone que se añade automáticamente una celda en blanco al extremo de la cinta bajo la cabeza lectora. La actividad de la máquina debe estar completamente determinada por el símbolo que esté leyendo y su "estado" interno en un momento dado. Para poder dar una definición formal introducimos dos símbolos auxiliares llamados *movimientos*

R "derecha"

L "izquierda"

y el símbolo \square llamado *blanco* que se supone no pertenece a Σ .

Definición Sea Σ un alfabeto finito. Una *máquina de Turing T sobre Σ* es una tripla $T = (K, q_0, I)$ en donde:

- (a) $K = \{q_0, q_1, \dots, q_n\}$ es un conjunto finito cuyos elementos se llaman (*a*) *estados* de T .
- (b) $q_0 \in K$ es el *estado inicial* de T .
- (c) I es una función de un subconjunto de $K \times (\Sigma \cup \{\square\})$ en $(\Sigma \cup \{\square\}) \times \{R, L\} \times K$.

Ejemplo 1

Sea $\Sigma = \{1, 0, a\}$ entonces la siguiente tabla para I determina una máquina de Turing sobre Σ , con estados $K = \{q_0, q_1, q_2, q_3\}$ y estado inicial q_0 :

I	1	0	a	\square
q ₀	aRq ₁	0Rq ₀	--	--
q ₁	1Rq ₁	0Rq ₁	--	\square Rq ₂
q ₂	1Rq ₂	--	--	1Lq ₃
q ₃	1Lq ₃	0Lq ₃	1Rq ₀	\square Lq ₃

La función I está completamente determinada por su grafo, es decir las quíntuplas de la forma:

$$q_i \xrightarrow{s,t} M q_j$$

con $q_i, q_j \in K$, $s, t \in \Sigma \cup \{\square\}$ y $M \in \{R, L\}$, tales que $I(q_i, s) = (t, M, q_j)$. Llamaremos *instrucciones* de T a tales quíntuplas. El ejemplo anterior queda:

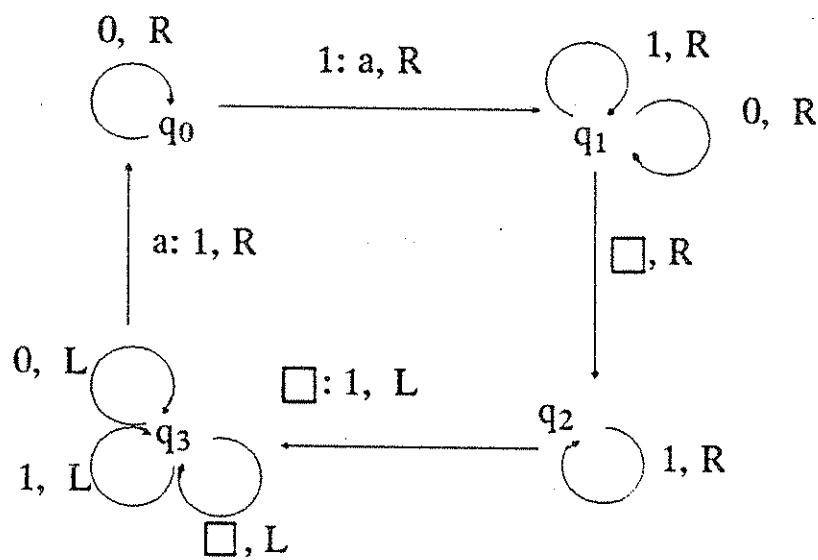
- (1) $q_0 1 a R q_1$
- (2) $q_0 0 0 R q_0$
- (3) $q_1 1 1 R q_1$
- (4) $q_1 0 0 R q_1$
- (5) $q_1 \square \square R q_2$
- (6) $q_2 \square 1 L q_3$
- (7) $q_2 1 1 R q_2$
- (8) $q_3 \square \square L q_3$
- (9) $q_3 0 0 L q_3$
- (10) $q_3 1 1 L q_3$
- (11) $q_3 a 1 R q_0$

Podemos suponer que la máquina de Turing T (K, q_0, I) es un autómata que puede asumir un número finito de configuraciones o estados internos (por ejemplo las posibles configuraciones de los circuitos electrónicos o del "hardware" de una máquina computadora, el número de éstos relevantes a la computación es inmensamente grande pero finito), posee una cabeza que

puede estar sobre cualquiera de las celdas de la cinta, y cuando está *activa* altera el símbolo de la celda que esté leyendo, se mueve a una celda vecina y cambia de estado interno, de acuerdo a la siguiente interpretación de sus instrucciones:

- $q_i s t R q_j$: Si está en el estado q_i leyendo el símbolo s , entonces cambie el símbolo s por t , muévase una celda a la derecha y pase al estado q_j . El símbolo \square se interpreta como celda en blanco.
- $q_i s t L q_j$: Se interpreta análogamente, pero el movimiento es una celda a la izquierda.

El funcionamiento de la máquina de Turing se explica gráficamente por un *diagrama de transacción* de estados. Del ejemplo 1 obtenemos el diagrama:



Aquí, s:t,R significa: cambie el símbolo s por t y muévase a la derecha;
s,R significa: muévase a la derecha (sin alterar el símbolo leído).
 Análogamente para s:t,L ó s,L, con la diferencia de que ahora el movimiento es a la izquierda.

Note que para cada pareja $q_i s \in K \times \Sigma \cup \{\square\}$ debe haber a los sumas una instrucción que comienza con $q_i s$. Además, puede haber parejas $q_i s$ que no aparecen al comienzo de ninguna instrucción (como $q_2 a$, $q_1 \square$, en el ejemplo anterior). Estas se llaman *combinaciones finales* pues al llegar a leer el

símbolo s mientras esté en el estado q_i , la máquina se detiene, y ésta es la única forma como puede detenerse.

Ejemplo 2

Supongamos que se trata de la máquina del ejemplo 1 y que en cierto momento esté en el estado q_0 leyendo el símbolo más a la izquierda de la palabra

11010011

al ejecutar las instrucciones pasa sucesivamente por las configuraciones siguientes (indicamos el estado por subíndice debajo de la celda que se está leyendo):

	/	/	0	/	0	0	/	/		(*)
0										
a	/	0	/	0	0	/	/			

1

En el estado q_1 se mueve a la derecha dejando intactos ceros y unos hasta llegar al primer blanco:

a	/	0	/	0	0	/	/			

1

al leer blanco en el estado q_1 se mueve una celda más a la derecha:

a	/	0	/	0	0	/	/			

2

En el estado q_2 al leer blanco imprime 1 y se devuelve pasando al estado q_3 :

a	/	0	/	0	0	/	/			

3

En el estado q_3 se mueve a la izquierda hasta hallar el símbolo a:

	a	/	0	/	0	0	/	/	/	/	
--	---	---	---	---	---	---	---	---	---	---	--

Repone el 1 vuelve al estado q_0 que busca el siguiente 1 a la derecha:

Se reinicia el proceso, marcando el segundo 1 con a:

Se copia el 1 en el segundo blanco de la derecha:

/ a 0 / 0 0 / / / / / /

y, como antes retorna a la situación inicial, ahora q₀ leyendo el tercer 1:

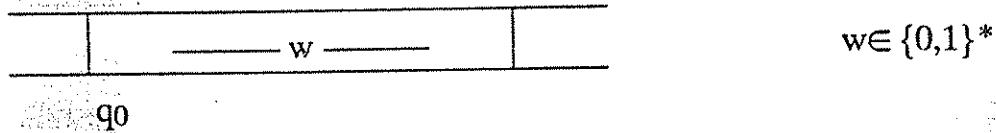
De esta manera se van copiando los unos de la palabra original en el extremo derecho (cuando q_0 lee 0, se lo salta a la derecha). Inmediatamente después de copiar el último 1 se tiene

	/	/	0	/	0	0	/	a		/	/	/	/	/	/	/
--	---	---	---	---	---	---	---	---	--	---	---	---	---	---	---	---

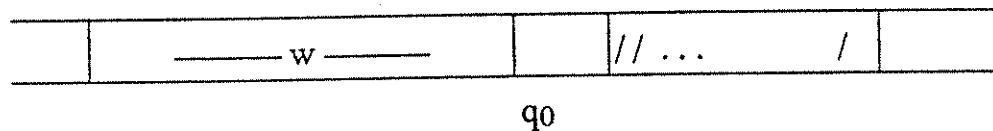
después de 7 pasos más:

	/	/	0	/	0	0	/	a		/	/	/	/	/	/
								3							
	/	/	0	/	0	0	/	/		/	/	/	/	/	/
								0							

Como no hay instrucción que comience en $q_0 \square$ la máquina se detiene en esta configuración. Podemos afirmar en general, si se comienza con la configuración



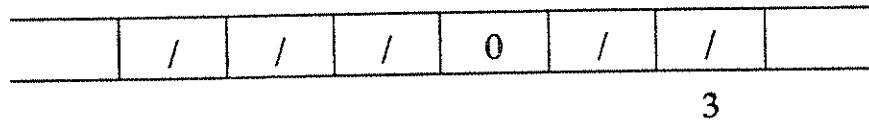
la máquina sólo se detiene al alcanzar la configuración



después de escribir a la derecha de W tantos unos seguidos, como unos tiene W.

Ejemplo 3

En algunos casos puede suceder que la máquina nunca termine su cálculo, si se inicializa la máquina del ejemplo anterior en la configuración:



esta permanecerá en el estado q_3 moviéndose eternamente a la izquierda. Otro ejemplo más interesante se obtiene reemplazando en la máquina anterior, la instrucción (5) por

$$q_1 \square 0 R q_2$$

entonces al inicializarse



producirá:

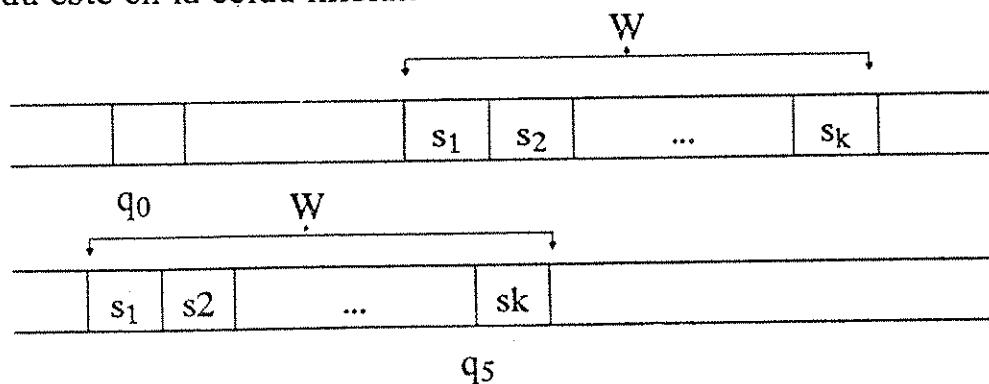
	a			
	1			
	a	0		
			2	
	a	0	/	
			3	
	/	0	/	
			0	
	/	0	/	
			0	

Y después, periódicamente, sin detenerse nunca:

	/	0	/	0	/	
					0	
	/	0	/	0	/	0
						0
	/	0	/	0	/	0
						0
	/	0	/	0	/	0
						...

Ejemplo 4

Trasladar una palabra en la cinta. La máquina siguiente es tal que si se inicializa en una celda arbitraria a la izquierda del comienzo de una palabra $W \in \{O, /\}^*$, traslada la palabra de manera que su primer símbolo a la izquierda esté en la celda inicial:



La máquina usa símbolos auxiliares a,b

	0	1	a	b	\square
q ₀				aRq ₄	aRq ₁
q ₁	bLq ₂		bLq ₃	bRq ₄	\square Rq ₁
q ₂			0Rq ₀	bLq ₂	\square Lq ₂
q ₃			1Rq ₀	bLq ₃	\square Lq ₃
q ₄	bLq ₂	bLq ₃		bRq ₄	\square Lq ₅
q ₅			\square Lq ₅	\square Lq ₅	\square Lq ₅

Indicamos el papel de cada estado:

q₀: Marca con la celda en que se debe copiar el próximo símbolo, luego pasa control a q₁.

q₁, q₄: q₁ Se mueve a la derecha saltando \square y buscando el primer 0, 1 ó b, si encuentra b pasa el control a q₄ que busca 0 ó 1 saltando bees. Cuando q₁ ó q₄ encuentran 0 ó 1 lo marcan b. Si es 0 pasan el control a q₂. Si es 1 pasan el control a q₃. Si q₄ encuentra \square antes de 0, 1 pasa control a q₅.

q₂: Retorna a la izquierda saltando b y \square y buscando a. Al encontrar a la cambia a 0, se mueve a la derecha y pasa control a q₀.

q₃: Retorna a la izquierda igual que q₂, buscando a, al encontrarlo la cambia a 1, se mueve a la derecha y pasa control a q₀.

q₅: Se mueve a la izquierda borrando bees, saltando \square y borrando las aes sobrantes.

Sería posible construir una máquina que traslade W a cualquier posición inicial, ya sea a la izquierda, la derecha o dentro de W misma.

Ejercicio

Describa una máquina de Turing que al inicializarse en el estado q₀ leyendo un símbolo interior de una palabra w, traslada la palabra de manera que su símbolo inicial esté en la posición inicial de q₀.

3.2 Ejemplos

Continuamos dando ejemplos de máquinas de Turing que realizan ciertas importantes operaciones básicas.

Ejemplo 5

Sumar 1 en notación decimal. La máquina se inicializa en q_0 leyendo el primer dígito a la izquierda de un número n en notación decimal. Se detiene produciendo la notación decimal de $n+1$, leyendo la celda anterior a su primer dígito.

	0	1	2	3	4	5	6	7	8	9	\square
q_0	0R q_0	1R q_0	2R q_0	3R q_0	4R q_0	5R q_0	6R q_0	7R q_0	8R q_0	9R q_0	$\square Lq_1$
q_1	1L q_2	2L q_2	3L q_2	4L q_2	5L q_2	6L q_2	7L q_2	8L q_2	9L q_2	0L q_1	1L q_2
q_2	0L q_2	1L q_2	2L q_2	3L q_2	4L q_2	5L q_2	6L q_2	7L q_2	8L q_2	9L q_2	—

Indicamos la actividad de la máquina en cada estado:

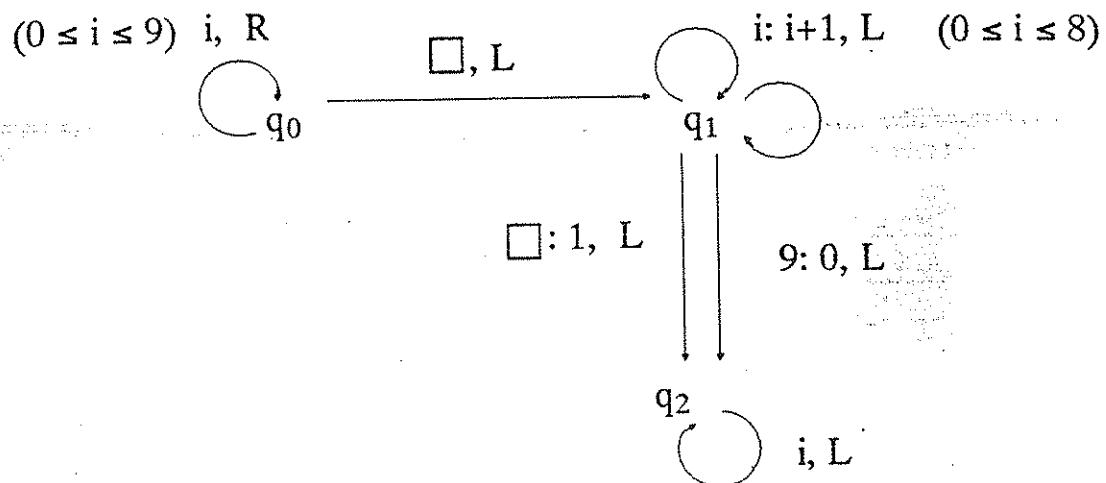
q_0 : Busca final de la palabra, pasa el control al estado leyendo el último dígito.

q_1 : Suma 1 al dígito que está leyendo y se mueve a la izquierda, si el dígito no es 9 pasa al estado q_2 , si es nueve la suma es 0 y se mantiene en el estado q_1 .

q_2 : Se mueve a la izquierda hasta el primer blanco, ahí se detiene. Aunque el estado q_2 es innecesario será útil en el próximo ejemplo.

q_5 : Se mueve a la izquierda borrando bees, saltando \square y borrando las aes sobrantes.

Note que después de terminar la suma, la máquina se detiene en la combinación $q_2 \square$ que es la única combinación final. Si $n = 9\ 9\dots 9$ en notación decimal, el resultado final aparecerá una celda corrida a la izquierda:



	9	9	...	9	
0					

	1	0	0	...	0	
2						

Es posible modificar la máquina de manera que el resultado comience en la posición en que comenzaba el dato original. Basta substituir la instrucción $q_1 \square 1 \mid L q_2$ por $q_1 \square \square R q_3$ y añadir estados e instrucciones adicionales:

$q_3 0 1 R q_4$

$q_4 0 0 R q_4$

$q_4 \square 0 R q_5$

Ejemplo 6

Cambiar de notación monádica a notación decimal. Sean

$$\bar{n} = //\dots / (n+1 \text{ veces})$$

$$\bar{n}^{10} = n \text{ en notación decimal}$$

La siguiente máquina hace la transformación $\bar{n} \mid \bar{n}^{10}$, si se inicializa en q_0 al extremo izquierdo de \bar{n} . Esto se puede obtener sumando n veces 1 a 0 en notación decimal. El dato inicial \bar{n} se usa como contador. Se marca el primer palote de \bar{n} y los demás se eliminan sucesivamente; por cada palote eliminado se suma 1 en decimal a una expresión que aparece a la derecha y es inicialmente 0. Para realizar la suma se usa la máquina del ejemplo anterior como *subrutina*. Tenemos:

$$\Sigma = \{/ , 0, 1, 2, \dots, 9\}$$

$$K = \{q_0', \dots, q_5', q_0, q_1, q_2\}$$

Es estado inicial q_0' (también es final).

	/	0	1	2	...	9	\square
q_0'	$0Rq_1'$						
q_1'		$/Rq_1'$					$\square Rq_2'$
q_2'							$0Lq_3'$
q_3'	$\square Rq_4'$	$\square Rq_0'$					$\square Lq_3'$
q_4'	$/Lq_5'$	$0Lq_5'$	$1Lq_5'$	$2Lq_5'$...	$0Lq_5'$	$\square Rq_4'$
q_5'							$\square Rq_0$
q_0							----- como en ejemplo 5
q_1							-----
q_2							$\square Lq_3'$

Las líneas punteadas denotan las instrucciones de la máquina del ejemplo 5. Los estados q_0' , q_1' y q_2' se usan para marcar 0 al comienzo de \bar{n} e imprimir 0 después de un blanco al extremo derecho de \bar{n} . El estado q_3' se devuelve a buscar el último palote de \bar{n} , lo borra y pasa control al estado q_4' . Entre q_4' y q_5' se busca el comienzo de la expresión decimal a la derecha de \bar{n} , y al encontrarlo se pasa el control al q_0 que es el estado inicial de la

máquina que suma 1 en decimal. Al terminar la suma la máquina queda en la configuración:

	0	/ ... /			d1	d2	...
					2		

La instrucción $q_2 \square \square L q_3'$ devuelve el control al estado q_3' que reinicia el proceso de borrar palotes y sumar 1 a la derecha. Cuando q_3' no encuentra más palotes sino la marca 0, la borra y pasa a q_0' , quedando la máquina en la combinación final $q_0' \square$.

Ejemplo 7

Multiplicación en monádico. Damos una máquina de Turing que multiplica en notación de palotes. $\Sigma = \{b, /\}$, $K = \{q_0, \dots, q_8\}$. Si comienza en la configuración:

	/ ... /		/ ... /	
	↑ m palotes		n palotes	

q_0

Termina en la configuración:

	/ ... /	

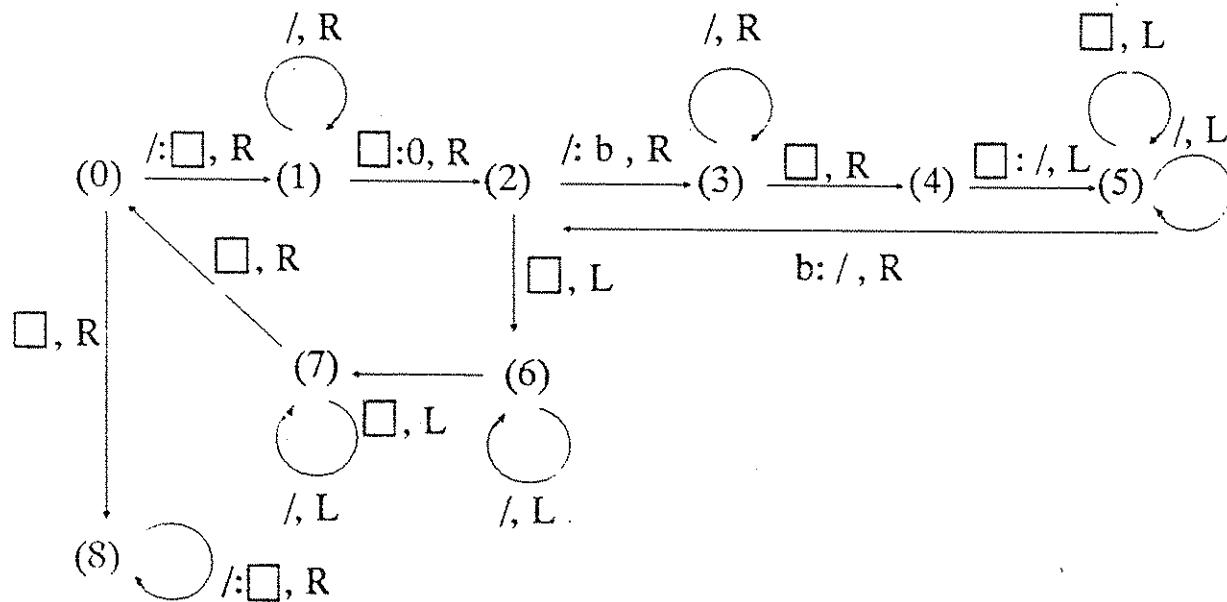
mn palotes

La idea es usar, igual que en ejemplo 6, a m como contador y copiar a n, m veces seguidas.

$q_0 / \square R q_1$ $q_1 / / R q_1$ $q_1 \square \square R q_2$	borra / de m y busca \square a la derecha
---	---

- $q_2 / b \text{ R } q_3$
 $q_3 / / \text{ R } q_3$
 $q_3 \square \square \text{ R } q_4$
 $q_4 / / \text{ R } q_4$
 $q_4 \square / \text{ L } q_5$
- $q_5 / / \text{ R } q_4$
 $q_5 \square \square \text{ L } q_5$
 $q_5 b / \text{ R } q_2$
- $q_2 \square \square \text{ L } q_6$
 $q_6 / / \text{ L } q_6$
 $q_6 \square \square \text{ L } q_7$
 $q_7 / / \text{ L } q_7$
 $q_7 \square \square \text{ R } q_0$
- $q_0 \square \square \text{ R } q_8$
 $q_8 / \square \text{ R } q_8$
- marca con b palote de n y va a copiarlo al extremo derecho
- se devuelve a b, repone /, y pasa control a q_2 en la celda siguiente a la derecha, para reiniciar el ciclo de copiado de siguiente / de n
- Terminó de copiar n, vuelve a buscar primer palote no borrado de m, al hallarlo pasa control a q_0 para reiniciar ciclo de copiado de n
- m ha sido borrado (o sea: n ha sido copiado m veces seguidas), borra n y para

Su diagrama sería:



Dejamos al lector modificar la máquina para que realmente transforme \bar{m} , \bar{n} , en mn , es decir $m + 1$ palotes $n + 1$ palotes en $mn + 1$ palotes.

Ejemplo 8

Cambiar notación decimal a monádica. Sea $m^{-10} = d_0 d_1 \dots d_n$ una expresión decimal con $d_i \in \{0, \dots, 9\}$. Entonces:

$$m = d_0 10^n + d_1 10^{n-1} + \dots + d_{n-1} 10 + d_n.$$

Para calcular n en monádica, se generan sucesivamente A_i palotes donde

$$A_0 = d_0$$

$$A_1 = d_0 10 + d_1$$

$$A_2 = (d_0 10 + d_1) + d_2$$

:

:

:

$$A_n = ((\dots(d_0 10 + d_1) 10 + d_2) 10 + d_3) 10 + \dots + d_{n-1}) 10 + d_n.$$

Claramente, $A_{k+1} = A_k 10 + d_k$ y $A_n = m$.

Describimos la máquina informalmente y dejamos al lector llenar los detalles.

La configuración inicial es:

$$\bar{n}^{-10}: \quad \boxed{ \mid d_0 d_1 \dots d_n \mid}$$

Paso 1. Pone marca a la izquierda de \bar{n}^{-10} . Salta la palabra \bar{n}^{-10} a la derecha y después de dejar *once* blancos imprime un 0

	0 1 ...			12	
X	$d_0 d_1 \dots d_n$			0	
				↑	

Paso 2. Retorna a buscar marca X, al encontrarla la bora y la mueve al dígito siguiente. Si éste es de pasa a un estado especial q_d ($0 \leq d \leq 9$). En este estado se mueve a la derecha, deja un blanco e imprime d palotes:

	0	1 ...	d_0	...	12	
	X	$d_1 \dots d_n$	/	...	/	0

↑

Si después de localizar la marca X, no encuentra dígito a la derecha (encuentra blanco), salta al **Paso 5**. De lo contrario continúa en el paso siguiente.

Paso 3. Repite 10 veces la expresión en palotes que aparece a partir de la celda 12. si lo que aparece allí es un cero, lo cual sucede en la primera vuelta, simplemente lo borra

	0	1 ...	d_0	...	12	
	X	$d_1 \dots d_n$	/	...	/	

↓

Paso 4. Añade a los palotes que comienzan en la celda 12 los d_0 palotes que comienzan en la celda 1, de manera que el resultado total comience en la celda 12:

	0	1 ...	d_0	...	12	
	X	$d_1 \dots d_n$	/	...	/	/.../

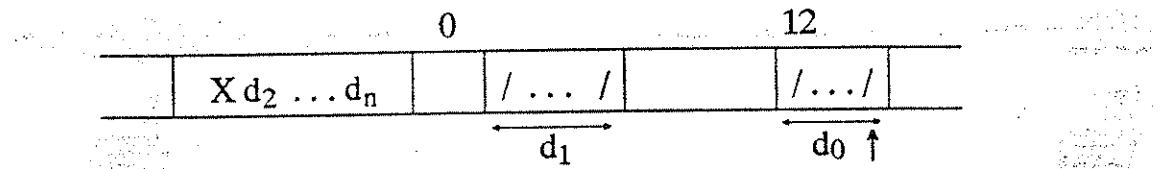
d_0 ↑

Luego vuelve al **Paso 2**.

Paso 5. Borra X, añade un palote a la expresión monádica y se detiene.

Indicamos la continuación del proceso cuya iniciación se representó en las cuatro figuras anteriores.

P2



P3

	$X d_2 \dots d_n$		/ ... /		/ ... /
		d_1		d_010	

P4

	$X d_2 \dots d_n$			/ ... /	
			d_1	$d_010 + d_1$	

P2

	$X d_3 \dots d_n$		/ ... /		/ ... /	
		d_2		$d_010 + d_1$		

P3

	$X d_3 \dots d_n$		/ ... /		/ ... /	
		d_2		$(d_010 + d_1)10$		

P4

	$X d_3 \dots d_n$		/ ... /		/ ... /	
		d_2		$(d_010 + d_1)10 + d_2$		

Se continúa de esta manera hasta marcar el último dígito:

P2

			0			12	
	X		/ ... /			/ ... /	
			d_n			A_{n-1}	

P3

	X		/ ... /		/ ... /	
			d_n		$A_{n-1}10$	

P4

	X				/ ... /	
					$A_{n-1}10 + d_n = m$	

P5

			\overline{m}		
			/ ... /		
				$m + 1$	

Si se desea es posible diseñar la máquina de Turing de manera que el resultado final se trasladea a la izquierda a la posición inicial de la expresión decimal, usando la máquina del ejemplo 4.

Es claro que podemos construir máquinas de Turing que traduzcan de notación en base p , $\overline{n^p}$, a notación monádica y viceversa, de la misma manera que lo hicimos para base 10. Concatenando las combinaciones finales de una máquina que traduzca de base p a monádico con el estado inicial de una que traduzca de monádico a base q , tendremos una máquina que calcula el cambio de base:

$$\overline{n^p} \rightarrow \overline{n^q}$$

Ejercicios

1. Dé máquinas de Turing que realicen las operaciones siguientes.
 - a. Suma de dos números en notación decimal (separados por \square)
 - b. Traduzca de palotes a notación binaria.
 - c. Invierta palabras de $\{0, 1\}^*$
 - d. Calcule la función característica de las expresiones decimales divisibles por 9.
 - e. Calcule $f(x,y) = x^y$ en notación de palotes.
2. Dé en detalle una máquina de Turing que realice las operaciones del ejemplo 8 (pasar de decimal a palotes)
3. Dé una máquina que traduzca de notación decimal a binaria.
4. Suponga que hay un solo 1 impreso en alguna celda de la cinta. Describa una máquina de Turing tal que si se inicializa en el estado q_0 en una celda arbitraria de la cinta, busque el 1 y se detenga al encontrarlo. Note que la máquina no puede saber si el 1 está a su izquierda o derecha inicialmente.
5. Dé una máquina de Turing tal que si se le dá una expresión en el alfabeto $\{p, q, r, N, K, A, C, E\}$ dé como resultado 1 si la expresión es una f b f en notación polaca, 0 de lo contrario.
6. Dé una máquina que traduzca de notación "standart" a polaca.
7. Considere máquinas de Turing que no pueden imprimir (o borrar) y moverse simultáneamente, sus instrucciones se pueden representar en la siguiente forma:
$$q_i \ t \ q_j, \quad q_i \ R \ q_j, \quad q_i \ L \ q_j.$$

Demuestre que toda máquina de este tipo es equivalente a una máquina corriente, y toda máquina corriente es equivalente a una de este tipo.

3.3 Turing- Calculabilidad de Funciones recursivas

En esta sección demostraremos que toda Función recursiva es calculable por medio de una máquina de Turing. A cada máquina M se puede asociar una Función de Σ^* en Σ^* de la manera siguiente. Suponga que $W \in \Sigma^*$ está impresa en la cinta símbolo por símbolo y la máquina se inicializa en el estado q_0 leyendo el primer símbolo a la izquierda (la palabra vacía se identifica con una celda en blanco). Si después de un número infinito de pasos la máquina se detienen en alguna combinación final entonces:

$$F_M(W) = \begin{cases} \text{Sucesión de símbolos de la configuración final} \\ \text{tal que se extiende desde la celda que la máquina} \\ \text{ha quedado leyendo al detenerse, hasta el primer} \\ \text{blanco de izquierda a derecha} \end{cases}$$

Si M no se detiene, $F_M(W)$ queda indefinida. El *dominio* de M es el conjunto:

$$\{ W \in \Sigma^*: F_M(W) \text{ está definido}\}.$$

Dos máquinas de Turing M y M' son *equivalentes* si $F_M = F_{M'}$, es decir la dos funciones tienen el mismo dominio y cuando $W \in D$ entonces $F_M(W) = F_{M'}(W)$. El siguiente resultado muestra que no se pierde generalidad si se supone que las máquinas de Turing trabajan solamente en la mitad derecha de la cinta, o en una *semicinta* que tiene una primera celda izquierda y se extiende a la derecha.

Lema 1 *Para toda máquina de Turing M existe una máquina equivalente M' que nunca visita las celdas a la izquierda de la celda anterior a la inicial.*

Demostración. Sea Σ el alfabeto de M . Formamos un nuevo alfabeto $\Sigma' = \Sigma \cup \{B\}$ donde B es un símbolo nuevo, el "blanco ficticio". Construimos M' en varios pasos.

- (1) Sea \overline{M} una máquina que hace exactamente lo mismo que M con la diferencia que marca con B cada celda en blanco visitada. Es decir, \overline{M} imprime B cuando M borraría y actúa al leer B como M actuaría al leer \square . Para obtenerla basta:

- (i) Cambiar cada instrucción $q_i s \square M q_j$ de M por $q_i s B M q_j$.
(ii) Añadir a cada instrucción $q_i \square s M q_j$ de M la nueva instrucción $q_i B s M q_j$.
- (2) La siguiente máquina T traslada cualquier palabra sobre el alfabeto Σ' una celda a la derecha. Tiene estados q_0, q_1, q_2 y un estado q^s para cada $s \in \Sigma'$. Comienza y termina en las configuraciones indicadas:

inicial:	W	
	q_0	
final:	W	
	q_2	

Instrucciones:

$$\begin{array}{ll}
q_0 s \square R q^s & \text{para cada } s \in \Sigma' \\
q^s t s L q^t & s, t \in \Sigma' \\
q^s \square s L q_1 & s \in \Sigma' \\
q_1 s s L q_1 & s \in \Sigma' \\
q_1 \square \square R q_2 & \\
q_0 \square \square R q_2 &
\end{array}$$

- (3) Sea ahora M' una máquina que hace lo mismo que \bar{M} con las siguientes diferencias:

- (i) Antes de comenzar la ejecución mueve toda la información a la derecha una posición y marca la inicial con $*$, un símbolo nuevo, luego inicia la ejecución en la segunda celda, es decir la inicial de la información:

	s_1	s_2		...
	q_0			
	$*$	s_1	s_2	...

q_0'

Esto puede hacerse modificando ligeramente la máquina T explicada en (2).

- (ii) Cada vez que visite la celda inicial marcada con * corre toda la información una celda a la derecha (esta información es una palabra continua en Σ^*). La ejecución en el mismo estado en que leyó *, pero leyendo el blanco que queda a la derecha de *:

	*	s_1	s_2	...	
q_i					
	*		s_1	s_2	...

q_i

Esto se logra añadiendo por cada estado q_i de M la instrucción

$$q_i \xrightarrow{*} R q_o^i$$

y una copia T_i de la máquina trasladadora T, con estados q_0^i, q_1^i, q_i en lugar de q_0, q_1, q_2 .

- (4) Añada instrucciones y estados para que la máquina después de haber terminado la ejecución (si se detiene) borre todos los B's y la *, deteniéndose en el mismo estado y posición relativa a la información final en que se habría detenido M. Esto se logra mandando cada configuración final de M, $q_k u$, a una sucesión de instrucciones que marquen la u en que se detuvo la máquina, borren las B's y * y se devuelvan a detenerse en el mismo u. Dejamos al lector los detalles. ■

Definición Una función $F: \mathbb{N} \rightarrow \mathbb{N}$ es T -calculable si existe un alfabeto Σ que contiene /, y una máquina M sobre Σ que calcula a F en notación de palotes, es decir $F_M(\bar{n}) = \overline{F(n)}$ para todo $n \in \mathbb{N}$. En general, $F: \mathbb{N}^k \rightarrow \mathbb{N}$ es T -calculable si existe una máquina cuyo alfabeto contiene /, * y calcula la función:

$$\bar{n}_1 * \bar{n}_2 * \dots * \bar{n}_k \rightarrow \overline{F(n_1, \dots, n_k)}$$

Como existen máquinas de Turing que transforman de notación en base p a notación monádica y viceversa (ejemplos 6 y 7, sección 2.2) es fácil ver que $F: \mathbb{N}^k \rightarrow \mathbb{N}$ es T - calculable si y solo si es calculable por una máquina de Turing en base p, para cualquier $p \geq 1$.

Teorema. *Toda función recursiva (parcial) es T-calculable.*

La demostración de este teorema se hará por medio de una sucesión de lemas, por inducción en la complejidad de la definición de la función recursiva. Es fácil ver que la función $Z(x) = 0$ es calculable, basta borrar la expresión de x y reemplazarla por la de 0. En el ejemplo 5 de la sección 2.2 se mostró como calcular $S(x) = x + 1$ en decimal, en monádico es más sencillo. Las proyecciones son T-calculables. Indicamos una máquina que calcula $P_2^3(x_1, x_2, x_3) = x_2$ en monádico, la generalización a otros valores de i, n o a otras notaciones es obvia:

Configuración inicial:

$$\begin{array}{c} \overline{x_1} * \overline{x_2} * \overline{x_3} \\ \uparrow \\ 0 \end{array}$$

Borra $\overline{x_1} *$:

$$q_0 / \square R q_0$$

$$q_0 * \square R q_1$$

Salta $\overline{x_2}$ y borra $* \overline{x_3}$:

$$q_1 // R q_1$$

$$q_1 * \square R q_0$$

$$q_0 \square \square L q_2$$

Vuelve al comienzo de \bar{x}_2 :

$q_2 \square \square L q_2$

$q_2 // L q_3$

$q_3 // L q_3$

$q_3 \square \square R q_4$.

De esta manera hemos demostrado que todas las funciones básicas son calculables.

Lema 2. Si $f_i(x_1, \dots, x_n)$, $i = 1, \dots, k$, y $g(x_1, \dots, x_k)$ son T-calculables, entonces $h(x_1, \dots, x_n) = g(f_1(x_1, \dots, x_n), \dots, f_k(x_1, \dots, x_n))$ es T-calculable.

*Demuestra*ción. Sean T_1, \dots, T_k máquinas que calculan a f_1, \dots, f_k respectivamente, y T una máquina que calcula a g . Podemos suponer que sus estados son disyuntos y que sólo utilizan la mitad derecha de la cinta (lema 1).

Sean q_o, q_o^t los estados iniciales de T, T_i , $i = 1, \dots, n$. Podemos suponer además que toda máquina termina siempre en el mismo estado final q_f, q_f^t , $i=1, \dots, n$. La máquina que calcula h :

Configuración inicial:

$$\begin{array}{c} \bar{x}_1 * \dots * \bar{x}_n \\ \uparrow \\ q_0' \end{array}$$

(1) Copia dato inicial después de un blanco a la derecha y pasa control a q_o^1 :

$$\begin{array}{c} \bar{x}_1 * \dots * \bar{x}_n \quad \square \quad \bar{x}_1 * \dots * \bar{x}_n \\ \uparrow \\ q_0^1 \end{array}$$

(2) Con las instrucciones de T_1 calcula $f_1(x_1, \dots, x_n)$:

$$\overline{x_1} * \dots * \overline{x_n} \quad \square \quad \overline{f_1(x_1, \dots, x_n)} \\ \uparrow \\ q_f^1$$

Note que por hipótesis T_1 no altera el dato inicial a la izquierda del blanco.

(3) Copia el dato inicial a la derecha y pasa control a q_0^2 :

$$\overline{x_1} * \dots * \overline{x_n} \quad \square \quad \overline{f_1(x_1, \dots, x_n)} \quad \square \quad \overline{x_1} * \dots * \overline{x_n} \\ \uparrow \\ q_0^2$$

(4) Calcula $f_2(x_1, \dots, x_n)$ por medio de T_2 :

$$\overline{x_1} * \dots * \overline{x_n} \quad \square \quad \overline{f_1(x_1, \dots, x_n)} \quad \square \quad \overline{f_2(x_1, \dots, x_n)} \\ \uparrow \\ q_f^2$$

Concatenando de esta manera máquinas $R_1, T_1, R_2, T_2, \dots, R_k, T_k$ donde las R_i son rutinas de copia a la derecha tenemos finalmente:

$$\overline{x_1} * \dots * \overline{x_n} \quad \square \quad \overline{f_1(x_1, \dots, x_n)} \quad \square \dots \quad \square \quad \overline{f_k(x_1, \dots, x_n)} \\ \uparrow \\ q_f^k$$

Añadiendo instrucciones que borren el dato inicial, coloquen '*'s y pasen el control al estado q_0 :

$$\overline{f_2(x_1, \dots, x_n)} * \dots * \overline{f_k(x_1, \dots, x_n)} \\ \uparrow \\ q_0$$

usando las instrucciones de T:

$$\overline{h(f_1(x_1, \dots, x_n, \dots, f_k(x_1, \dots, x_n))}$$

↑
qf

dejamos al lector llenar los detalles. ■

Lema 3. Si $f(\vec{x})$ y $g(\vec{x}, y, z)$ son T-calculables entonces

$$\begin{aligned} h(\vec{x}, 0) &= f(\vec{x}) \\ h(\vec{x}, n + 1) &= g(\vec{x}, n, h(\vec{x}, n)) \text{ es T-calculable.} \end{aligned}$$

Demostación. Sean T_f y T_g máquinas que calculan f y g , usando solo el lado derecho de la cinta. Describimos informalmente una máquina que calcula h , pos simplicidad suponemos que \vec{x} es una sola variable. Configuración inicial : $\vec{x} * \overline{n}$

(1) Copia \vec{x} a la derecha y marca último palote de \overline{n} con a:

$$\begin{array}{c} \vec{x} * \overline{n-1} a \quad \square \quad \overline{n} \\ \uparrow \end{array}$$

(2) calcula con T_f el valor $h(\vec{x}, 0) = f(\vec{x})$:

$$\begin{array}{c} \vec{x} * \overline{n-1} \quad a \quad \square \quad \overline{f(\vec{x})} \\ \uparrow \end{array}$$

(3) Copia $(\vec{x}, 0, f(\vec{x}))$ a la derecha y marca un palote de $\overline{n-1}$ con a:

$$\begin{array}{c} \vec{x} * \overline{n-2} \quad aa \quad \square \quad \vec{x} * \overline{0} * \overline{f(\vec{x})} \\ \uparrow \end{array}$$

(4) Calcula $h(\vec{x}, 1) = g(\vec{x}, 0, f(\vec{x}))$ con T_g :

$$\overline{x} * \overline{n-2} \text{ aa } \square \overline{h(x, 1)}$$

↑

(5) Copia a la derecha ($x, 1, h(x, 1)$) y marca un palote de $\overline{n-2}$:

$$\overline{x} * \overline{n-3} \text{ aaa } \square \overline{x} * \overline{1} * \overline{h(x, 1)}$$

↑

(6) Calcula $h(x, 2) = g(x, 1, h(x, 1))$ con T_g :

$$\overline{x} * \overline{n-3} \text{ aaa } \square \overline{h(x, 2)}$$

↑

En general si ha llegado a la configuración:

$$\overline{x} * \text{///...//} \underbrace{\text{a...a}}_{k+1} \square \overline{h(x, k)}$$

↑

Copia a la derecha ($x, k, h(x, k)$) y marca otro palote:

$$\overline{x} * \text{///...//} \underbrace{\text{a...a}}_{k+2} \square \overline{x} * \overline{k} * \overline{h(x, k)}$$

Luego calcula $h(x, k + 1) = g(x, k, h(x, k))$ con T_g :

$$\overline{x} * \text{///...//} \underbrace{\text{a...a}}_{k+2} \square \overline{h(x, k+1)}$$

↑

Finalmente se llega a la situación:

$$\overline{x}^* \text{ // } \dots \text{ // } \underbrace{a \dots a}_{n+1}, \square \quad \overline{h(x, n)}$$

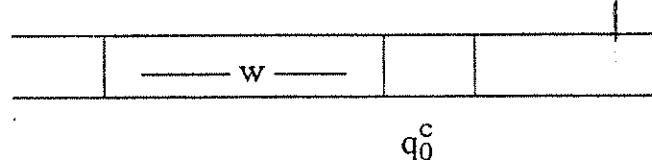
Al no haber más palotes que marcar, se borra \overline{x}^* a ... a y la máquina se detiene. ■

Lema 4. Si $f(\vec{x}, y)$ es T-calculable entonces la función

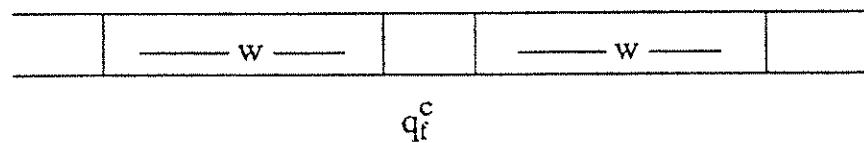
$$h(\vec{x}) = \mu^* y (f(\vec{x}, y) = 0)$$

es T-calculable. Aquí suponemos que si no existe y tal que $f(\vec{x}, y) = 0$ y $f(\vec{x}, i)$ está definida para $i = 0, 1, \dots$, y entonces $h(\vec{x})$ queda indefinida.

Demostración. Por simplicidad suponemos que \vec{x} es una sola variable. Sea T_f una máquina que calcula f usando solo el lado derecho de la cinta, y sean q_0^f, q_f^f sus estados inicial y final. Estamos suponiendo que $\bar{n} = (n + 1)$ palotes. Sea C una máquina que si se inicia en la configuración:

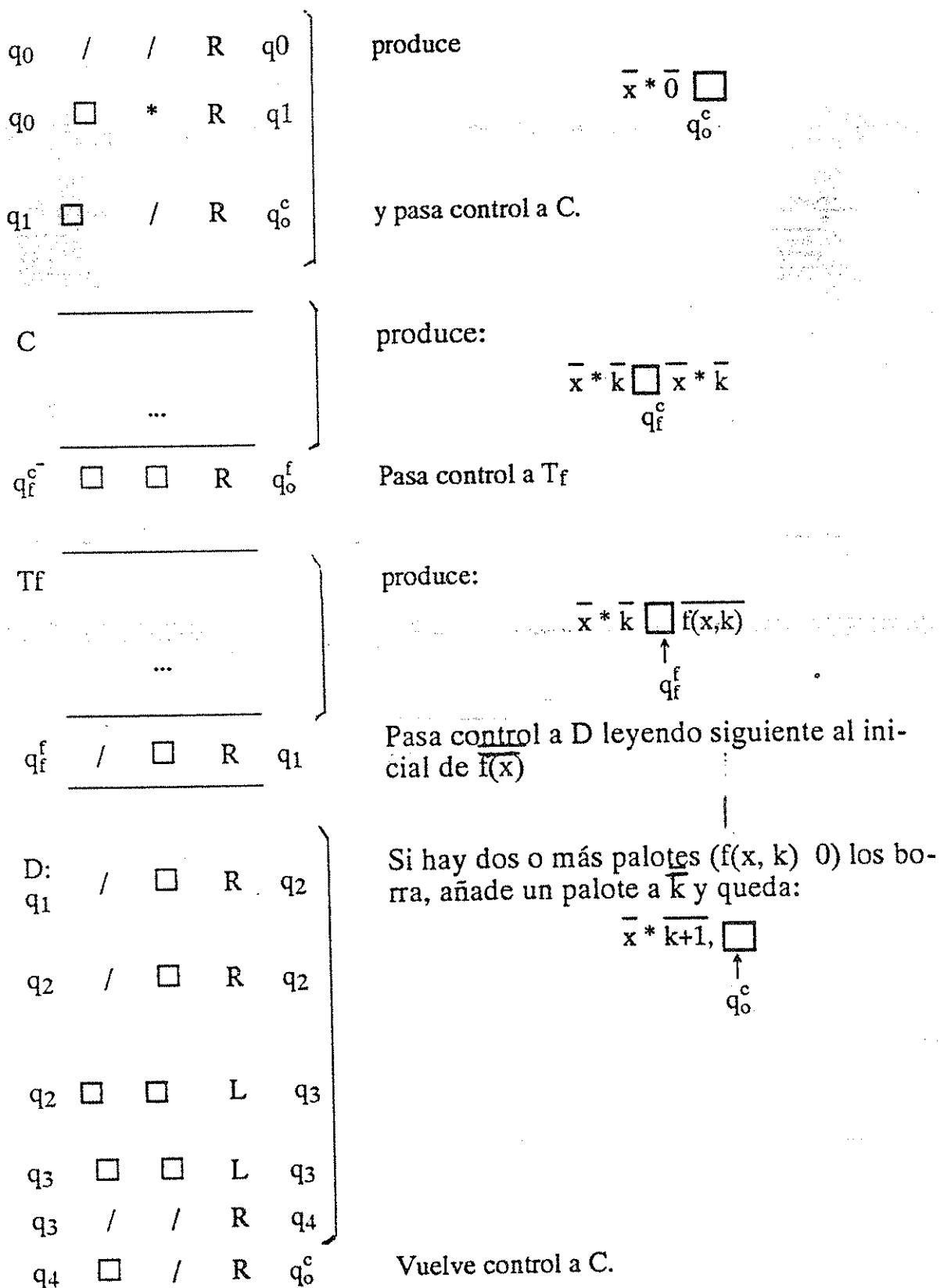


Hace una copia de w a la derecha y queda en la configuración:



Describimos una máquina que calcula h .

Configuración inicial: en el estado q_0 leyendo el primer palote de \vec{x} .



q_1	\square	\square	L	q_5	Si hay un solo palote ($f(x,k) = 0$) borra todo excepto k y se detiene en q_7 al comienzo de k .
q_5	\square	\square	L	q_5	
q_5	/	/	L	q_5	
q_5	*	*	L	q_6	
q_6	/	\square	L	q_6	
q_6	\square	\square	R	q_6	
q_6	*	\square	R	q_7	■

De los lemas 2, 3, 4 resulta que las funciones T-calculables son cerradas bajo composición, recurrencia y minimización de funciones totales; por lo tanto incluyen a todas las funciones recursivas parciales (y por supuesto totales), es decir toda función T-calculable de números naturales es recursiva (parcial). Se verá más adelante que el converso del Teorema también vale. Toda función T-calculable de \mathbb{N}^k es \mathbb{N} es recursiva (Capítulo 5).

Colorario Si $S \subseteq \mathbb{N}^k$ es un conjunto recursivo, su función característica es calculable por una máquina de Turing.

Ejercicios

- Demuestre que si $S \subseteq \mathbb{N}$ es recursivamente enumerable, existe una máquina de Turing que al inicializarse sobre la cinta vacía no se detiene nunca e imprime continuamente de izquierda a derecha una lista exhaustiva de los elementos de S en notación monádica, separados por \square .

2. Demuestre que si $S \subseteq \mathbb{N}$ es recursivamente enumerable existe una máquina M tal que $S = \{n : \bar{n} \text{ está en el dominio de } F_M\}$.

3.* *Cintas de dos o más pistas.* Sea T una máquina de Turing que lee e imprime en una cinta de dos pistas:

1			s				
2					t		

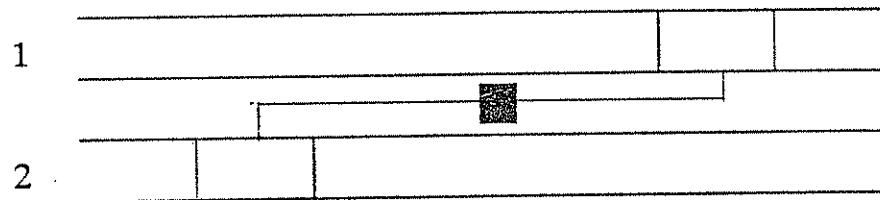
Sus instrucciones son del tipo:

$$q_i \ s(n) \ s(m) \ q_j \quad n, m \in \{1, 2\}$$

"Si está en el estado q_i leyendo el símbolo s en la pista n, imprima t en la pista m, muévase a la M y pase al estado q_j ".

Se puede definir f_T^1 , la función T que calcula en la pista 1 (con ayuda de la pista 2). Demuestre que existe una máquina T' de una sola pista tal que $f_{T'} = f_T^1$.

4.* *Máquinas de dos o más cintas.* Sea T una máquina que lee y escribe en dos cintas por medio de cabezas independientes:



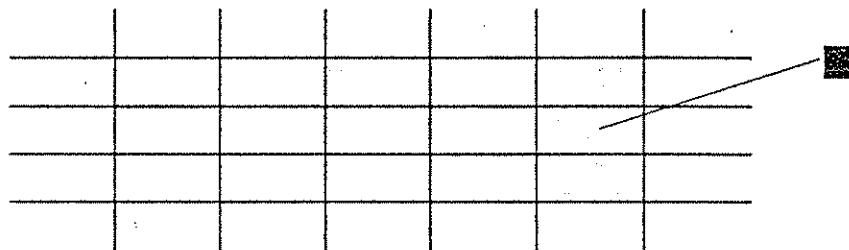
Sus instrucciones son de la forma:

$$q_i \left[\begin{matrix} s \\ s' \end{matrix} \right] \left[\begin{matrix} t \\ t' \end{matrix} \right] \left[\begin{matrix} M \\ M' \end{matrix} \right] q_j$$

donde M, M' puede ser R (derecha), L (izquierda) o Q (quieta). "Si está en el estado q_i leyendo s en la primera cinta y s' en la segunda, imprima t, en la primera cinta y t' en la segunda, muévase a ma M en la primera cinta y a la M' en la segunda, pase entonces al estado q_j ".

Demuestre que existe T' de una sola cinta tal que $F_{T'}$ es igual a la función T que calcula en la cinta 1.

5.* *Máquinas bidimensionales.* Sea T una máquina que lee e imprime sobre las celdas de una cuadrícula, y puede moverse a izquierda, derecha, arriba, (N) o abajo (S)



Sus instrucciones tienen la forma $q_i \ s \ t \ M \ q_j$, en donde M puede ser R, L, N, y S. Suponga que se inicializa con la cabeza al comienzo de una palabra W escrita en una banda horizontal de la cuadrícula. Sea $F_T(w)$ el contenido de tal banda cuando T se detiene, si es que lo hace. Demuestre que existe una máquina de Turing T' de una sola cinta tal que $F_{T'} = F_T$. Este resultado se puede generalizar a máquinas que trabajan sobre "memorias" n -dimensionales para cualquier n , y sirve para convencernos de que una computadora digital puede simularse por medio de una máquina de Turing.

Capítulo 4

El problema de la parada

4.1 Máquinas Universales

Las instrucciones de una máquina de Turing se pueden considerar como un “programa”. Las máquinas que hemos estudiado hasta ahora calculan su propio programa. ¿Es posible tener una máquina de Turing que calcule con diferentes programas? Veremos en esta sección que existe una máquina de Turing tal que si se le dà como dato la descripción de una máquina arbitraria T (el programa) y una palabra w , calcula $f_T(w)$. Tales máquinas son llamadas *máquinas universales*, y son la versión matemática de un computador digital, o de un compilador.

Sabemos que no hay pérdida de generalidad si nos limitamos a máquinas que trabajen en una semi cinta potencialmente infinita a la derecha. Si T es una máquina de Turing sobre el alfabeto $\Sigma = \{s_1, \dots, s_l\}$ con estados q_0, \dots, q_m , la configuración de la cinta en un momento dado del funcionamiento de T está completamente determinada por un segmento de la cinta que contenga todos los símbolos impresos de Σ , una indicación de la celda que está leyendo y el estado en que se halla. La configuración:

	a_1		a_2	\dots	r	s	u	\dots	a_k
	q_i					u			

puede codificarse como una palabra sobre el alfabeto $\Sigma \cup \{B, q_0, \dots, q_n\}$:

$$a_1 B a_2 \dots r q_i s u \dots a_k B \quad (1)$$

donde B denota \square y q_i se escribe a la izquierda del símbolo que esté leyendo. La ejecución de una instrucción $q_i s t R q_j$ de T corresponde a trasformar el código (1) en el código :

$$a_1 B a_2 \dots r t q_j u \dots a_k B.$$

Y la aplicación de la instrucción $q_i st L q_j$ corresponde a transformar (1) en:

$$a_1 B a_2 \dots q_j r t u \dots a_k B.$$

Si q_i está en algún extremo de la configuración la transformación se hace como si estuviera seguido (o antecedido) por B. Para tener una máquina universal basta construir una máquina capaz de realizar las transformaciones arriba indicadas.

Como no hay cota en el número de símbolos del alfabeto o los estados de T, una máquina que trabajara con todas las posibles configuraciones debería trabajar con un alfabeto infinito lo cual no es posible por definición. Sin embargo, esto puede obviarse. Cualquiera que sea la máquina T, los símbolos de su alfabeto y sus estados pueden codificarse como palabras sobre el alfabeto:

$$\{s, q, /, R, L\}$$

Basta hacer la identificación $\square \mapsto s, s_i \mapsto \bar{s}_i, q_j \mapsto \bar{q}_j$,

donde \bar{n} es n en monádico. De esta manera cada configuración podría codificarse como una palabra sobre el mismo alfabeto. La configuración

C:		s_3	s_1		s_3	s_1			s_2
							q_2		

quedaría

$$\bar{C} : s // / s / s q // / s // / s / s s s //$$

Las instrucciones de T y la máquina T misma también podrían codificarse como palabras sobre el mismo alfabeto. La instrucción $I = q_i s_e s_r R q_j$ quedaría:

$$\bar{I} : \quad q \quad // . / \quad s \quad // . / \quad s \quad // . / \quad R \quad q \quad // . / \\ i \qquad e \qquad r \qquad j$$

y si las instrucciones de T son I_1, \dots, I_k , la codificación de T sería:

$$\bar{T} : \bar{I}_1 \bar{I}_2 \dots \bar{I}_k$$

En general, usamos “—” para indicar la codificación de una configuración, instrucción o máquina.

Si C y D son configuraciones para la máquina T usamos la notación $T : C \vdash D$ para indicar que T contiene una instrucción que transforma la configuración C en D; $T : C |— D$ significa que existen configuraciones $C = C_0, C_1, \dots, C_n = D$ tales que $T : C_i \vdash C_{i+1}$.

Teorema. Existe una máquina de Turing U sobre el alfabeto $\{s, q, /, R, L, *\}$ tal que para toda máquina de Turing T y configuraciones C, D se tiene:

(a) Si $T : C \vdash D$ entonces $U :$

$$\begin{array}{ccc} \bar{T} * \bar{C} & \vdash & \bar{T} * \bar{D} \\ \uparrow & & \uparrow \\ q_0 & & q_0 \end{array}$$

(b) Si T se detiene en la configuración $\alpha q_i \beta$ entonces U se detiene en $\bar{T} * \bar{\alpha} q_i \bar{\beta}$.

Demostración. Suponga que $T = I_1 \dots I_k$ y sea $q_i s_e s_r R q_j$ una de las instrucciones. Debemos poner instrucciones en U de manera que transforme:

$$\bar{I}_1 \dots \bar{I}_k * \overline{\alpha} q / \dots / s / \dots / \overline{\beta}$$

en

$$\bar{I}_1 \dots \bar{I}_k \stackrel{*}{\sim} \overline{\alpha} s / \dots / q / \dots / \overline{\beta}$$

(recuérdese que usamos s para codificar \square , la configuración codificada αq debe cambiarse a $\alpha q \bar{s}$ antes de ser transformada). De la misma manera, si contiene la instrucción $q_i s_e s_r L q_j$ entonces U debe transformar

T * a sm qī se B

en

$\overline{T}^* \overline{\underline{\alpha}} q \bar{j} s \bar{m} s \bar{r} \overline{\underline{\beta}}$

cualquiera que sea m (Esto incluye el caso especial

T * q̄ s̄ B̄

antes de hacer la transformación todo el “dato” debe correrse un lugar a la derecha y debe imprimirse s en la celda que queda en blanco:

—T * s qī sè B

Lo anterior se logra haciendo que U compare el segmento inicial $q_i^n s_e^n$ de cada instrucción \bar{I}_n , $n=1,2 \dots$ con el segmento $q_i s_e$. La comparación se hace símbolo por símbolo y en caso de haber coincidencia se pasa el control a una rutina que realiza la transformación de acuerdo a la instrucción en cuestión. Si no hay coincidencia se pasa a la siguiente instrucción. Si se agotan las instrucciones sin que haya coincidencia con ninguna, la máquina se detiene. Describimos a U informalmente, indicando su actividad en algunos de sus estados aunque más estados de los indicados son necesarios.

I. Comparación

Inicialmente U está en el estado q_0 leyendo el primer símbolo q de \bar{I}_1 . En aplicaciones posteriores estará inicialmente en el mismo estado leyendo el primer símbolo q de \bar{I}_n $n=1, 2, \dots, k$:

	máquina						*	"dato"			
	$q_{\bar{i}_n}$	$s_{\bar{e}_n}$	$s_{\bar{r}_n}$	M	q	\bar{J}_n	$q_{\bar{i}}$	$s_{\bar{e}}$	β
	\uparrow										
	o										

- q0: Busca primer palote no marcado de $q_{\bar{i}_n}$.
 Si lo hay, lo marca * y pasa a q_1 .
 Si no lo hay ha marcado todo $q_{\bar{i}_n}$ y el siguiente símbolo no marcado es s, pasa a q_2 .
- q1: Busca símbolo q en el "dato" (segmento después de \bar{T}^*) y el primer palote no marcado después de q.
 Si lo hay, lo marca * y pasa a q_3 .
 Si no lo hay la comparación ha fallado, $i_n > i$, pasa control a q_5 .
- q3: Se devuelve a buscar marca * en instrucción \bar{I}_n al hallarla pasa control a q_0 .
- q2: Busca símbolo q en el "dato" y al primer símbolo no marcado después de q.
 Si es palote la comparación ha fallado, $i_n < i$, pasa control a q_5 .
 Si es s ó \square entonces $q_{\bar{i}_n}$ y $q_{\bar{i}}$ coinciden. El blanco lo cambia a s, pasa control a q_4 .
- q4: (Se tiene $q_{\bar{i}_n}=q_{\bar{i}}$). Se devuelve a buscar marca * en \bar{I}_n , al hallarla localiza primer palote no marcado después de $q_{\bar{i}_n}$ s.
 Si lo hay, lo marca * y pasa a q_1' .
 Si no lo hay, ya ha marcado todo $q_{\bar{i}_n}s_{\bar{e}_n}$, el siguiente símbolo no marcado es s, pasa a q_2' .
- q1': Busca símbolo q en el "dato" y el primer palote no marcado después de $q_{\bar{i}}s$.
 Si lo hay, lo marca * y pasa a q_4 .

Si no lo hay la comparación ha fallado, $e_n > e$, pasa control a q_5 .

q_2' : Busca símbolo q en el "dato" y el primer símbolo no marcado después de $q \bar{i} s$.

Si es palote la comparación ha fallado, $e_n < e$, pasa a q_5 .

Si es s ó \square entonces q_{in} sen y q_i se coinciden, pasa control a q_9 .

q_5 : ($q_{in} \bar{s} \bar{e}_n \neq q_i \bar{s} \bar{e}$). Se devuelve reponiendo palotes en $q_i \bar{s} \bar{e}$. Al hallar marca * en \bar{I}_n repone palotes en $q_{in} \bar{s} \bar{e}_n$, pasa a q_6 .

q_6 : Busca primer símbolo después de \bar{I}_n .

Si es q , es el primer símbolo de \bar{I}_{n+1} , pasa control a q_0 leyendo este símbolo y se reinicia el proceso de comparación.

Si es *, agotó todas las instrucciones, la máquina se detiene en un estado final q_f .

II. Transformación

Inicialmente U está en la configuración

... <u>$q \bar{i} s \bar{e} s \bar{r}_n M q \bar{j}_n$</u> ...	$\bar{\alpha} q \bar{i} s \bar{e} \bar{\beta}$
↓ I_n	q_9

en donde doble barra indica palotes marcados.

q_9 : Cambia s de $q \bar{i} s \bar{e}$ por * y se devuelve hasta \bar{I}_n (lo recorre por las marcas =).

Examina símbolo M .

Si $M = R$ pasa a q_{10} .

Si $M = L$ pasa a q_{20} .

q_{10} : Busca primer palote no marcado de $\bar{s} \bar{r}_n$.

Si lo hay lo marca * y pasa a q_{11} .

Si no lo hay, agotó $\bar{s} \bar{r}_n$, pasa a q_{12} .

q_{11} : Busca símbolo q en el dato y primer símbolo diferente de / después de q .

Si es * la reemplaza por / y pasa a q_{13} .

Si es s ó \square (el primer símbolo β), corre β una celda a la derecha, imprime / en la celda que queda en blanco y pasa a q_{13} .

q_{13} : Se devuelve a \bar{I}_n y pasa control a q_{10} .

q_{12} : Ya copió \bar{s}_{r_n} . Busca símbolo q en el dato y primer símbolo diferente de / después de q.

Si es * lo reemplaza por q y pasa a q_{13}' .

Si es s ó \square , corre β un lugar a la derecha e imprime q en la celda que queda en blanco, pasa a q_{13}' .

q_{13} : Se devuelve a \bar{I}_n y pasa control a q_{10} .

q_{10} : Busca primer palote no marcado de $\bar{s}_{r_n} R q \bar{j}_n$.

Si lo hay, lo marca * y pasa a q_{11}' .

Si no lo hay, agotó $\bar{s}_{r_n} q \bar{j}_n$, pasa a q_{12}' .

q_{11} : Busca segundo símbolo q en el dato, *de paso cambia el primero a s*, y busca primer símbolo diferente de / después de q.

Si es * lo reemplaza por / y pasa a q_{13}' .

Si es s ó \square (primer símbolo de β) corre β un lugar a la derecha e imprime en la celda que queda, pasa a q_{13}' .

q_{12} : Ya copió todo $\bar{s}_{r_n} q \bar{j}_n$. Busca símbolo q en el dato y el primer símbolo marcado * después de q.

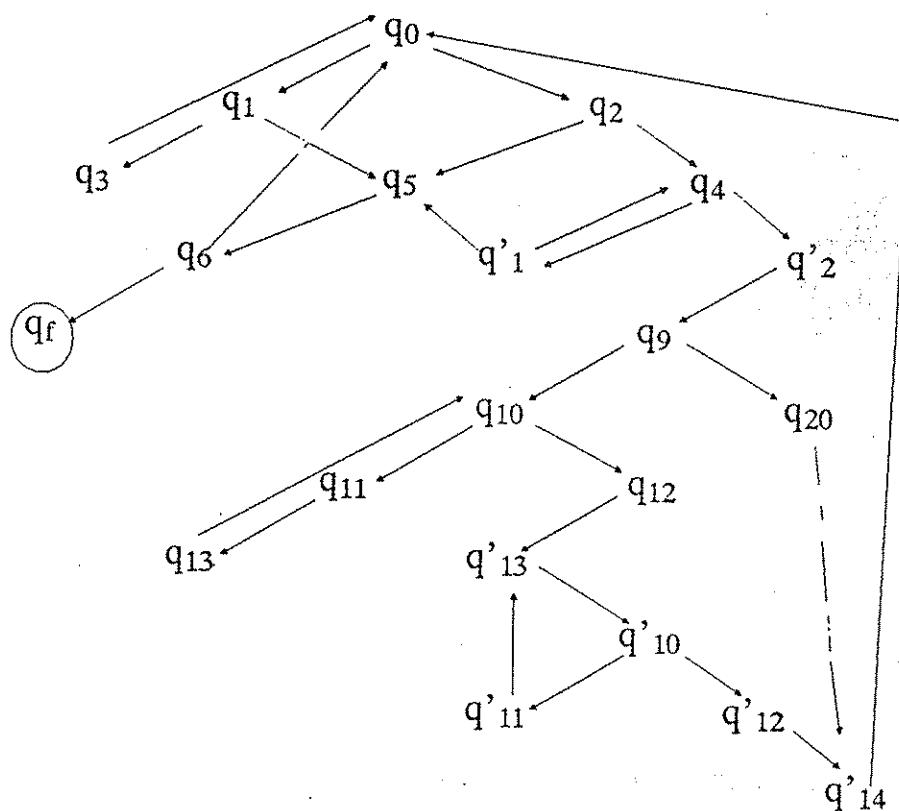
Si no lo hay, terminó pasa a q_{14} .

Si lo hay, borra el bloque de * que sigue y corre β a la izquierda a llenar los blancos resultantes. Entonces pasa a q_{14} .

q_{14} : Se devuelve a \bar{I}_n , repone palotes en todo \bar{I}_n , y termina en el extremo izquierdo de \bar{T} en el estado q_0 (lista a iniciar un nuevo ciclo).

q_{20} : Aquí comienza la rutina de movimiento a la izquierda, es similar a lo anterior con algunas complicaciones adicionales pues hay que intercalar el segmento S_m del dato, anterior a q, entre $q \bar{j}_n$ y $s \bar{e}_n$. Al terminar la transformación pasa al estado q_{14} y finalmente q_0 .

El siguiente diagrama da una idea del funcionamiento de la máquina, cada flecha de transición representa una o varias acciones. Es un buen ejercicio anotar sobre el diagrama las condiciones bajo las cuales se toma un camino u otro, y las acciones realizadas en cada transición.



Colorario 1. Existe una máquina universal U tal que para toda máquina T y palabra W sobre el alfabeto $\{s_1, \dots, s_m\}$ de T :

$$f_U(\overline{T} * \overline{W}) = \begin{cases} f_T(W) & \text{si } f_T(W) \text{ está definida} \\ \text{indefinida} & \text{de lo contrario} \end{cases}$$

En particular, $\overline{T} * \overline{W}$ está en el dominio de f_U si y solo si W está en el dominio de f_T .

Demostración. Del Teorema anterior es claro que si W no está en el dominio de f_T hay una sucesión infinita de configuraciones

$$T: q_0 W \perp C_1 \perp C_2 \perp \dots$$

y por lo tanto una sucesión infinita:

$$U: q_0 \overline{T} * \overline{W} \vdash q_0 \overline{T} * \overline{C}_1 \vdash q_0 \overline{T} * \overline{C}_2 \vdash \dots$$

y U no se detiene; entonces $f_U(\bar{T} * \bar{W})$ y $f_T(W)$ están ambas indefinidas. En cambio, si W está en el dominio de f_T la primera sucesión se detiene y tenemos:

$$T: q_0 W \vdash \alpha q_i \beta$$

terminalmente. Por el teorema anterior

$$\begin{array}{ccc} U: q_0 \bar{T} * \bar{W} \vdash & \bar{T} * \overline{\alpha q_i \beta} \\ & | \\ & q_f \end{array}$$

terminalmente. Para tener la máquina deseada basta añadir a U instrucciones que a partir del estado q_f borran $\bar{T} * \overline{\alpha q_i \beta}$, deteniéndose en el primer símbolo terminando en la configuración final $q_f^3 \beta$. De esta manera $f_U(\bar{T} * \bar{W}) = \beta = f_T(W)$. ■

Sea $\Sigma_o = \{ q, s, /, R, L, *\}$, el alfabeto de la máquina universal. Podemos identificar:

$$q \leftrightarrow s/, / \leftrightarrow s//, s \leftrightarrow s///, R \leftrightarrow s////, L \leftrightarrow s/////, * \leftrightarrow s////////$$

y para toda palabra $W \in \Sigma_o^*$ hay una palabra codificada $\bar{W} \in \{s, /\}^*$. El colorario anterior nos dice que para toda máquina T

$$f_U(\bar{T} * \bar{W}) = \overline{f_T(W)}.$$

¿Es posible modificar U de manera que si $W \in \Sigma_o^*$ $\psi \phi_T(\Omega) \in \Sigma_o^*$ εντονχεσ

$$f_U(\bar{T} * W) = f_T(W) ?$$

Claro que sí, basta añadir a U instrucciones iniciales y finales de manera que tengamos:

$$U: \bar{T} * W \vdash \bar{T} * \bar{W} \vdash \overline{f_T(W)} \vdash f_T(W)$$

es decir rutinas que codifiquen de S_o^* a $\{s,/\}^*$ y viceversa. Esto puede requerir símbolos auxiliares.

Colorario 2. *Existe una máquina U' cuyo alfabeto contiene a Σ_o y tal que para toda máquina T y palabra $W \in \Sigma_o^*$, si $\phi_T(\omega) \in \Sigma_o^*$, entonces*

$$f_{U'}(\bar{T} * W) = f_T(W).$$

Además, para cualquier palabra X, si $f_{U'}(X)$ está definida, entonces pertenece a Σ_o^* .

Demostración. Además de las observaciones anteriores, basta anotar que si la decodificación de $f_{U'}(X)$ es imposible porque hay símbolos no en Σ_o , o porque aparecen sucesiones de más de 5 palotes que no corresponden al código de ningún símbolo de Σ_o se pueden dar instrucciones para que el resultado se reduzca a puros palotes. ■

4.2 El problema de la parada

¿Es posible predecir para qué configuraciones iniciales una máquina de Turing T dada no se detiene? Este es el problema de la parada. Aparentemente bastaría analizar las instrucciones de T para determinar bajo qué condiciones T entra en “loop” y por lo tanto no se detiene. Si embargo esto no es así, veremos que hay máquinas para las cuales es imposible decidir algorítmicamente si se detiene ante una configuración inicial dada. El problema de la parada es *insoluble*. Claro está que hay máquinas para las cuales el problema es soluble, por ejemplo aquellos que paran siempre!. Formulamos el problema más precisamente. Dada una máquina de Turing T, ¿existe una máquina D tal que para toda $w \in \Sigma^*$ (Σ el alfabeto de T):

$$f_D(W) = \begin{cases} 1 & \text{si } f_T(W) \text{ está definido} \\ 0 & \text{de lo contrario} \end{cases}$$

es decir, D calcula la función característica del dominio de f_T ?

En adelante, sea $\text{dom}(T) = \text{dominio de } f_T$.

Teorema 1. Existe una máquina de Turing T para la cual el problema de la parada es insoluble (por medio de máquinas de Turing y así por medio de cualquier algoritmo conocido).

Demostración. Sea U' una máquina universal como en el Colorario 2 de la sección anterior cuyo alfabeto contiene a Σ_0 . Es fácil construir una máquina I tal que para todo $W \in \Sigma_0^*$

$$f_I(W) = f_{U'}(W^* W)$$

Basta añadir instrucciones iniciales que transformen W en $W^* W$ y pasen el control a U . Note que para $w \in \Sigma_0^*$, $f_I(w) \in \Sigma_0^*$ si está definida, por la segunda parte del Colorario 2.

Suponga que el problema de la parada es soluble para I y sea D una máquina de Turing tal que para toda $w \in \Sigma_0^*$

$$f_D(w) = \begin{cases} 1 & // \text{ si } w \in \text{dom}(I) \\ 0 & / \text{ si } w \notin \text{dom}(I) \end{cases}$$

Entonces se puede construir M tal que:

$$f_M(w) = \begin{cases} f_I(w) & // \text{ si } w \in \text{dom}(I) \\ / & \text{si } w \notin \text{dom}(I) \end{cases}$$

Basta que M pueda hacer las transformaciones:

$$M: w \vdash w \square w \vdash w \square f_D(w)$$

$$M: w \square // \vdash f_I(w) //$$

$$M: \square w / \vdash /.$$

Obviamente f_M está definida para todo $w \in \Sigma_0^*$ y además $f_M(W) \in \Sigma_0^*$. Sea M el código de M en Σ_0^* , como $f_M(\overline{M}) \in \Sigma_0^*$ entonces $f_{U'}(\overline{M} * \overline{M}) = f_M(\overline{M})$ por la primera parte del Colorario 2, y así:

$$\begin{aligned}\overline{M} \notin \text{dom}(I) &\Leftrightarrow f_I(\overline{M}) = f_U(\overline{M} * \overline{M}) \text{ está indefinida} \\ &\Leftrightarrow f_M(\overline{M}) \text{ está indefinida}\end{aligned}$$

Como f_M está siempre definida, tenemos que $\overline{M} \in \text{dom}(I)$, por lo tanto las siguientes palabras son idénticas:

$$f_M(\overline{M}) = f_I(\overline{M}) // = f_U(\overline{M} * \overline{M}) // = f_M(\overline{M}) //$$

lo cual es una contradicción. Debemos concluir que la máquina D no puede existir! ■

La insolubilidad del problema de la parada implica la insolubilidad de otros problemas igualmente interesantes.

Ejemplo 1

El problema de determinar si dos máquinas M, M' sobre el mismo alfabeto son equivalentes ($f_M = f_{M'}$) es insoluble.

Demostraremos que si este problema fuera soluble al de la parada también lo sería. Sea E una máquina que resuelve el problema de la equivalencia:

$$f_E(\overline{M} * \overline{M'}) = \begin{cases} 1 & \text{si } f_M = f_{M'} \\ 0 & \text{si } f_M \neq f_{M'} \end{cases}$$

para máquinas sobre el alfabeto Σ .

Sea I una máquina cuyo problema de la parada es insoluble sobre Σ .

Sea T_o una máquina que calcula la función constante $f(u) = 0, \forall u \in \Sigma^*$. Para cada $W \in \Sigma^*$ existe una máquina T_w que calcula para todo $u \in \Sigma^*$

$$f_{T_w}(u) = \begin{cases} 0 & \text{si } f_{T(w)} \text{ definido} \\ \text{Indefinido} & \text{si } f_{T(w)} \text{ indefinido} \end{cases}$$

ésto se logra componiendo una máquina que transforma cualquier palabra en w , calcula luego $f_I(w)$ y manda todas las combinaciones de parada de a

instrucciones que imprimen 0 como resultado final. No es difícil ver que las instrucciones de T_w se pueden construir algorítmicamente a partir de las instrucciones de I y la palabra W; por lo tanto existe una máquina de Turing que calcula para todo $W \in \Sigma^*$

$$w \vdash \bar{T}_w.$$

Además, $W \in \text{dom}(I) \Leftrightarrow T_w$ calcula la función constante 0 en Σ^*

$$\Leftrightarrow f_{T_w} = f_{T_0}$$

$$\Leftrightarrow f_E(\bar{T}_w * \bar{T}_0) = 1$$

Por lo tanto una máquina que calcula

$$w \vdash \bar{T}_w \vdash \bar{T}_w * \bar{T}_0 \vdash E(\bar{T}_w * \bar{T}_0)$$

resuelve el problema de la parada para I, una contradicción.

Ejercicios

- Demuestre que el problema de la parada es insoluble para cualquier máquina universal U.
- ** Sea M una máquina de Turing tal que para cualquier palabra $w \in \Sigma^*$, en el cálculo de $f_M(w)$ (que pueda prolongarse indefinidamente) la máquina nunca va más lejos que $K|w$ celdas de la posición inicial, donde K es una constante y $|w|$ es la longitud de w. Demuestre que el problema de la parada para M es *soluble* por una máquina de Turing.
- Describa la máquina de Turing que realiza la transformación $w \vdash \bar{T}_w$ en el Ejemplo 1 de esta sección.

** Demuestre que toda función total f de $\Sigma^* = \{q, s, /, R, L, *\}^*$ en Σ^* que extiende a f_U donde U es una máquina universal, no puede ser calcu-

lable. (Ayuda: proceda como en la prueba del Teorema 1 y defina M tal que

$$f_M(w) = \begin{cases} f(w) & \text{si } w \in \text{dom}(I) \\ \text{Indefinido} & \text{de lo contrario} \end{cases}$$

y demuestre que $f(\bar{M} * \bar{M})$ no está definida.)

4. Demuestre que el problema de decidir si una máquina M sobre Σ está definida o no para todo $w \in \Sigma^*$ no es soluble.
5. Demuestre que el problema de decidir si el dominio de una máquina de Turing es vacío o no, no es soluble.
6. Demuestre que existe una máquina de Turing que calcula una función parcial $f: \mathbb{N} \rightarrow \mathbb{N}$ tal que $\{n : f(n) \text{ está definida}\}$ no es recursivo.

Capítulo 5

Recursividad de funciones Turing-calculables

5.1 Codificación recursiva de sucesiones finitas

Sea p_n el n -ésimo primo. Sabemos que la función $p(n) = p_n$ es primitivamente recursiva. Podemos establecer una biyección.

$$g: \bigcup_{k=1}^{\infty} \mathbb{N}^k \rightarrow \mathbb{N}$$

definiendo $g((n_1, \dots, n_k)) = (p_1^{y_1} p_2^{y_2} \dots p_k^{y_{k+1}}) - 2$, gracias al teorema de descomposición única en primos de los números naturales mayores o iguales que 2. Escribiremos:

$$\langle n_1, \dots, n_k \rangle = g((n_1, \dots, n_k)).$$

De esta manera podemos afirmar que cada número $n \in \mathbb{N}$ codifica una única k -tupla, cuya longitud k también está únicamente determinada por n . Veremos que toda la información sobre la tupla puede recobrarse de n por medio de funciones primitivamente recursivas. Sean:

$$L(n) = \text{longitud de la tupla codificada por } n.$$

$$P(i, n) = \begin{cases} i\text{-ésima componente de tupla codificada por } n, & \text{si } 1 \leq i \leq L(n) \\ 0 & \text{de lo contrario} \end{cases}$$

Estas funciones son p.r. pues si

$$n = \langle n_1, \dots, n_k \rangle = p_1^{n_1} p_2^{n_2} \dots p_k^{n_k+1} - 2$$

entonces $n + 2 = p_1^{n_1} \dots p_k^{n_k+1}; L(n) = k$ y P_k es el máximo primo que divide a $n+2$. Como $k < P_k \leq n+2$, tenemos que P_k es el único primo tal que $P_k | (n+2)$ y $P_{k+1}, \dots, P_{n+2} \nmid (n+2)$.

Por lo tanto:

$$L(n) = \mu r \leq n+2 [p(r) | n+2] \wedge \forall s \leq n+2 (s > r \rightarrow p(s) \nmid (n+2))$$

Análogamente:

$$P(i, n) = \begin{cases} \mu \alpha \leq n+2 [p(i)^\alpha | n+2 \wedge p(i)^{\alpha+1} \nmid n+2] & \text{si } 0 < i < L(n) \\ \mu \alpha \leq n+2 [p(i)^{\alpha+1} | n+2 \wedge p(i)^{\alpha+2} \nmid n+2] & \text{si } i = L(n) \\ 0 & \text{si } i = 0 \end{cases}$$

Por el teorema de minimización acotada y definición por casos, las funciones resultan p.r..

Definimos ahora una operación entre números naturales que corresponde a la concatenación de las tuplas por ellos codificadas.

Definición.

$$n * m = \left[\frac{n+2}{p(L(n))} \right] \left[\prod_{i=1}^{L(m)} P(L(n) + i)^{P(i,m)} \right] P(L(n) + L(m))^{P(L(m), m)} - 2$$

Entonces si $n = \langle n_1, \dots, n_k \rangle$ y $m = \langle m_1, \dots, m_e \rangle$ tenemos $L(n) = k$, $L(m) = e$, $n+2 = p_1^{n_1} \dots p_k^{n_k+1}$, $P(i, m) = m_i$, y por lo tanto

$$n * m = \left[\frac{p_1^{n_1} \dots p_k^{n_k+1}}{p_k} \right] \left[\prod_{i=1}^e P(k+i)^{m_i} \right] P_{k+e}^{m_e} - 2$$

$$= (p_n^{n_1} \dots p_k^{n_k}) (p_{k+1}^{m_1} \dots p_{k+e}^{m_e+1}) - 2$$

$$= \langle n_1, \dots, n_k, m_1, \dots, m_e \rangle$$

Podemos resumir todo lo anterior en el siguiente lema:

Lema. Existen funciones p.r. $L(n)$, $P(i, n)$, n^*m tales que

a) $L(\langle n_1, \dots, n_k \rangle) = k$

b) $P(i, \langle n_1, \dots, n_k \rangle) = \begin{cases} n_i & \text{si } 1 \leq i \leq k \\ 0 & \text{de lo contrario} \end{cases}$

c) $\langle n_1, \dots, n_k \rangle * \langle m_1, \dots, m_e \rangle = \langle n_1, \dots, n_k, m_1, \dots, m_e \rangle$.

La operación $*$ es obviamente asociativa de $\bigcup_{k=1}^{\infty} \mathbb{N}^k$ y g es un isomorfismo entre

el semigrupo $\bigcup_{k=1}^{\infty} \mathbb{N}^k$ con concatenación y el semigrupo $(\mathbb{N}, *)$. La siguiente

observación es trivial pero importante:

Para k fijo, la función $g_k: \mathbb{N}^k \rightarrow \mathbb{N}$ que restringe a g ,

$$g_k(n_1, \dots, n_k) = \langle n_1, \dots, n_k \rangle, \text{ es p.r.}$$

A veces se define una función por recurrencia de manera que $f(\vec{x}, y+1)$ depende no solamente de $f(\vec{x}, y)$ sino de todos o algunos de los valores anteriores $f(\vec{x}, 0), \dots, f(\vec{x}, y)$, es decir $f(\vec{x}, y+1)$ depende del número $\langle f(\vec{x}, 0), \dots, f(\vec{x}, y) \rangle$. El siguiente resultado muestra que esto no nos saca de la clase de las funciones primitivamente recursivas (o recursivas).

Teorema. Sea $g(x)$ y $h(x, y, z)$ funciones primitivamente recursivas (respectivamente, recursivas) entonces la función f definida por:

$$\begin{cases} f(x, 0) = g(x) \\ f(x, y+1) = h(x, y, \langle f(x, 0), \dots, f(x, y) \rangle) \end{cases}$$

es primitivamente recursiva (resp., recursiva).

Demostración. Defina $f(x, y)$ de la manera siguiente

$$\begin{aligned}\tilde{f}(\vec{x}, 0) &= g(\vec{x}) \\ \tilde{f}(\vec{x}, y+1) &= \tilde{f}(\vec{x}, y) * \langle h(\vec{x}, y), \tilde{f}(\vec{x}, y) \rangle\end{aligned}$$

\tilde{f} es obviamente primitivamente recursiva. Además

$$\tilde{f}(\vec{x}, y) = (\vec{x}, 0), \dots, f(\vec{x}, y)$$

como lo muestra la siguiente inducción:

$$\begin{aligned}\tilde{f}(\vec{x}, 0) &= \langle g(\vec{x}) \rangle = \langle f(\vec{x}, 0) \rangle \\ \tilde{f}(\vec{x}, y+1) &= \tilde{f}(\vec{x}, y) * \langle h(\vec{x}, y), \tilde{f}(\vec{x}, y) \rangle \\ &= \langle f(\vec{x}, 0), \dots, f(\vec{x}, y) \rangle * \langle h(\vec{x}, y), \langle f(\vec{x}, 0), \dots, f(\vec{x}, y) \rangle \rangle\end{aligned}$$

por hipótesis de inducción

$$\begin{aligned}&= \langle f(\vec{x}, 0), \dots, f(\vec{x}, y) \rangle * \langle \vec{x}, y+1 \rangle \\ &= \langle f(\vec{x}, 0), \dots, f(\vec{x}, y), f(\vec{x}, y+1) \rangle\end{aligned}$$

Por lo tanto,

$$f(\vec{x}, y) = P(y+1, \tilde{f}(\vec{x}, y))$$

debe ser primitivamente recursiva. ■

Ejemplo

Los números de Fibonacci definidos por

$$F_0 = 1, F_1 = 1, F_{n+1} = F_n + F_{n-1} \quad \text{si } n \geq 1$$

son tales que la función $f(n) = F_n$ es primitivamente recursiva, pues puede definirse por

$$\begin{cases} f(0) = 1 \\ f(n+1) = h(n, \langle f(0), \dots, f(n) \rangle) \end{cases}$$

donde $h(y, z) = P(y+1, z) + P(y, z)$.

Ejercicios

1. Sea $f, r: \mathbb{N} \rightarrow \mathbb{N}$ funciones p.r., demuestre que la función

$$h(n) = \langle f(0), \dots, f(r(n)) \rangle$$

es p.r.

2. Utilice el teorema demostrado en esta sección para demostrar que si $g(x)$, $h(x,y,z)$ y $r(x,y)$ son funciones p.r. tales que

$$r(x,y) \leq x, s(x,y) \quad \text{y} \quad \text{para } y > 0$$

entonces la función f definida por

$$\begin{cases} f(x,0) = g(x) \\ f(x,y) = h(x,y,f(r(x,y),s(x,y))) \quad \text{si } y > 0 \end{cases}$$

es p.r.

3. Demuestre que existen funciones p.r., $R^-(n,i)$; $R^+(n,i)$; $S(n,i,m)$ tales que,

$$R^-(\langle n_1, \dots, n_k \rangle, i) = \begin{cases} \langle n_1, \dots, n_i \rangle & \text{si } i \leq k \\ \langle n_1, \dots, 0, \dots, 0 \rangle & \text{si } i > k \end{cases}$$

$$R^+(\langle n_1, \dots, n_k \rangle, i) = \begin{cases} \langle n_i, \dots, n_k \rangle & \text{si } i \leq k \\ 0 & \text{si } i > k \end{cases}$$

$$S(\langle n_1, \dots, n_k \rangle, i, m) = \begin{cases} \langle n_1, \dots, n_{i-1}, m, n_{i+1}, \dots, n_k \rangle & \text{si } i < k \\ \langle n_1, \dots, n_{k-1}, m \rangle & \text{si } i = k \\ \langle n_1, \dots, n_{k-1}, 0, \dots, 0, m \rangle & \text{si } i > k \end{cases}$$

4. Demuestre que si $g_1(x)$, $g_2(x)$, $h_1(x,y,z)$ son funciones p.r. entonces las funciones $f_1(x,y)$, $f_2(x,y)$ definidas simultáneamente por recurrencia "cruzada":

$$\begin{cases} f_1(x, 0) = g_1(x) \\ f_1(x, y+1) = h_1(x, y, f_2(x, y)) \end{cases}$$

$$\begin{cases} f_2(x, 0) = g_2(x) \\ f_2(x, y+1) = h_2(x, y, f_1(x, y)) \end{cases}$$

son p.r..

5.2 Recursividad de la función de Ackermann

Como ilustración de los métodos anteriores mostramos que la función de Ackermann:

$$\begin{aligned} f(0, y) &= y + 1 \\ f(x+1, 0) &= f(x, 1) \\ f(x+1, y+1) &= f(x, f(x+1, y)) \end{aligned}$$

es recursiva. Otro posible método sería exhibir una máquina de Turing que calcule a f e invocar el Teorema de la sección siguiente; dejamos esto como ejercicio. Lo que faremos será codificar los cálculos de la función de Ackermann. Como ilustración considere el cálculo, paso a paso de $f(2, 1)$:

n	$f(n)$	$C(2, 1, n)$
0	$f(2, 1)$	$<2, 1>$
1	$f(1, f(2, 0))$	$<1, 2, 0>$
2	$f(1, f(1, 1))$	$<1, 1, 1>$
3	$f(1, f(0, f(1, 0)))$	$<1, 0, 1, 0>$
4	$f(1, f(0, f(0, 1)))$	$<0, 0, 1>$
5	$f(1, f(0, 2))$	$<1, 0, 2>$
6	$f(1, 3)$	$<1, 3>$
7	$f(0, f(1, 2))$	$<0, 1, 2>$
8	$f(0, f(0, f(1, 1)))$	$<0, 0, 1, 1>$

9	$f(0, f(0, f(0, f(1, 0))))$	0, 0, 0, 1, 0	$\langle 0, 0, 1, 1 \rangle$
10	$f(0, f(0, f(0, f(0, 1))))$	0, 0, 0, 0, 1	$\langle 0, 0, 0, 1 \rangle$
11	$f(0, f(0, f(0, 2)))$	0, 0, 0, 2	$\langle 0, 0, 0, 2 \rangle$
12	$f(0, f(0, 3))$	0, 0, 3	$\langle 0, 0, 3 \rangle$
13	$f(0, 4)$	0, 4	$\langle 0, 4 \rangle$
14	5	5	$\langle 5 \rangle$
15		5	$\langle 5 \rangle$
16		5	$\langle 5 \rangle$

En la primera columna aparece el cálculo mismo, cada paso numerado de 0 en adelante. El cálculo termina en el paso 14 con el valor 5. En la segunda columna aparece lo esencial del cálculo, una sucesión de números, puesto que el símbolo f , las comas y los paréntesis no son esenciales. Podemos codificar el n -ésimo paso del cálculo por un número $C(2,1,n)$ que se indica en la tercera columna. Podemos suponer que $C(2,1,n)$ está definida para todo n , haciendo $C(2,1,n) = 5$ para $n \geq 14$. En general, si x, y son números naturales arbitrarios, sea

$$C(x, y, n) = \langle n\text{-ésimo paso en el cálculo de } f(x, y) \rangle$$

Mostraremos que $C(x, y, n)$ es una función primitivamente recursiva.

Las dos funciones definidas a continuación son p.r.

$$A(x, y) = \begin{cases} y+1 & \text{si } x = 0 \\ x-1, 1 & \text{si } x = 0, y = 0 \\ x-1, x, y-1 & \text{si } x > 0, y > 0 \end{cases}$$

$$B(x) = \begin{cases} x & \text{si } L(x) = 1 \\ \left[\prod_{k=0}^{L(x)-2} p(k)^{P(k, x)} \right] * A(P(L(x)-1, x), P(L(x), x)) & \text{si } L(x) \geq 2 \end{cases}$$

Observe que si $a = (a_1, \dots, a_n)$, incluyendo la tupla vacía,

$$B(a, x, y) = \begin{cases} \langle \bar{a}, y + 1 \rangle & \text{si } x = 0 \\ \langle \bar{a}, x - 1, 1 \rangle & \text{si } x > 0, y = 0 \\ \langle a, x - 1, x, y - 1 \rangle & \text{si } x > 0, y > 0 \end{cases}$$

es decir, si B se aplica al código del n -ésimo paso en el cálculo de $f(x,y)$ produce el código del $n+1$ -ésimo paso. En caso de que el cálculo termine en un valor y , el código será $\langle y \rangle$, y así $B(y) = \langle y \rangle$. Es claro pues que $C(x,y,n)$ puede definirse por el esquema de recurrencia primitiva:

$$\begin{cases} C(x, y, 0) = \langle x, y \rangle \\ C(x, y, n + 1) = B(C(x, y, n)) \end{cases}$$

Es claro que el cálculo de $f(x,y)$ termina la primera vez que aparece un código de longitud 1. Tenemos pues que

$$f(x,y) = P(1, C(x, y, n))$$

donde $n = \mu z(L(C(x, y, z)) = 1)$. Por tanto

$$f(x,y) = P(1, C(x, y, \mu z(|L(C(x, y, z)) - 1| = 0)))$$

Ejercicios

1. Describa una máquina de Turing que calcule la función de Ackermann.
2. Demuestre que el rango de la función de Ackermann, es decir el conjunto $\{f(x,y) | x, y \in \mathbb{N}\}$ es generado por una función p.r. (Ayuda: utilice la función $C(x, y, n)$ introducida en la sección).
3. Sea $h(x, y)$ la función que da el número de pasos necesarios para calcular $f(x, y)$ con el método obvio de substituir como se describe en la sección para $f(2,1)$. Demuestre que $h(x, y)$ crece más rápido de cualquier función p.r., es decir para cualquier función p.r. $g(x, y)$ existen $(n_1, m_1), (n_2, m_2), (n_3, m_3), \dots$ con $n_1 < n_2 < n_3 < \dots; m_1 < m_2 < m_3 < \dots$ tales que $h(n_i, m_i) > g(n_i, m_i)$ para todo i . (Ayuda: $h(x, y) = n(C(x, y, n) = C(x, y, n+1))$)

5.3 Toda función Turing-calculable es recursiva

Demostramos en esta sección que toda función numérica calculable por una máquina de Turing es recursiva. Como ya hemos mostrado el converso, tenemos que para una función $f: \mathbb{N}^k \rightarrow \mathbb{N}$ ser calculable por una máquina de Turing es equivalente a ser recursiva. Ahora, es difícil concebir una noción de algoritmo más comprensiva que la de máquina de Turing, en particular todo lo que un gran computador moderno puede hacer ya se puede hacer por medio de una máquina de Turing, y en realidad un computador se puede considerar como una máquina de Turing universal. Lo anterior da evidencia para la *Tesis de Church* de que las funciones recursivas *son* las funciones numéricas calculables en el sentido intuitivo de la palabra "calculable". La verdad es que todas las definiciones que han sido propuestas para la noción de calculabilidad: Algoritmos de Markov, Sistemas de Post, Sistemas ecuacionales de Herbrand-Gödel, diversos lenguajes de programación, etc., todos conducen a la misma clase de funciones recursivas.

Para llegar al mencionado resultado utilizaremos una idea debida originalmente a Gödel en otro contexto (Teorema de incompletitud) de codificar numéricamente las máquinas de Turing y todos sus cálculos.

Sea T una máquina de Turing con conjunto de estados K y alfabeto:

$$K = \{q_0, \dots, q_k\}, \Sigma = \{s_1, \dots, s_m\}$$

Podemos hacer la identificación siguiente entre estados o símbolos y números:

$$q_i \leftrightarrow i \quad i = 0, \dots, k$$

$$\square \leftrightarrow 0$$

$$s_i \leftrightarrow i \quad i = 1, \dots, m$$

$$R \leftrightarrow 2$$

$$L \leftrightarrow 0$$

De esta manera, las instrucciones de T pueden representarse por tres funciones de algún subconjunto de $\{0, \dots, k\} \times \{0, \dots, m\}$ en \mathbb{N} . Si T contiene la instrucción $q_i s_e s_r M q_j$ las funciones S, Q y M están definidas en la pareja (i, e) por

$$\begin{array}{ll} S(i, e) = r & \text{Función símbolo impreso} \\ Q(i, e) = j & \text{Función próximo estado} \\ M(i, e) = \begin{cases} 2 & \text{si } M = R \\ 0 & \text{si } M = L \end{cases} & \text{Función movimiento} \end{array} \quad (1)$$

Estas funciones pueden extenderse a todo \mathbb{N}^2 definiendo en las parejas restantes:

$$\begin{array}{ll} S(i, e) = e \\ Q(i, e) = i \\ M(i, e) = 1 \end{array} \quad (2)$$

Estas funciones son p.r. pues se pueden considerar como definidas por casos trivialmente recursivos. Por ejemplo, si de la máquina T tenemos que $S(i, e) \neq r_{i,e}$ para $(i, e) \in \{0, \dots, K\} \times \{0, \dots, m\}$ entonces la extensión de S está definida por:

$$S(i, e) = \begin{cases} r_{i,e} & \text{si } 0 \leq i \leq k, 0 \leq e \leq m \\ e & \text{de lo contrario} \end{cases}$$

Supongamos que T trabaja en una semicinta infinita a la derecha con las celdas numeradas. Podemos suponer que T no recibe nunca orden de moverse a la izquierda de la celda numerada 0. La configuración:

C:

	0	1	2	\dots	P	\dots	K	\dots
	s_{e0}	s_{e1}	s_{e2}	\dots	s_{ep}	\dots	s_{ek}	\dots

\uparrow
 q_i

donde las celdas a la derecha de la k -ésima están en blanco puede codificarse por un número natural

$$\hat{C} = \langle i, p, \langle e_0, e_1, \dots, e_k \rangle \rangle$$

i representa el estado, p la celda leída y $\langle e_0, e_1, \dots, e_k \rangle$ codifica el contenido de la cinta.

Si $T: C \xrightarrow{1} C'$ decimos que C' es la configuración *sucesora* de C . Del significado de las funciones Q, S, M tenemos que

$$\hat{C}' = Q(i, e_p, [p + M(i, e_p)] \div 1, \langle e_0, \dots, e_{p-1}, S(i, e_p), e_{p+1}, \dots, e_k \rangle)$$

ya que el nuevo estado es $Q(i, e_p)$, el símbolo s_{e_p} se cambia por

$$s_{S(i, e_p)} \text{ y } (p + M(i, e_p)) \div 1 \text{ es } p + 1 \text{ ó } p - 1$$

según que $M(i, e_p)$ sea 2 ó 0.

Lema 1. Existe una función p.r., $H: \mathbb{N} \rightarrow \mathbb{N}$ tal que

$$H(\hat{c}) = \hat{C}' \text{ si } T: C \xrightarrow{1} C'.$$

Además

$$H(\hat{c}) = \hat{c} \text{ si } T \text{ se detiene en la configuración } C.$$

Demostración. Para simplificar la notación escribimos $(n)_i = P(i, n)$, la función definida en la sección anterior. Si $n = \langle i, p, \langle e_0, \dots, e_k \rangle \rangle$ es claro que

$$i = (n)_1, p = (n)_2 \text{ y } e_p = ((n)_3)_{(n)2}$$

Entonces podemos definir:

$$H(n) = \langle Q(n)_1, (n)_3)_{(n)2}, [(n)_2 + M((n)_1, ((n)_3)_{(n)2})] \div 1, R(n) \rangle$$

donde

$$R(n) = \left[\prod_{j=0}^{(n)_2-1} P(j)^{(n)_3 j} \right] P((n)_2)^{S((n)_1, ((n)_3)(n)_2)} \prod_{j=(n)_2+1}^{L((n)_3)} P(j)^{(n)_3 j}$$

Como H es compuesta de funciones p.r., es p.r.. Supongamos ahora que la máquina T se detiene en la configuración C , entonces no existe instrucción que comience con $q_i s_p$. Por lo tanto, de acuerdo con (2) tenemos:

$$Q(i, e_p) = i$$

$$S(i, e_p) = e_p$$

$$M(i, e_p) = 1$$

y así

$$\begin{aligned} H(\hat{c}) &= Q(i, e_p), [p + M(i, e_p)] \doteq 1, \langle e_0, \dots, e_{p-1}, , e_p, e_{p+1}, \dots, e_k \rangle \\ &= \langle i, p, \langle e_0, \dots, e_k \rangle \rangle = \hat{c}. \blacksquare \end{aligned}$$

Recuérdese que hemos hecho la identificación: $\square \leftrightarrow 0$, si $\leftrightarrow i$, $i = 1, \dots, m$,

De esta manera, podemos asociar a cada palabra sobre el alfabeto $\Sigma = \{s_1, \dots, s_m\}$ un código o *número de Gödel*.

Si $w = s_{i1} \dots s_{ik}$ entonces

$$c(w) = \hat{w} = \langle i_1, \dots, i_k \rangle$$

Igualmente a cada función (posiblemente parcial) de Σ^* en Σ^* podemos asociar una función numérica:

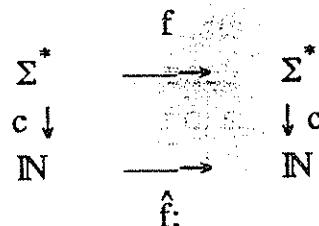
$$\hat{f}: \mathbb{N} \rightarrow \mathbb{N}$$

(posiblemente parcial) tal que:

$$\hat{f}(\hat{w}) = \widehat{f(w)}$$

si $f(w)$ está definida, e indefinida de lo contrario. Evidentemente

$$\hat{f} = c \cdot f \cdot c^{-1}:$$



Teorema 2. Si f es una función T -calculable de Σ^* en Σ^* (posiblemente parcial), entonces $\hat{f}: \mathbb{N} \rightarrow \mathbb{N}$ es recursiva (posiblemente parcial).

Dem Sea T una máquina que calcula a f en una semicinta infinita a la derecha. Podemos suponer que $f(w)$ aparece a partir de la celda 0 cuando está definida. Introducimos la función:

$$G(d, 0) = \langle 0, , 0, d \rangle$$

$$G(d, n + 1) = H(G(d, n))$$

en donde H es la función del Lema 1. Si $d = \hat{w}$ entonces $G(d, n)$ da el código de la configuración de T en el n -ésimo paso del cálculo de $f(w)$. Es claro que $\langle 0, 0, d \rangle$ codifica a la máquina en el estado q_0 leyendo la celda 0 cuando w está escrita comenzando en la celda 0. El resto es obvio de las propiedades de H .

$f(w)$, está definida si y sólo si existe n tal que $G(d, n)$ codifica una configuración en la cual T se detiene. Por el Lema 1, en este caso $H(G(d, n + 1)) = G(d, n)$. Además, si k codifica una configuración C que no es de parada entonces C y C' difieren por lo menos en la posición de la cabeza lectora y así $(H(k))_2 = p \pm 1 \neq p = (k)_2$, lo cual significa $H(k) \neq k$.

De esta manera, si $f(w)$ está definida, y sólo si $f(w)$ está definida, existe n tal que

$$G(d, n + 1) = H(G(d, n)) = G(d, n).$$

Defina

$$N(d) = (G(d, \mu n(G(d, n+1) = G(d, n)))_3.$$

Entonces N es parcialmente recursiva, y está definida para $d = \hat{w}$ si y solo si $f(w)$ está definida. Además, en el último caso

$$\begin{aligned} N(\hat{w}) &= (\text{Código de la configuración final de } T \text{ en el cálculo de } f(W))_3 \\ &= (\langle i, p, \widehat{f(w)} * \langle 0, \dots, 0 \rangle \rangle)_3 \\ &= \widehat{f(w)} * \langle 0, \dots, 0 \rangle \end{aligned}$$

para algún i, p , donde los 0's adicionales representan posibles codificaciones de blancos. Si K es una función p.r. tal que

$$K(\langle i_0, i_1, \dots, i_m \rangle = \langle i_0, \dots, i_k \rangle)$$

donde i_k es el último número diferente de 0, entonces

$$K(N(\hat{w})) = \widehat{f(w)} = \hat{f}(\hat{w}). \blacksquare$$

Lema 3. La función $h_k: \mathbb{N}^k \rightarrow \mathbb{N}$ tal que

$$h_k(n_1, \dots, n_k) = \zeta(\bar{n}_1 \square \bar{n}_2 \square \dots \square \bar{n}_k),$$

donde ζ es el número de Gödel de una expresión, es primitivamente recursiva.

Demostración. Suponga $\square \leftrightarrow 0, / \leftrightarrow 1$. La función $h_1(n) = \zeta(\bar{n})$ es p.r. pues

$$h_1(n) = [\prod_{j=0}^n p(j)] p(n) = \underbrace{\langle 1, \dots, 1 \rangle}_{n+1 \text{ veces}}$$

Por lo tanto, h_k está dada por

$$h_k(n_1, \dots, n_k) = h(n_1) * \langle 0 \rangle * H(n_2) * \langle 0 \rangle * \dots * \langle 0 \rangle * h(n_k)$$

y es una composición de p.r. \blacksquare

Colorario 4 Si $f: \mathbb{N}^k \rightarrow \mathbb{N}$ es T -calculable entonces f es recursiva.

Demostración Supongamos que f es T -calculable en notación monádica es decir la función $\bar{f}: \Sigma^* \rightarrow \Sigma^*$

$$\bar{f}(\bar{n}_1 \square \bar{n}_2 \square \dots \square \bar{n}_k) = \overline{f(n_1, \dots, n_k)}$$

es T -calculable. Entonces

$$\hat{f} = \varsigma \circ \bar{f} \circ \varsigma^{-1}: \varsigma(\bar{n}_1 \square \dots \square \bar{n}_k) \mapsto \varsigma(\overline{f(n_1, \dots, n_k)})$$

es una función recursiva por el Teorema 2. Por el lema 3 la función

$$\begin{aligned} f &= h_1^{-1} \circ \hat{f} \circ h_k: (n_1, \dots, n_k) \mapsto \varphi(\bar{n}_1 \square \dots \square \bar{n}_k) \\ &\mapsto f \circ (\overline{f(n_1, \dots, n_k)}) \mapsto f(n_1, \dots, n_k) \end{aligned}$$

es recursiva, como compuesta de recursivas. n

Ejercicios

1. Muestre que $f: \Sigma^* \rightarrow \Sigma^*$ es T -calculable si y solo si $\hat{f}: \mathbb{N} \rightarrow \mathbb{N}$ es parcialmente recursiva.
- 2.* Existe una enumeración de todas las funciones recursivas parciales de \mathbb{N} en \mathbb{N} :

$$g_0, g_1, \dots$$

y una función recursiva $\mu(x, y)$ tal que

$$\mu(n, m) = g_n(m).$$

(Ayuda: Existe una enumeración de todas las máquinas de Turing sobre el alfabeto $\{s, /\}$ tal que la función

$$\bar{n} \mapsto \bar{T}_n$$

es calculable por una máquina de Turing. Si U es una máquina de Turing universal entonces es posible construir una máquina que realice las siguientes transformaciones:

$$\bar{n} * \bar{m} \vdash \bar{T}_n * \bar{m} \vdash f_U(\bar{T}_n * \bar{m}) = T_n(\bar{m})$$

si g_n es la función calculada por T_n , entonces $T_n(\bar{m}) = \bar{g}_n(m)$, como $\bar{n} * \bar{m} \vdash \bar{g}_n(m)$ es T-calculable, debe ser recursiva la función correspondiente: $\mu(n,m) = g_n(m)$.

3. Sea $\Sigma = \{s_0, \dots, s_N\}$ y defina $g: \Sigma^* \rightarrow \mathbb{N}$

$$g(a_{i0} \dots a_{ik}) = i_0 + i_1 N + i_2 N^2 + \dots + i_{k-1} N^{k-1} + (i_k + 1) N^k$$

Muestre que g es una biyección con $\mathbb{N} - \{0\}$. Demuestre que $f: \Sigma^* \rightarrow \Sigma^*$ es T-calculable si y solo si $g \circ f \circ g^{-1}$ es recursiva.

4. Sea f una función recursiva parcial, demuestre que los conjuntos siguientes son recursivamente enumerables:

$$\text{Dom}(f) = \{x \mid f(x) \text{ no está definida}\}$$

$$\text{Im}(f) = \{f(x) \mid f(x) \text{ está definida}\}$$

5. Si $E \subseteq \mathbb{N}$ es recursivamente enumerable, existe $g: \mathbb{N} \rightarrow \mathbb{N}$ primitivamente recursiva que enumera a E .

(Ayuda: note que en la prueba del Teorema 2, la función $G(d,n)$ es tal que si $G(d,n+1) = G(d,n)$, aunque n no sea el mínimo con tal propiedad, entonces

$$f(d) = K((G(d,n))_3).$$

Ahora, si E es enumerado recursivamente por una función f_T calculada por una máquina T , entonces como en el Colorario 4,

$$g(x) = h_1^{-1}(f_T(h_1(x))) = h_1^{-1}(K((G(h_1(x), n))_3))$$

6. La función $h(d) = \mu n(G(d, n+1) = G(d, n))$ introducida en la prueba del Teorema 2 da el número de pasos necesarios para calcular $f(d)$ con la máquina de Turing T . Demuestre:
- Si $h(d)$ está acotada por una función primitivamente recursiva entonces f es primitivamente recursiva.
 - Si \hat{f} no es primitivamente recursiva entonces h crece más rápido que cualquier función p.r. Es decir, para cualquier función p.r. $g(x)$ existen $n_1 < n_2 < n_3 < \dots$ tales que $h(n_i) > g(n_i)$ para toda i .

5.4 Forma normal de Kleene

De la demostración de la recursividad de las funciones Turing-calculables podemos extraer una función universal que genera todas las funciones (parcialmente) recursivas. Sea T una máquina de Turing con conjunto de estados K y alfabeto Σ . Podemos suponer:

$$K \subseteq \{q_0, q_1, \dots\}, \Sigma \subseteq \{s_1, s_2, s_3, \dots\}$$

Para cada instrucción $I = q_i s_e s_r R q_j$ podemos definir un código:

$$\hat{I} = \langle i, e, r, 2, j \rangle$$

Igualmente $I = q_i s_e s_r L q_j$ tendrá código

$$\hat{I} = i, s, r, 0, j.$$

De esta manera podemos asociar a toda la máquina T un código

$$t = \hat{T} = \hat{I}_1, \dots, \hat{I}_k$$

donde I_1, \dots, I_k son todas las instrucciones de T . Las funciones S , Q , M , introducidas en la sección 5.2 pueden ahora definirse en función de t .

Haciendo

$$h(t, i, e) = \mu z[((t)_z)_1 = i \wedge ((t)_z)_2 = e],$$

la función que da el código de la instrucción que comienza por $q_i s_e$ si tal instrucción existe, ó 0 si no existe, tenemos:

$$S(t, i, e) = \begin{cases} (h(t, i, e))_3 & \text{si } h(t, i, e) > 0 \\ e & \text{si } h(t, i, e) = 0 \end{cases}$$

$$Q(t, i, e) = \begin{cases} (h(t, i, e))_5 & \text{si } h(t, i, e) \\ i & \text{de lo contrario} \end{cases}$$

$$M(t, i, e) = \begin{cases} (h(t, i, e))_4 & \text{si } h(t, i, e) > 0 \\ 1 & \text{de lo contrario} \end{cases}$$

Estas funciones resultan primitivamente recursivas.

Igual que en la prueba del Teorema 2, sección 5.2, podemos definir una función primitivamente recursiva $H(t, n)$ tal que para cualquier configuración instantánea C de la cinta:

$$H(t, \hat{C}) = \begin{cases} \hat{C}' & \text{si } T: C \vdash C' \\ \hat{C} & \text{si } T \text{ se detiene en } C \end{cases},$$

Podemos entonces definir la función p.r.

$$\begin{cases} G(t, d, 0) = 0, 0, d \\ G(t, d, n+1) = H(G(t, d)) \end{cases}$$

tal que $g(t, \hat{w}, n)$ da el código de la configuración de la cinta en el paso n del cálculo de T inicializada leyendo el primer símbolo de w en el estado q_0 , y $G(t, w, n) = G(t, w, n+1) = G(t, w, n+2) = \dots$, si T se detiene en el paso n .

Finalmente tenemos que si f_T es la función (posiblemente parcial) de Σ^* en Σ^* calculada por T , entonces:

$$f_T(d) = K[(G[t, d, \mu n(G(t, d, n+1) = G(t, d, n))])_3] \quad (1)$$

donde K es cierta función p.r. que "borra blancos".

Teorema 1 (Forma normal de Kleene). Existen funciones primitivamente recursivas $u(x)$ y $k(x, y, z)$ tales que para toda función recursiva (parcial) f de \mathbb{N} en \mathbb{N} existe algún t tal que:

$$f(n) = u(\mu z(k(t, n, z) = 0))$$

Demostración. Haciendo referencia a la ecuación (1) arriba, sea

$$G'(x) = K[(G((x)_1, (x)_2, (x)_3))_3]$$

entonces

$$\begin{aligned} f_T(d) &= G'[\langle t, d, \mu n(G(t, d, n+1) = G(t, d, n)) \rangle] \\ &= G'[\mu z((z)_1 = t \wedge (z)_2 = d \wedge G(t, d, (z)_3 + 1) = G(t, d, (z)_3))] \end{aligned}$$

puesto que $\langle t, d, n \rangle$ es creciente en n . La condición minimizada es primitivamente recursiva, luego existe una función p.r. $g(x, y, z)$ tal que:

$$f_T(d) = G'[\mu z(g(t, d, z) = 0)].$$

Ahora, hemos visto en la sección 5.2 que si T es una máquina que calcula a f en monádico, es decir $f_T = \bar{f}$, entonces $f = h_1^{-1} \circ f_T \circ h_1$, por lo tanto:

$$f(n) = h_1^{-1}(G'(\mu z(g(t, h_1(n), z) = 0)))$$

haciendo $u(x) = h_1^{-1}(G'(x))$ y $k(x, y, z) = g(x, h_1(y), z)$ tenemos el resultado deseado. ■

La función

$$U(t, x) = u(\mu z(t(t, x, z) = 0))$$

es recursiva parcial, y lo es también como función de x para cada t fijo. Por lo tanto el teorema anterior implica que

$$U(0, x), U(1, x), U(2, x), \dots$$

es una enumeración de *todas* las funciones recursivas parciales (incluyendo por supuesto las totales) de una variable. Podemos dar una versión del problema de la parada para funciones parciales recursivas. Antes tenemos:

Teorema 2 *Sea $f(x)$ una función recursiva parcial, entonces $\text{Dom}(f) = \{x \in \mathbb{N} \mid f(x) \text{ está definida}\}$ es recursivamente enumerable.*

*Demuestra*ción Sea $f(x) = u(\mu z(k(t, x, z) = 0))$, entonces es evidente que $\text{Dom}(f) = \{x \in \mathbb{N} \mid \exists z (k(t, x, z) = 0)\}$ y este conjunto es r. e. por lo que k es recursiva (ver Cap. 2, Sec. 2.5). ■

Pero el dominio de una función recursiva parcial no es necesariamente recursivo.

Teorema 3 *Existe una función recursiva parcial cuyo dominio no es recursivo.*

*Demuestra*ción Sea $f(x) = U(x, x)$. Esta función es recursiva parcial. Suponga por contradicción que $\text{Dom}(f)$ es recursivo y defina

$$g(x) = \begin{cases} f(x) + 1 & \text{si } x \in \text{Dom}(f) \\ 0 & \text{de lo contrario} \end{cases}$$

Como $g(x)$ es también recursiva, pues ha sido definida por casos recursivos, entonces debe existir t tal que $g(x) = U(t, x)$. En particular $g(t) = U(t, t)$. Como g es total por definición entonces $U(t, t)$ está definida y así $t \in \text{Dom}(f)$. De acuerdo a la definición de g ésto significa

$$g(t) = f(t) + 1 = U(t, t) + 1$$

una contradicción. ■

De la prueba del teorema tenemos el siguiente ejemplo explícito de un conjunto recursivamente enumerable que no es recursivo:

$$\{ n \in \mathbb{N} \mid \exists z (k(n, n, z) = 0) \}$$

es decir la proyección de un conjunto primitivamente recursivo puede no ser recursiva. También nos da el Teorema 3 una nueva prueba de la insolubilidad del problema de la parada.

Colorario Existe una máquina de Turing T para la cual el problema de la parada es indecidible (por medio de máquinas de Turing).

Demostración Sea T una máquina de Turing que calcula la función recursiva parcial $f(x) = U(x, x)$, como su dominio no es recursivo su función característica no puede ser calculada por una máquina de Turing. ■

Ejercicios

1. Enuncie y demuestre la existencia de la forma normal de Kleene para funciones de k variables.
- 2.* Demuestre que *no* existe una función recursiva $H(t, x)$ tal que

$$H(0, x), H(1, x), H(2, x), \dots$$

sea una enumeración de *todas* las funciones recursivas de una variable.

- 3.** Demuestre que existe una función $P_r(t, x)$, recursiva tal que

$$P_r(0, x), P_r(1, x), P_r(2, x), \dots$$

es una enumeración de *todas* las funciones primitivamente recursivas de una variable (pero P_r no puede ser primitivamente recursiva).

4. Sea $h(x) = U(x, x) + 1$, esta función es recursiva parcial. Demuestre que h no puede ser extendida a una función recursiva total.

- 5.* Demuestre que existe una función primitivamente recursiva $E(t, x)$ tal que si se define $E_n = \{ E(n, x) \mid x \in \mathbb{N} \}$ entonces:

$$E_0, E_1, E_2, \dots$$

es una enumeración de *todos* los conjuntos r. e. de \mathbb{N} .

6. Demuestre que no existe función recursiva $R(t, x)$ tal que los conjuntos $R_n = \{ R(n, x) \mid x \in \mathbb{N} \}$ enumeren exactamente los subconjuntos recursivos de \mathbb{N} .

7. Demuestre que si f es recursiva parcial de \mathbb{N} en \mathbb{N} entonces $L_n(f) = \{f(x) \mid f(x) \text{ está definida}\}$ es r. e..
8. Demuestre que el conjunto $\{t \mid f(x) = U(t,x) \text{ es total}\}$ no es recursivamente enumerable (y por lo tanto no es recursivo). Ayuda: vea problema 2.
9. Demuestre que toda función recursiva parcial *total* es recursiva.

Bibliografía

Boolos, George; Jeffrey, Richard. *Computability and Logic..* Cambridge University Press, 1974.

Caicedo, Xavier. *Extensiones del Cálculo de Predicados de Primer Orden,* Lecturas Matemáticas, S.C.M. Vol. 1, No. 2, 1980.

Church, A. *Introduction to Mathematical Logic.* Vol. I. Princeton, NJ, 1956.

Ebbinghaus, H.D.; Flum, J.; Thomas, W.. *Mathematical Logic.* Springer Verlag, 1985.

Davis, Martin. *Hilbert's tenth problem is unsolvable.* American Mathematical Monthly, March 1973, pp 233-269.

Deaño, Alfredo. *Introducción a la lógica formal.* Alianza Editorial, 1980.

De Long, Howard. *A Profile of Mathematical Logic.* Addison Wesley, 1971.

Gödel, Kurt. *Obras completas.* Alianza Editorial, 1981.

Van Heijenoort, Jean. *From Frege to Gödel.* (Editor) Harvard University Press, 1967.

- Hamilton, A.G. *Logic for Mathematicians*. Cambridge University Press (revised edition), 1988.
- Hatcher, William S. *Foundations of Mathematics*. W.B. Saunders Company, 1968.
- Hermes, Han. *Enumerability, decidability, computability*. Springer Verlag, Academic Press, 1965.
- Hilbert, David; Ackermann, Wilhem. *Elementos de Lógica Teórica*. Editorial Tecnos, 1962.
- Hopcroft, J.E.; Ullman, J.D. *Formal Languages and their relation to automata*. Addison Wesley, 1969.
- Huges, G.E; Cresswell, M.J. *Introducción a la lógica modal*. Editorial Tecnos, 1973.
- Kleene, Stephen C. *Mathematical Logic*. John Wiley and Sons, 1967.
- Stephen, C. Kleene. *Introducción a la Metamatemática*. Editorial Tecnos Madrid, 1974.
- Kneebone, G.T. *Mathematical Logica and the foundations of Mathematics*. Van Nostrand 1963.
- Manin, Y.I. *A course in Mathematical Logic*. Springer Verlag, 1977.
- Malitz, J. *Introduction to Mathematical Logic*. Springer Verlag, 1979.
- Manna, Z. *Mathematical Theory of Computation*, Mc Graw-Hill, 974.
- Mendelson, Eliot. *Introduction to Mathematical Logic*. Van Nostrand Reinhold, 1964.
- Robbin, Joel W. *Mathematical Logic, a first course*. Benjamin, 1969.

Rotman, B.; Kneebone, G.T. *The theory of sets and transfinite numbers.*
Van Nostrand, 1966.

Trakhtenbort, B.A. *Algoritmos y computadores.* Limusa Mexico, 1973.

Van Dalen, Dirk. *Logic and Structure.* Springer Verlag, 1983.

Whitehead, Alfred Northon; Russell, Bertrand . *Principia Mathematica to
56. Cambridge University Press, 1967.

