Lógica

José Luis Tábara

jltabara@gmail.com

Índice

1. Lenguajes formales **3**

2. Lenguaje de la lógica proposicional **11**

3. Semántica del lenguaje proposicional **23**

4. Equivalencia y álgebras de Boole **28**

5. Consecuencia lógica. Compacidad **34**

6. Sistemas deductivos **41**

7. Completitud de la lógica proposicional **48**

8. Lenguajes de primer orden. Términos **52**

9. Lenguajes de primer orden. Fórmulas **60**

10.Sistemas **66**

11.Semántica **72**

12.Ejemplos semánticos **81**

13.Modelos **84**

14.Teorías **89**

15.Cálculo deductivo **97**

16.Consistencia **99**

17.Teorema de completitud **101**

18.Bibliografía **104**

1. Lenguajes formales

Nuestro objetivo es formalizar la noción de lenguaje escrito. A efectos prácticos, una expresión de un lenguaje escrito es simplemente una cadena de símbolos escritos consecutivamente. Cada uno de esos símbolos se llama *letra* y el conjunto de todas las letras utilizadas en un lenguaje se denomina *alfabeto*. Todos los alfabetos de las lenguas naturales tienen un número finito de letras, pero nada nos impide imaginar alfabetos que tengan infinitos símbolos.

Definición 1.1 *Llamamos* `alfabeto` *a cualquier conjunto A.*

Denotaremos por A^n al producto cartesiano de n copias de A. Por convenio supondremos que $A^0 = \varnothing$.

Lema 1.1 *Si A es un alfabeto finito entonces A^n es un conjunto finito.*

Demostración.

Sea A un conjunto finito. Entonces $|A^n| = |A|^n$ (las barras verticales denotan el número de elementos de un conjunto). □

Además de los alfabetos finitos, son muy utilizados los alfabetos numerables. Para poder hablar de ellos con precisión debemos recordar el concepto de conjunto numerable y algunos criterios de numerabilidad.

Definición 1.2 *Un conjunto A es* `numerable` *si existe una biyección*

$$f : A \mapsto \mathbb{N}$$

El conjunto de los enteros pares $2\mathbb{N}$ es numerable, puesto que la función

$$\begin{array}{rccc} f: & \mathbb{N} & \mapsto & 2\mathbb{N} \\ & n & \mapsto & 2n \end{array}$$

es claramente una biyección. En realidad todos los subconjuntos de $\mathbb{N}$ que no son finitos, son también numerables, tal como demostramos a continuación.

Lema 1.2 *Sea A un conjunto no finito y $f : A \mapsto \mathbb{N}$ una aplicación inyectiva. Entonces A es numerable.*

Demostración.

Definimos una aplicación $g : A \mapsto \mathbb{N}$ de la siguiente forma: $g[1]$ es el menor elemento de $f[A]$. Eliminamos dicho elemento del conjunto imagen, obteniendo el conjunto $A_2 = f[A] - \{g[1]\}$. Definimos entonces $g[2]$ como el menor elemento de A_2. En general, construimos el conjunto

$$A_n = f[A] - \{g[1]\} - \cdots - \{g[n-1]\}$$

y definimos $f[n]$ como el menor elemento de A_n. La aplicación g de A en $\mathbb{N}$ que es claramente inyectiva. Además es epiyectiva puesto que el conjunto A_n nunca es vacío. □

Siguiendo un proceso similar al anterior se puede probar el siguiente

Lema 1.3 *Sea $f : \mathbb{N} \mapsto A$ una aplicación epiyectiva. Entonces A es finito o numerable.*

Corolario 1.4 *El conjunto $\mathbb{N} \times \mathbb{N}$ es numerable. En general el conjunto $\mathbb{N}^r$ es numerable.*

Demostración.

Consideramos la aplicación

$$\begin{array}{rccc} f: & \mathbb{N} \times \mathbb{N} & \mapsto & \mathbb{N} \\ & (n, m) & \mapsto & 2^n \cdot 3^m \end{array}$$

Claramente es inyectiva y por el lema anterior (puesto que el conjunto no es finito) el conjunto es numerable.

Para el caso general se procede por inducción, utilizando el hecho de que $\mathbb{N}^r = \mathbb{N}^{r-1} \times \mathbb{N}$. □

En el lenguaje natural un conjunto finito de letras colocadas consecutivamente se denomina expresión.

Definición 1.3 *Si A es un alfabeto denotamos por A^* al conjunto*

$$A^* = \bigcup_{i=0}^{\infty} A^n$$

Los elementos de A^* se llaman `expresiones`.

En general es costumbre en matemáticas denotar a cualquier elemento de A^n mediante $(a_1, \ldots, a_n)$. Sin embargo nosotros suprimiremos los paréntesis y las comas y escribiremos dicho elemento en la forma $a_1 \ldots a_n$. De esta forma queda más claro la denominación de expresión que hemos adjudicado a los elementos de A^*.

Los conjuntos A^n y A^m son disjuntos si $n \neq m$. Dado un elemento $\alpha \in A^*$, tenemos que $\alpha \in A^n$ para un único n. Diremos entonces que α es una expresión de `longitud` n. Denotaremos este hecho mediante $\mathsf{long}[\alpha] = n$. Como hemos decidido que $A^0 = \varnothing$ esté contenido en A^*, existe una única palabra de longitud nula. Para no confundirla con el conjunto vacío la denotaremos por $\otimes$. Las palabras de longitud 1 se pueden identificar con los elementos del alfabeto.

Lema 1.5 *Una unión numerable de conjuntos finitos o numerables es numerable.*

Demostración.

Sean A_i la colección de conjuntos. Denotemos por $a_{i1}, a_{i2}, \ldots$ los elementos del conjunto A_i. Esto es posible hacerlo puesto que A_i es finito o numerable. Consideremos la aplicación

$$\begin{array}{rccc} f: & \mathbb{N} \times \mathbb{N} & \mapsto & \bigcup_{i=1}^{\infty} A_i \\ & (i,j) & \mapsto & a_{ij} \end{array}$$

Por construcción esta aplicación es epiyectiva. Como $\mathbb{N} \times \mathbb{N}$ es numerable se concluye por los resultado anteriores. $\square$

Proposición 1.6 *Si A es un alfabeto finito o numerable, el conjunto A^* es siempre numerable.*

Demostración.

Si A es finito, ya hemos comentado que A^n es finito. Como el conjunto A^* es una unión numerable de conjuntos finitos, necesariamente es numerable.

Si el alfabeto es numerable, entonces A^* es una unión numerable de conjuntos numerables, y por lo tanto también es numerable. □

Observación. Como lo que nos va a interesar de un alfabeto es su conjunto de expresiones, debido a este resultado existe muy poca diferencia entre considerar alfabetos finitos o considerar alfabetos numerables. □

Definición 1.4 *Dado un alfabeto A, llamamos* `lenguaje` *sobre A a cualquier conjunto $L \subset A^*$.*

Si el alfabeto es finito o numerable, entonces L es finito o numerable, puesto que está contenido en un conjunto numerable.

Ejemplos.

- Sea $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Consideremos el lenguaje

$$L_1 = \{\alpha \in A^* \text{ tales que } \mathsf{long}[\alpha] < 100\}$$

Este lenguaje es finito. Sin embargo si consideramos el lenguaje

$$L_2 = \{\alpha \in A^* \text{ tales que su primera cifra es el } 0\}$$

tenemos un lenguaje de cardinal infinito. En el lenguaje

$$L_3 = \{\alpha \in A^* \text{ tales que su primera cifra no es } 0\}$$

es claro como se puede establecer de manera efectiva una biyección del lenguaje con el conjunto de los naturales.

- Dado un alfabeto A, si $L_1 \subset L_2$ decimos que L_1 es un `sublenguaje` de L_2, o bien que L_2 es una `extensión` de L_1. Si tenemos dos alfabetos A y A' tales que $A \subset A'$ tenemos definiciones similares.

- En informática el alfabeto más utilizado es sin duda el $\{0, 1\}$. Cada uno de sus elementos se llaman *bits*. Una «letra» en informática es una sucesión de 8 bits, que es lo que se denomina *byte*. Existen 256 bytes distintos. Las expresiones en informática son cadenas de bytes. Esto es, son expresiones formadas por ceros y unos, pero siempre de longitud múltiplo de 8.

Dado un alfabeto A y dos palabras α y β denotamos por $\alpha\beta$ a la palabra que consiste en colocar primeramente todos los símbolos de α y a continuación todos los de β. Esta definición da lugar a una operación

$$\begin{array}{rcl} A^* \times A^* & \to & A^* \\ (\alpha, \beta) & \to & \alpha\beta \end{array}$$

llamada `concatenación`. De la definición dada se deduce rápidamente la asociatividad de esta operación. El elemento neutro de esta operación es la palabra vacía $\otimes$. Además siempre se cumple que $\mathsf{long}[\alpha\beta] = \mathsf{long}[\alpha] + \mathsf{long}[\beta]$.

Observación. En matemáticas se llama `semigrupo` a un conjunto dotado de una operación asociativa y con elemento neutro. Entonces el conjunto de expresiones es un semigrupo. La función longitud es un morfismo entre este semigrupo y el semigrupo $(\mathbb{N}, +)$. $\square$

Definición 1.5 *Decimos que α es una* `subexpresión` *de β si existen expresiones, no necesariamente únicas, γ y δ tales que $\beta = \gamma\alpha\delta$. Decimos que α es una* `parte inicial` *o* `prefijo` *de β si existe una expresión γ tal que $\beta = \alpha\gamma$.*

La operación concatenación nos permite definir subconjuntos que son invariantes por su acción.

Definición 1.6 *Decimos que un lenguaje $L \subset A^*$ es* `cerrado por concatenación` *si para todo $\alpha, \beta \in L$ se tiene que $\alpha\beta \in L$.*

Naturalmente el conjunto A^* es cerrado por concatenación. Del mismo modo es claro que el conjunto vacío es también cerrado por concatenación.

Proposición 1.7 *Dado un alfabeto A y una colección cualquiera $\{L_i\}_{i\in I}$ de lenguajes cerrados por concatenación, entonces el conjunto intersección $L = \bigcap_{i\in I} I_i$ es cerrado por concatenación.*

Demostración.

Sean $\alpha, \beta \in L$. Entonces $\alpha, \beta \in L_i$ para todo índice i. Como L_i es cerrado por concatenación, tenemos que $\alpha\beta \in L_i$. Como este resultado es cierto para todo i, tenemos que dicho elemento también pertenece a la intersección de los conjuntos. Concluimos que $\alpha\beta \in L$, lo que prueba que L es cerrado por concatenación. □

En principio, puede ocurrir en la proposición anterior que el conjunto intersección sea el conjunto vacío, aun cuando ninguno de los conjuntos L_i sea un conjunto vacío. Para evitar esto, lo «mejor» es poner algunos elementos en la intersección.

Definición 1.7 *Dado un conjunto $B \subset A^*$, llamamos* `conjunto generado por concatenación` *a partir de B a la intersección de todos los conjuntos cerrados por concatenación que contienen a B. Lo denotaremos $\langle B \rangle$.*

Como $B \subset A^*$ y A^* es cerrado por concatenación, siempre existe al menos un conjunto cerrado por concatenación y que contiene a B, por lo que la definición siempre tiene sentido.

En el caso de conjuntos generados, la intersección tiene siempre algún elemento y el conjunto generado nunca es vacío. También es clara la inclusión $B \subset \langle B \rangle$. Si el conjunto B no es cerrado por concatenación la inclusión es estricta. En el caso en que B ya sea cerrado por concatenación se tiene la igualdad.

Ejemplos.

- Si $L_1 \subset L_2$ tenemos también la inclusión $\langle L_1 \rangle \subset \langle L_2 \rangle$. Sin embargo puede ocurrir que $\langle L_1 \rangle = \langle L_2 \rangle$ siendo $L_1 \neq L_2$.

- Sea L_1 el lenguaje formado por las expresiones cuya longitud es un múltiplo de 20 y L_2 el lenguaje de las expresiones de longitud múltiplo de 35. Ambos lenguajes son cerrados por concatenación. Su intersección está formada por las expresiones cuya longitud es múltiplo, a la vez, de 20 y de 35. Esto es, deben ser múltiplos del mínimo común múltiplo de ambos números. Sin embargo la unión de ambos lenguajes no es cerrada por concatenación. Por ejemplo, si $\mathsf{long}[\alpha] = 20$ y $\mathsf{long}[\beta] = 35$, su concatenación tiene longitud 55 y no pertenece a la unión.

- Si A es un alfabeto, entonces $\langle A \rangle = A^*$ pues todo elemento de A^* se construye a base de concatenaciones de elementos de A. Si tenemos que $A \subset B$ entonces $\langle B \rangle = A^*$.

- Sea $B \subset A$. Entonces los elementos de $\langle B \rangle$ se forman concatenando elementos de B. Este conjunto se identifica con el conjunto de todas las expresiones de B. En otras palabras, $\langle B \rangle = B^*$.

- Sea $A = \{0, 1\}$ y sea $B = A^8$. Entonces el conjunto generado por B está formado por las palabras cuya longitud es un múltiplo de 8. Es el lenguaje de la informática que hemos construido anteriormente.

- Sea $f : A^* \times A^* \to A^*$ una operación interna del conjunto de expresiones. En principio f es arbitraria y no tiene que cumplir ninguna propiedad especial. Decimos que un conjunto $L \subset A^*$ es cerrado bajo f, si para todo $\alpha, \beta \in L$ se tiene que $f[\alpha, \beta] \in L$. La intersección de conjuntos cerrados bajo f es de nuevo cerrado bajo f.

- Sea $\{f_i\}_{i \in I}$ una colección, posiblemente infinita, de operaciones en A^*. Decimos que un conjunto L es cerrado bajo el conjunto de operaciones $\{f_i\}$ si para todo $\alpha, \beta \in L$ y para todo $i \in I$, se tiene que $f_i[\alpha, \beta] \in L$. De nuevo la intersección de conjuntos cerrados bajo estas operaciones es cerrado.

- Llamamos `operación de rango` n a una función

$$f : A^* \times \overset{n)}{\ldots} \times A^* \mapsto A^*$$

Se puede definir del mismo modo la noción de conjunto cerrado bajo f y se cumple la misma propiedad para la intersección. Si consideramos un conjunto de operaciones, de rangos variables, se tiene el mismo resultado.

2. Lenguaje de la lógica proposicional

Para construir la lógica proposicional necesitamos primeramente construir un alfabeto. En muchos libros se opta por definir un alfabeto infinito, aunque nosotros seguiremos otro camino, utilizando un alfabeto finito. Es fácil demostrar que ambos caminos llevan en realidad al mismo lugar y la elección de uno u otro es más que nada estética.

Definición 2.1 *El alfabeto de la lógica proposicional es el conjunto*

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

Los símbolos $\neg, \vee, \wedge, \rightarrow$ se llaman `conectivos`. Como muchos de los razonamientos que haremos serán similares para los tres conectivos $\vee, \wedge, \rightarrow$, emplearemos el símbolo $\star$ para designar a cualquiera de ellos. Para mejorar la notación, a la expresión que consiste en el símbolo p seguido de n barras $|$, lo denotaremos por p_n

$$p_0 = p, \quad p_1 = p\,|, \quad p_2 = p\,|\,|, \ldots$$

Las expresiones de la forma p_n las llamaremos `variables proposicionales` o simplemente `variables`. El subconjunto de todas las variables proposicionales lo denotaremos $\mathcal{V}$. Por construcción es claro que el conjunto $\mathcal{V}$ es infinito numerable.

En el conjunto de expresiones A^* de este lenguaje se definen cuatro operaciones internas, asociada cada una a un conectivo. Una de ellas, la asociada con $\neg$, es de rango 1, y las otras son todas de rango 2 (o binarias). Sus definiciones precisas son:

$$\begin{array}{lcl} E_\neg : \alpha & \longmapsto & \neg(\alpha) \\ E_\vee : (\alpha, \beta) & \longmapsto & (\alpha \vee \beta) \\ E_\wedge : (\alpha, \beta) & \longmapsto & (\alpha \wedge \beta) \\ E_\rightarrow : (\alpha, \beta) & \longmapsto & (\alpha \rightarrow \beta) \end{array}$$

donde debemos entender, por ejemplo, que $(\alpha \wedge \beta)$ es la concatenación de un paréntesis, junto con la expresión α, junto con el conectivo $\wedge$, etc.

De la misma forma que en la sección anterior estabamos interesados en los lenguajes cerrados por concatenación, en este tema son de vital importancia, los conjuntos cerrados por cada una de las cuatro operaciones anteriores.

Definición 2.2 *Decimos que un conjunto $I \subset A^*$ es* `inductivo` *si:*

- *Contiene a la variables: $\mathcal{V} \subset I$.*
- *Es cerrado por cada una de las cuatro operaciones que hemos construido anteriormente: si $\alpha, \beta \in I$, entonces $\neg(\alpha)$, $(\alpha \vee \beta)$, $(\alpha \wedge \beta)$, $(\alpha \to \beta)$ también pertenecen a I.*

Por construcción el conjunto total A^* es inductivo. Ya podemos definir el lenguaje, que denotaremos por **Form**. Sus elementos se llaman `fórmulas` o también `fórmulas bien formadas`, que muchas veces abreviaremos como fbf (en inglés es corriente encontrar la abreviatura wff derivada de *well-formed formula*).

Definición 2.3 *El lenguaje de la lógica proposional,* **Form***, es la intersección de todos los conjuntos inductivos.*

Proposición 2.1 *El lenguaje* **Form** *es inductivo. Además es el menor de los conjuntos inductivos, en el sentido de que si I es inductivo, entonces* **Form** $\subset I$.

Demostración.

Supongamos que α y β son elementos de **Form**. Entonces pertenecen a todos los conjuntos inductivos. De esta forma tanto $\neg(\alpha)$ como $(\alpha \star \beta)$ pertenecen a todos los conjuntos inductivos, y por lo tanto pertenecen a su intersección, que es justamente el lenguaje **Form**. Por construcción las variables están contenidas en el lenguaje, lo que prueba que en efecto es inductivo.

La otra parte se deriva directamente de la definición. □

Observación. El último de los resultados de la proposición anterior se suele denominar `principio de inducción para fbf`. Para demostrar que toda fbf posee una propiedad P se pueden hacer las siguientes comprobaciones:

- Las variables tienen la propiedad P.
- Si α y β tienen la propiedad P, entonces $\neg(\alpha)$, $(\alpha \vee \beta)$, $(\alpha \wedge \beta)$ y $(\alpha \to \beta)$ tienen la propiedad P.

Decimos entonces que hemos demostrado la propiedad P por inducción sobre las fbf. $\square$

En la sección anterior definimos en general el concepto de longitud de una expresión. Aplicando dicha definición tenemos que $\mathsf{long}[p_n] = n+1$ pues recordemos que p_n consiste en el símbolo p seguido de n barras. También se cumple que $\mathsf{long}[(\alpha \star \beta)] = 3 + \mathsf{long}[\alpha] + \mathsf{long}[\beta]$ pues estamos contando los paréntesis. Queremos una nueva noción de longitud que asigne la unidad a todas las variables y que «no cuente» los paréntesis.

Definición 2.4 *Dada* $\alpha \in$ **Form** *llamamos* `longitud modificada` *de* α *y denotamos* $\mathsf{long}^*[\alpha]$ *a la suma de las variables y de los conectivos que aparecen en* α*, contados tantas veces como aparezcan.*

Muchas veces cometemos el abuso de notación de llamar símplemente longitud a lo que en realidad es la longitud modificada. El contexto debe aclarar este abuso de notación.

Ejemplos.

- Si $\alpha = (((p_0 \wedge p_1) \to p_4) \wedge \neg(p_4))$ tenemos que su longitud modificada es 8, pues aparecen cuatro variables (algunas repetidas) y cuatro conectivos, algunos también repetidos.
- La longitud modificada cumple las propiedades

$$\begin{aligned}
&\mathsf{long}^*[p_n] = 1\\
&\mathsf{long}^*[\neg(\alpha)] = 1 + \mathsf{long}^*[\alpha]\\
&\mathsf{long}^*[(\alpha * \beta)] = 1 + \mathsf{long}^*[\alpha] + \mathsf{long}^*[\beta]
\end{aligned}$$

No es difícil probar que es la única función $f : \mathbf{Form} \mapsto \mathbb{N}$ que cumple las tres propiedades anteriores.

- Muchas veces cometeremos el abuso de notación de no escribir los paréntesis que se sobreentiendan. Por ejemplo, es bastante habitual suprimir los paréntesis externos y escribir $\alpha \star \beta$ en lugar de $(\alpha \star \beta)$. La longitud modificada no se ve afectada por este convenio, puesto que «no cuenta» los paréntesis.

La anterior construcción del lenguaje es rápida y elegante, pero no es constructiva. Veremos otras formas de definir este mismo lenguaje, pero que carezcan de este inconveniente. Para ello se suelen emplear procesos inductivos. Para ello construimos los conjuntos:

$$
\begin{aligned}
&Form_0 = \mathcal{V} \\
&Form_1 = Form_0 \cup \{\neg(\alpha), \alpha \in Form_0\} \cup \{(\alpha \star \beta), \alpha, \beta \in Form_0\} \\
&\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \\
&Form_{i+1} = Form_i \cup \{\neg(\alpha), \alpha \in Form_i\} \cup \{(\alpha \star \beta), \alpha, \beta \in Form_i\}
\end{aligned}
$$

Con ayuda de estos conjuntos definimos el conjunto (ojo a la falta de negrita):

$$Form = \bigcup_{n=0}^{\infty} Form_i$$

Proposición 2.2 *Se tiene que* $\mathbf{Form} = Form$.

Demostración.

Como **Form** es cerrado por las operaciones asociadas a los conectivos, probemos, inductivamente, que $Form_i \subset \mathbf{Form}$. Como $Form_0$ consta únicamente de las variables tenemos la inclusión. Supongamos que $Form_i$ está contenido para todo $i < n$. El conjunto $Form_{i+1}$ está compuesto por los propios elementos de $Form_i$, más los elementos obtenidos aplicando las operaciones E a dichos elementos. Como **Form** es cerrado por las operaciones, concluimos que también $Form_{i+1}$ está contenido. Por lo tanto la unión también está contenida. Esto prueba que $Form \subset \mathbf{Form}$.

Para demostrar el recíproco, probaremos que $Form$ es inductivo. Sean α y β elementos de $Form$. Como $Form_j \subset Form_i$ si $j < i$, debe existir un índice i tal que $\alpha, \beta \in Form_i$. Entonces $(\alpha \star \beta) \in Form_{i+1} \subset Form$. El resto de los detalles son elementales. $\square$

Ahora nos hacemos una pregunta básica. ¿Si nos encontramos con cualquier expresión formada con el alfabeto A, como sabemos si es una fbf? Para responder con comodidad a la pregunta debemos introducir algún material nuevo.

Definición 2.5 *Una* `cadena de formación` *de longitud n es una sucesión $X_1, X_2, \ldots, X_n$ de elementos de A^* que satisface las siguientes condiciones:*

- *X_i es una variable, o bien*
- *$X_i = \neg(X_j)$ con $j < i$, o bien*
- *$X_i = (X_j \star X_k)$ donde j y k son estrictamente menores que i.*

Podemos entender que una cadena de formación parte de variables proposicionales y aplica un número finito de veces las operaciones E o bien a dichas variables, o bien a los resultados de dichas variables. Como el lenguaje **Form** es cerrado por dichas operaciones, tenemos que todos los miembros de una serie de formación pertenecen al lenguaje. En realidad veremos que toda fbf se puede obtener de este modo. Por ello muchas veces se dice que

$$E = \{E_\neg, E_\vee, E_\wedge, E_\rightarrow\}$$

son las `reglas de formación` y que el conjunto de variables es el `generador` del lenguaje.

Definición 2.6 *Decimos que $\alpha \in A^*$* `admite una serie de formación`*, si existe una serie de formación $X_1, \ldots, X_n$ con $\alpha = X_n$.*

Proposición 2.3 *Un elemento $\alpha \in A^*$ admite una serie de formación si y solo si $\alpha \in$* **Form**.

Demostración.

Vamos a ver primeramente que toda expresión que admita una serie de formación es una fbf. Sea $X_1, \ldots, X_n = \alpha$ una serie de formación. Entonces X_1 es necesariamente una variable y por tanto es una fbf. En general X_i es o bien una variable o bien se obtiene de las anteriores expresiones de la cadena, aplicando alguna de las operaciones E. Como **Form** es cerrado por todas estas operaciones, inductivamente se prueba que X_n es una fbf.

Denotemos por $Form$ al conjunto de expresiones que admiten una serie de formación. Demostraremos que este conjunto es inductivo. Claramente las variables pertenecen al conjunto, pues admiten series de formación de longitud unidad. Supongamos que $X_1, \ldots, X_n$ e $Y_1, \ldots, Y_m$ son series de formación. Entonces

$$X_1, \ldots, X_n, Y_1, \ldots, Y_m, (X_n \star Y_m)$$

es una serie de formación (compruebese) de $(X_n \star Y_m)$. El conjunto es cerrado por tres de las cuatro operaciones. El caso del conectivo $\neg$ es incluso más sencillo. $\square$

Si queremos probar que una expresión es una fórmula bien formada, simplemente tenemos que exhibir una serie de formación.

Ejemplos.

- Dada la expresión $(((p_0 \wedge p_1) \rightarrow p_4) \vee \neg(p_4))$, podemos comprobar que la siguiente es una serie de formación:

$$\begin{aligned} X_1 &= p_0 \\ X_2 &= p_1 \\ X_3 &= (X_1 \wedge X_2) \\ X_4 &= p_4 \\ X_5 &= (X_3 \rightarrow X_4) \\ X_6 &= \neg(X_4) \\ X_7 &= (X_5 \vee X_6) \end{aligned}$$

Esta misma expresión puede admitir otras series de formación distintas, pero es imposible encontrar una que tenga menos de siete «eslabones». La razón es la siguiente: como en la expresión aparecen tres variables, deben existir tres eslabones asociados a dichas variables. Como tenemos cuatro conectivos, debemos emplear al menos otros cuatro eslabones, uno para cada aparición del conectivo.

- La expresión $((p_1 \vee \rightarrow (p_2)$ no es una fbf. Hay muchas razones que implican que esta expresión no puede ser una fbf. Una razón es que nunca pueden existir dos conectivos binarios seguidos, debido al método de construcción de las series de formación. Otra posible razón es que en la expresión hay más paréntesis izquierdos que derechos. En el siguiente punto probamos que esto es imposible.

- Dada una expresión $X \in A^*$ llamamos `peso` de X, y denotamos $\text{peso}[X]$, al número que se obtiene restanto el número de paréntesis izquierdos «(» menos el número de paréntesis derechos «)» que posee X. Las variables tienen peso cero, puesto que carecen de paréntesis. Aplicando la definición observamos que

$$\text{peso}[\neg(X)] = \text{peso}[X] \qquad \text{peso}[(X \star Y)] = \text{peso}[X] + \text{peso}[Y]$$

  El conjunto de los elementos de peso nulo es inductivo. Toda fbf tiene peso nulo.

- Denotemos por C_n al conjunto de fbf que admiten una serie de formación de longitud n. Entonces $C_1 = \mathcal{V}$ y $C_i \subset C_j$ si $i < j$. Además la proposición anterior afirma que

$$\mathbf{Form} = \bigcup_{n=1}^{\infty} C_i$$

Como toda fbf es el último eslabón de una serie de formación, cada fbf γ debe ser de alguno de los siguientes cinco tipos

1) $\gamma = p_n$

2) $\gamma = \neg(\alpha)$ con $\alpha \in \mathbf{Form}$

3) $\gamma = (\alpha \vee \beta)$ con $\alpha, \beta \in \mathbf{Form}$

4) $\gamma = (\alpha \wedge \beta)$ con $\alpha, \beta \in \mathbf{Form}$

5) $\gamma = (\alpha \to \beta)$ con $\alpha, \beta \in \mathbf{Form}$

Esta claro que una fbf del primer tipo nunca puede ser de ningún otro tipo, pues los tipos 2)-5) no comienzan por una variable. También es claro que una fórmula de tipo 2) no puede ser de otro tipo, pues las de tipo 2) son las únicas que comienzan con el símbolo $\neg$. Lo que no está tan claro es que una del tipo 3) no pueda ser también del tipo 4) o 5).

Recordemos que una expresión α es prefijo de otra expresión β si $\beta = \alpha\gamma$.

Lema 2.4 *Una fbf no puede ser prefijo de otra fbf.*

Demostración.

Haremos la demostración por inducción sobre la longitud modificada. Si $\mathsf{long}^*[\alpha] = 1$ necesariamente $\alpha = p_n$. Pero una constante nunca es prefijo de una fbf de mayor longitud, puesto que todas comienzan, o con el símbolo $\neg$ o con un paréntesis.

Por hipótesis de inducción suponemos que el enunciado es cierto para toda fbf de longitud (modificada) menor o igual que n. Si una fbf tiene longitud $n+1$ entonces pueden darse cuatro casos:

a) La fbf es de la forma $\neg(\alpha)$. Si $\neg(\alpha)$ es prefijo de una fórmula, necesariamente esta fórmula es de la forma $\neg(\beta)$. Por lo tanto α es prefijo de β, lo cual es imposible, pues α cumple la hipótesis de inducción.

b) La fbf es de la forma $(\alpha \vee \beta)$. Si $(\alpha \vee \beta)$ es prefijo de una fórmula, esta debe ser del tipo $(\gamma \star \delta)$. Dependiendo de la posición relativa de $\vee$ y $\star$ tenemos tres posibilidades:

   1.- Si $\vee$ está a la izquierda de $\star$, entonces α es prefijo de γ. Imposible por hipótesis de inducción, puesto que $\mathsf{long}^*[\alpha] < n+1$.

2.- Si $\vee$ está a la derecha de $\star$, entonces γ es prefijo de α. De nuevo esto es imposible.

3.- Si $\vee$ y $\star$ están en la misma posición entonces β es prefijo de δ. De nuevo esto implicaría una contradicción.

Las demostraciones para los otros conectivos son similares. $\square$

Teorema 2.5 *Cada una de las fbf es solamente de un tipo de los anteriores.*

Demostración.

Todos los casos problemáticos se reducen a demostrar la siguiente cuestión

Si $(\alpha \star \beta) = (\alpha' \circ \beta')$ *entonces* $\alpha = \alpha'$, $\star = \circ$ *y* $\beta = \beta'$.

Lo demostraremos por reducción al absurdo. Supongamos que en efecto es cierto que $(\alpha \star \beta) = (\alpha' \circ \beta')$ y que $\alpha \neq \alpha'$. Si $\star$ está a la izquierda de $\circ$, α es prefijo de α', lo cual contradice el lema anterior. Los otros casos se analizan de la misma forma. Hemos visto que necesariamente $\alpha = \alpha'$. Ahora es trivial demostrar que $\star = \circ$ y que $\beta = \beta'$. $\square$

Si denotamos por $A_\vee$ al conjunto de fbf del tipo $(\alpha \vee \beta)$, y definiciones análogas para los otros conectivos, el teorema anterior afirma que

$$\mathbf{Form} = \mathcal{V} \cup A_\neg \cup A_\vee \cup A_\wedge \cup A_\rightarrow$$

siendo los conjuntos disjuntos dos a dos.

El resultado anterior se puede enunciar de un modo más algebraico.

Corolario 2.6 *Consideremos las cuatro aplicaciones E. Entonces:*

- *Cada una de las aplicaciones E es inyectiva.*
- *Los conjuntos imagen de $E_\star$ y $E_\circ$ son disjuntos si $\star \neq \circ$.*

Demostración.

La inyectividad es consecuencia de la unicidad de la descomposición. Por ejemplo si $E_\vee[(\alpha,\beta)] = E_\vee[(\alpha',\beta')]$, tenemos que $(\alpha\vee\beta) = (\alpha'\vee\beta')$, de donde deducimos las igualdades $\alpha = \alpha'$ y $\beta = \beta'$, lo que prueba la inyectividad. De la unicidad del conectivo se deduce que los conjuntos son disjuntos. □

Definición 2.7 *Dado $\alpha \in$* **Form**, *llamamos* `grado` *de α al número de conectivos que aparecen en α, contados tantas veces como aparezcan.*

Las fbf de grado cero son las constantes. Cualquier otra fbf que no sea constante es de uno de los cuatro tipos anteriormente vistos. Si $\alpha = \neg(\beta)$ el grado de α es una unidad mayor que el grado de β. Si $\alpha = (\beta_1 \star \beta_2)$ el grado de α es superior a los grados de β_1 y β_2. Con más precisión se cumple

$$\mathsf{grado}[\neg(\alpha)] = 1 + \mathsf{grado}[\alpha] \qquad \mathsf{grado}[(\alpha \star \beta)] = 1 + \mathsf{grado}[\alpha] + \mathsf{grado}[\beta]$$

Estas fórmulas, unidas a la unicidad del tipo de descomposición, nos permitirán realizar razonamientos inductivos sobre el grado de las fbf. Un ejemplo de ello es la siguiente

Definición 2.8 *El concepto de* `subfórmula` *se define inductivamente por las propiedades:*

- *Si* $\mathsf{grado}[\alpha] = 0$ *la única subfórmula es ella misma.*
- *Si $\alpha = \neg(\beta)$, las subfórmulas de α son la propia α y las subfórmulas de β.*
- *Si $\alpha = (\beta_1 \star \beta_2)$ las subfórmulas de α son la propia α, más todas las subfórmulas de β_1 más todas las subfórmulas de β_2.*

Es bastante habitual, al descomponer una fbf en subfórmulas, utilizar un árbol. Sin embargo también puede realizarse el mismo trabajo sin emplear dichos árboles.

Ejemplos.

- Calculemos todas las subfórmulas de

$$(((p_0 \wedge p_1) \to p_4) \vee \neg(p_4))$$

Esta fbf es de tipo $\vee$. Por lo tanto tenemos dos subfórmulas

$$((p_0 \wedge p_1) \to p_4) \qquad \neg(p_4)$$

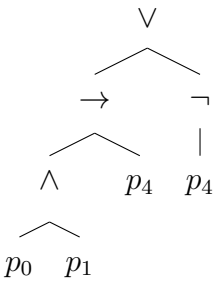
La primera de estas fbf es de tipo $\to$. Obtenemos entonces

$$(p_0 \wedge p_1) \qquad p_4$$

Continuando de este modo, obtenemos que el conjunto de subfórmulas es

$$\{((p_0 \wedge p_1) \to p_4), \neg(p_4), (p_0 \wedge p_1), p_4, p_0, p_1\}$$

El árbol asociado a esta descomposición es



- Todo árbol asociado a una descomposición en subfórmulas tiene variables en los vértices terminales y conectivos en los vértices interiores. Conociendo el árbol se pueden reconstruir todas las subfórmulas y la fórmula de partida.

- Se puede definir de un modo más simple la noción de `subfórmula`. Una subfórmula de α es una subexpresión de α que sea a la vez una fbf. Sin embargo hemos optado por definirla de modo inductivo, pues ello nos da un método efectivo para encontrar todas las subfórmulas. Con un poco de trabajo se ve que este método también sirve para crear series de formación de cualquier fbf.

- Hemos definido el concepto de subfórmula por inducción sobre el grado. Del mismo modo podríamos haber utilizado la longitud modificada para realizar la inducción. En general casi siempre se puede utilizar el grado o la longitud para realizar inducciones.

Sea α una fbf y supongamos que $\neg$ forma parte de la expresión de α. Al construir la serie de formación de α, en algún eslabón de la cadena «aparece» el conectivo. Dicho eslabón es de la forma $\neg(\beta)$. Aplicando los argumentos anteriores se prueba sin dificultad la unicidad de β. Del mismo modo, si $\star$ es un conectivo binario que forma parte de α, existen dos subfórmulas β_1 y β_2 tales que $(\beta_1 \star \beta_2)$ es también subfórmula. De nuevo se tiene la unicidad.

Definición 2.9 *Sea α una fbf. Si $\neg$ forma parte de α se llama* `alcance` *de $\neg$ a la única subfórmula β tal que $\neg(\beta)$ es también subfórmula. Para conectivos binarios se definen de modo análogo el* `alcance anterior` *y el* `alcance posterior`.

Observación. Aunque no daremos las definiciones precisas, se tiene el siguiente resultado: *dadas dos subfórmulas β_1 y β_2 de α o bien son disjuntas, o bien una está contenida en la otra.* □

Ahora que ya hemos terminado de explicar la construcción del lenguaje proposicional, digamos clásico, estamos en disposición de generalizar el concepto. En realidad para construir todo el proceso hemos consideramos un conjunto $\mathcal{V}$ de variables y hemos añadido al lenguaje los conectivos y los paréntesis. Una vez hecho esto hemos procedido a crear, utilizando series de formación, el concepto de fbf. En nuestro caso el conjunto $\mathcal{V}$ es un conjunto numerable, pero nada nos impide adoptar como conjunto de variables un nuevo conjunto que puede ser finito o innumerable. Podemos tomar como conjunto de variables cualquier conjunto B y repetir todos los pasos de la construcción del lenguaje. Todos los resultados se extrapolan a esta nueva situación.

3. Semántica del lenguaje proposicional

En el conjunto $B = \{0, 1\}$ introducimos una serie de cuatro operaciones internas. Una de ellas será de rango 1 y las otras tres serán binarias. Aunque se emplean símbolos similares a los conectivos lógicos, en principio no tienen nada que ver y perfectamente podríamos haber utilizado otros. La razón de emplear estos se verá al definir el concepto de valuación. Mediante x e y denotamos elementos de B. Las definiciones de las cuatro operaciones internas son:

$$\begin{aligned} \neg\, x &= 1 - x \\ x \vee y &= \text{máx}\,[x, y] \\ x \wedge y &= \text{mín}\,[x, y] \\ x \to y &= \text{máx}\,[1 - x, y] \end{aligned}$$

Como el conjunto B solamente posee dos elementos es fácil especificar las tablas de multiplicación de todas estas operaciones

x	y	$x \vee y$	$x \wedge y$	$x \to y$	$\neg\, x$
1	1	1	1	1	0
1	0	0	1	0	0
0	1	0	1	1	1
0	0	0	0	1	1

El conjunto B junto con estas operaciones es lo que en matemáticas se denomina un `álgebra de Boole`. Naturalmente todas estas operaciones tienen una serie de propiedades, como asociatividad, conmutatividad,... que no nos detenemos en explorar, puesto que no son útiles en este momento.

Definición 3.1 *Llamamos* `valuación` *a toda aplicación* $v : \mathbf{Form} \mapsto B$ *que cumple, para toda* $\alpha, \beta \in \mathbf{Form}$*, las siguientes propiedades:*

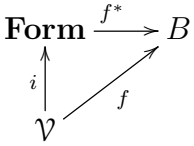
- $v[\neg\, \alpha] = \neg\, v[\alpha]$
- $v[\alpha \vee \beta] = v[\alpha] \vee v[\beta]$.
- $v[\alpha \wedge \beta] = v[\alpha] \wedge v[\beta]$.

- $v[\alpha \to \beta] = v[\alpha] \to v[\beta]$.

El conjunto de todas las valuaciones lo denotamos **Val**.

Una valuación «respeta» las operaciones y podemos entenderla como una especie de morfismo entre estructuras algebraicas. Como el conjunto **Form** está generado por las variable y las aplicaciones E, es hasta cierto punto natural que una valuación quede determinada por su valor sobre las variables. Ello se demuestra en el

Teorema 3.1 *Sea $f : \mathcal{V} \mapsto B$ una función arbitraria. Existe una única valuación $f^* : \mathbf{Form} \mapsto B$ que hace conmutativo el diagrama*



Demostración.

Estamos pidiendo que f^* sea una valuación y que sobre las variables coincida con f. Pasemos a construir, por inducción sobre el grado, f^*.

Si α es de grado cero, entonces es una variable y necesariamente debemos definir

$$f^*[\alpha] = f[\alpha]$$

Supongamos que hemos definido f^* para todas las fbf de grado menor que n. Si α es de grado n y es de la forma $\alpha = \neg(\beta)$, si queremos que sea valuación debemos definir

$$f^*[\alpha] = \neg\, f^*[\beta]$$

Si α es de otro tipo entonces $\alpha = \beta_1 \star \beta_2$. De nuevo, si queremos que sea valuación debemos definir

$$f^*[\alpha] = f^*[\beta_1] \star f^*[\beta_2]$$

lo que completa la definición inductiva de f^*.

Es fácil probar que la aplicación f^* así construida es en efecto una valuación y que necesariamente es única, pues cada uno de los pasos de la construcción nos vienen impuestos por la necesidad de que la extensión sea una valuación. □

Observación. Existe un concepto más general que el de valuación. Si X es un álgebra de Boole arbitraria, podemos considerar las funciones de **Form** en X que conservan los conectivos. Muchas demostraciones se extienden a este caso más general. □

Corolario 3.2 *El conjunto* **Val** *se identifica, de modo canónico, con el conjunto de funciones de $\mathcal{V}$ en B. También se identifica con el conjunto $\mathcal{P}(\mathcal{V})$, formado por los subconjuntos de $\mathcal{V}$.*

Demostración.

En el teorema hemos probado la primera identificación. Para establecer la segunda basta observar que cada subconjunto de $\mathcal{V}$ queda identificado por su función característica, que es una función de $\mathcal{V}$ en $\{0, 1\}$. □

Como el conjunto de las variables es numerable, el conjunto de sus partes tiene el cardinal del continuo. El cardinal del conjunto de valuaciones coincide con el del continuo.

Corolario 3.3 *Dado cualquier subconjunto $\{x_1, \ldots, x_n\} \subset \mathcal{V}$ y cualquier sucesión $a_1, \ldots, a_n$ de elementos de B, existe una valuación v que cumple $v(x_i) = a_i$ para todo índice.*

Demostración.

Construimos la función $f(p_n) = a_k$ si $p_n = x_k$ y $f(p_n) = 0$ en cualquier otro caso. La extensión f^* cumple el enunciado. □

Observación. Por el método que hemos seguido para construir la valuación observamos que existen infinitas valuaciones que cumplen el enunciado. Basta con variar la definición de f sobre las variables que no coincidan con las x_i y

tendremos nuevas valuaciones que cumplen el corolario. Con un poco más de trabajo, se demuestra el corolario aun en el caso de que el conjunto no sea finito. □

Definición 3.2 *Decimos que* α *es una* `tautología` *si* $v(\alpha) = 1$ *para toda valuación. Decimos que es una* `contradicción` *si* $v(\alpha) = 0$ *para toda valuación. Decimos que es una* `contingencia` *si existen valuaciones* v_0 *y* v_1 *que cumplen* $v_0(\alpha) = 0$ *y* $v_1(\alpha) = 1$.

Cada una de las fbf es de alguno de los tipos y no puede ocurrir que una misma fbf sea de dos tipos distintos. Esta clasificación induce una descomposición de **Form** en tres subconjuntos disjuntos.

En principio puede parecer difícil probar que una fbf es una tautología, pues debemos demostrar que toda valuación, y hay un conjunto infinito de valuaciones, da la unidad sobre ella. Sin embargo veremos que basta hacer la comprobación para un conjunto finito de valuaciones.

Ejemplos.

- Tenemos que $\alpha = p_n \vee \neg(p_n)$ es siempre una tautología. En efecto, si una valuación cumple $v(p_n) = 0$, se tiene, aplicando que v «respeta» las operaciones, que $v[\alpha] = v[p_n \vee \neg(p_n)] = v[p_n] \vee \neg v[p_n] = 1 \vee 0 = 1$. Del mismo modo si $v[p_n] = 0$ también se cumple.

- Por el mismo método anterior es fácil verificar que $p_n \wedge \neg(p_n)$ es siempre una contradicción.

- Sea $\alpha = ((p_0 \wedge p_1) \rightarrow p_2) \rightarrow ((p_0 \rightarrow p_2) \vee (p_1 \rightarrow p_2))$. Aplicando reiteradamente las propiedades que definen la valuación se tiene que

  $$v[\alpha] = ((v[p_0] \wedge v[p_1]) \rightarrow v[p_2]) \rightarrow ((v[p_0] \rightarrow v[p_2]) \vee (v[p_1] \rightarrow v[p_2]))$$

  En definitiva, el valor de $v[\alpha]$ depende solo y exclusivamente de los valores de sus variables. Para analizar todos los casos posibles, es común

emplear un método conocido como `tablas de verdad`. He aquí la tabla de verdad de esta fbf.

p_0	p_1	p_2	α
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	1
0	1	1	1
0	1	0	1
0	0	1	1
0	0	0	1

En este caso hemos demostrado que α es una tautología.

- α es una tautología si y solo si $\neg(\alpha)$ es una contradicción.

Hemos visto en un ejemplo anterior que el valor de una valuación sobre una fbf depende solo y exclusivamente del valor de la valuación sobre las variables que intervienen en la fbf.

Corolario 3.4 *Sea α una fbf tal que sus variables son un subconjunto de $\{x_1, \ldots, x_n\}$. Si dos valuaciones v y w cumplen que $v[x_i] = w[x_i]$ entonces $v[\alpha] = w[\alpha]$.*

Demostración.

Se puede demostrar por inducción sobre el grado. Simplemente se debe utilizar que las valuaciones respetan las operaciones. □

Por ello, realizando una tabla de verdad, que es un proceso finito, podemos saber si la fórmula es una contingencia, una tautología o una contradicción Además de este método de las tablas de verdad existen otros métodos, también finitos, que nos permiten averiguar el tipo de cada fórmula. No los analizaremos aquí.

4. Equivalencia y álgebras de Boole

A nivel sintáctico dos expresiones son iguales si tienen exactamente los mismos símbolos y colocados en los mismos lugares. Sin embargo, en los lenguajes naturales, dos frases o palabras pueden ser distintas y sin embargo expresar la misma idea. Dos frases que significan lo mismo, a efectos semánticos, son totalmente equivalentes. En la lógica proposicional puede ocurrir que dos fbf sean sintácticamente distintas, pero todas sus posibles interpretaciones sean iguales.

Definición 4.1 *Decimos que* $\alpha, \beta \in$ **Form** *son* `equivalentes` *(o mejor dicho son* `semánticamente equivalentes`*) si para toda valuación* $v \in$ **Val** *se cumple* $v[\alpha] = v[\beta]$*. Si dos fbf son equivalentes lo denotamos* $\alpha \equiv \beta$*.*

Es fácil comprobar que efectivamente la relación anterior es de equivalencia. Esto es, que cumple las propiedades reflexiva, simétrica y transitiva.

Ejemplos.

- Si α y β son tautologías entonces $\alpha \equiv \beta$. Análogamente si ambas son contradicciones. Sin embargo puede ocurrir que ambas sean contingencias y sin embargo no sean equivalentes.
- Las fbf p_0 y $\neg\neg p_0$ son equivalentes. En efecto, si $v[p_0] = 1$ entonces también $v[\neg\neg p_0] = 1$ y lo mismo en el otro caso. En general si α es cualquier fbf entonces $\alpha \equiv \neg\neg\alpha$.
- Las fbf $(p_0 \vee p_1)$ y $(p_1 \vee p_0)$ son equivalentes. Basta comprobar todos los casos, o lo que es lo mismo, realizar sus tablas de verdad.
- Existe otro método, distinto a las tablas de verdad, para probar el resultado anterior. Calcular $v[p_0 \vee p_1]$ es equivalente a calcular $v[p_0] \vee v[p_1]$. Esta última operación se realiza en B y aquí la operación $\vee$ es conmutativa. Por ello si calculamos $v[p_1 \vee p_0]$ obtenemos el mismo resultado.

Como hemos visto en los ejemplos anteriores, para comprobar que dos fbf son equivalentes basta realizar sus tablas de verdad.

Proposición 4.1 *Sean α y β dos fbf con las mismas variables. Entonces son equivalentes si y solo si sus tablas de verdad coinciden.*

Demostración.

Si las fbf tiene n variables $\{x_1, \ldots, x_n\}$, la fila de la tabla de verdad que tiene como primeros n números la sucesión $(a_1, \ldots, a_n)$, le corresponde en la posición $n+1$ el número a_{n+1} construido de la siguiente forma: se toma una valuación v tal que $v[x_i] = a_i$ y entonces $a_{n+1} = v[\alpha]$. Si α y β son equivalentes, sus tablas de verdad son iguales.

Recíprocamente, si v es una valuación arbitraria denotamos por $a_i = v[x_i]$. La sucesión así construida pertenece a las primeras n posiciones de la tabla de verdad. Entonces $v[\alpha]$ es el valor a_{n+1} de dicha fila. Como la tabla de verdad de β es igual, tenemos que $v[\beta] = v[\alpha]$. Como esta construcción es válida para toda valuación, concluimos que $\alpha \equiv \beta$. $\square$

Aunque para nosotros no presenta un interés especial puede ser conveniente familiarizarse con algunas equivalencias. Para comprobarlas, basta realizar la tabla de verdad. También se puede utilizar que las valuaciones respetan los conectivos y utilizar algunas propiedades de las álgebras de Boole.

- Leyes conmutativas para $\wedge$ y $\vee$:

$$\alpha \wedge \beta \equiv \beta \wedge \alpha \qquad\qquad \alpha \vee \beta \equiv \beta \vee \alpha$$

- Leyes asociativas para $\wedge$ y $\vee$:

$$(\alpha \wedge \beta) \wedge \gamma \equiv \alpha \wedge (\beta \wedge \gamma) \qquad\qquad (\alpha \vee \beta) \vee \gamma \equiv \alpha \vee (\beta \vee \gamma)$$

- Negación:

$$\neg\neg\alpha \equiv \alpha$$

- Leyes de De Morgan:

$$\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta \qquad\qquad \neg(\alpha \wedge \beta) \equiv \neg\alpha \vee \neg\beta$$

- Relaciones entre los conectivos:

$$\alpha \wedge \beta \equiv \neg(\alpha \to \neg\beta) \qquad\qquad \alpha \vee \beta \equiv \neg\alpha \to \beta$$

Proposición 4.2 *Los fórmulas α y β son equivalentes si y solo si $\alpha \to \beta$ y $\beta \to \alpha$ son tautologías.*

Demostración.

Si $\alpha \equiv \beta$, supongamos que $v[\alpha] = 0$. Entonces $v[\beta] = 0$ y se deduce que $v[\alpha \to \beta] = 1$. Análogamente si $v[\alpha] = 1$ deducimos que $v[\alpha \to \beta] = 1$ y es una tautología.

Si $\alpha \to \beta$ y $\beta \to \alpha$ son tautologías, α y β deben ser equivalentes, puesto que si $v[\alpha] \neq v[\beta]$ tenemos una contradicción. Por ejemplo, si $v[\alpha] = 1$ y $v[\beta] = 0$, entonces $v[\beta \to \alpha] = 0$ y no sería una tautología. □

Observación. En muchas referencias se introduce un nuevo conectivo $\leftrightarrow$ definiendo

$$(\alpha \leftrightarrow \beta) = (\alpha \to \beta) \wedge (\beta \to \alpha)$$

Entonces $\alpha \equiv \beta$ si y solo si $\alpha \leftrightarrow \beta$ es una tautología. Este resultado se utiliza a menudo como definición de equivalencia. □

Como resumen de todo lo tratado se tiene el siguiente resultado.

Teorema 4.3 *El conjunto cociente* $\mathbf{Form}/\equiv$ *junto con las operaciones heredadas tiene estructura de álgebra de Boole.*

En particular, como las operaciones $\vee$ y $\wedge$ tienen la propiedad asociativa, muchas veces suprimiremos los paréntesis, como se hace habitualmente con las operaciones asociativas.

Observación. En nuestro álgebra de Boole se verifica

$$\alpha \wedge \beta \equiv \neg(\alpha \to \neg\beta) \qquad\qquad \alpha \vee \beta \equiv \neg\alpha \to \beta$$

Esta equivalencias (igualdades a nivel en el conjunto cociente) nos informan, que en álgebra de Boole, o lo que es lo mismo, a nivel semántico, basta únicamente con utilizar los conectivos $\neg$ y $\to$ para obtener todas las fórmulas. Por ello se dice que el par $\{\neg, \to\}$ es un sistema completo de conectivas. Existen otros sistemas completos de conectivas como por ejemplo $\{\neg, \wedge\}$ y $\{\neg, \vee\}$.

En definitiva, a nivel semántico, nos hubiese bastado con utilizar únicamente como conectivos $\neg$ y $\to$. Si hemos empleado los demás es simplemente por comodidad, para poder expresar de un modo más sencillo algunos resultados. □

Supongamos que $Var[\alpha] = \{x_1, x_2, \ldots, x_n\}$. Entonces la fbf α induce una función, que denotamos $f_\alpha : B^n \mapsto B$. Para definir como actua f_α sobre un elemento $(a_1, a_2, \ldots, a_n) \in B^n$, tomamos una valuación v que cumpla $v[x_i] = a_i$ para todo índice i. Entonces definimos

$$f_\alpha[(a_1, a_2, \ldots, a_n)] = v[\alpha]$$

Hemos visto anteriormente que siempre existe una valuación v que cumple el enunciado y que si tomamos otra valuación v' que lo cumple se tiene que $v[\alpha] = v'[\alpha]$. Por lo tanto la función está bien definida.

Corolario 4.4 *Si $\alpha \equiv \beta$ entonces $f_\alpha = f_\beta$.*

Corolario 4.5 *Si $Var[\alpha] = Var[\beta]$ y $f_\alpha = f_\beta$ entonces $\alpha \equiv \beta$.*

Demostración.

Si las variables son iguales, y las funciones inducidas también, entonces ambas fbf tienen la misma tabla de verdad y son equivalentes. □

Hemos visto que toda clase de equivalencia de fórmulas induce una función de B^n en B. El recíproco también es cierto: fijadas un conjunto de n variables

y una función de B^n en B, existe una fbf que la induce. No vamos a realizar una demostración abstracta de este resultado, sino que daremos un ejemplo (que se puede generalizar para obtener la demostración). Para ello, lo primero que debemos tener en cuenta, es que fijadas las variables, una función de B^n en B es lo mismo que una tabla de verdad. Tomemos la siguiente tabla de verdad

p_3	p_5	p_7	f
1	1	1	1
1	1	0	0
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	1
0	0	0	1

Nos fijamos en la primera línea que produce un 1 en la tabla de verdad. Construimos la fbf $\beta_1 = p_3 \wedge p_5 \wedge p_7$ (recordar la asociatividad). La siguiente fila que tiene un 1 es la tercera. Escribimos ahora $\beta_3 = p_3 \wedge \neg p_5 \wedge p_7$. Hemos colocado un signo $\neg$ delante de p_5 puesto que en esa fila la variable p_5 tiene un valor de 0. Del mismo modo construimos

$$\begin{aligned} \beta_5 &= \neg p_3 \wedge p_5 \wedge p_7 \\ \beta_7 &= \neg p_3 \wedge \neg p_5 \wedge p_7 \\ \beta_8 &= \neg p_3 \wedge \neg p_5 \wedge \neg p_7 \end{aligned}$$

Una vez que tenemos formadas estas sentencias las unimos mediante la disyunción

$$\beta = \beta_1 \vee \beta_3 \vee \beta_5 \vee \beta_7 \vee \beta_8$$

Es muy sencillo (e instructivo) comprobar que f_β es en efecto la función pedida.

Observación. Decimos que una fbf α es `normal disyuntiva`, si es de la forma

$$\alpha = \beta_1 \vee \beta_2 \vee \cdots \vee \beta_n$$

donde cada β_i es una conjunción de variables o negaciones de variables. El resultado anterior prueba que toda fbf es equivalente a una fórmula normal disyuntiva. También nos da un proceso para encontrar dicha fórmula normal disyuntiva. □

Corolario 4.6 *Si $\{x_1, \ldots, x_n\}$ es un conjunto de variables, el conjunto de clases de equivalencia que se pueden formar con dichas variables se identifica con el conjunto de funciones $f : B^n \mapsto B$.*

Como el conjunto de funciones de B^n en B tiene exactamente 2^{2^n} elementos, sabemos cuantas clases de equivalencia distintas podemos formar con n variables. En particular, con dos variables podemos formar 16 clases de equivalencia. Resulta que $p_0 \wedge p_1$, $p_0 \vee p_1$ y $p_0 \to p_1$ son tres clases de equivalencia distintas. Quedan otras 13. Con cada clase de esas 13 podríamos formar un nuevo conectivo binario, con una tabla de verdad distinta (uno de ellos es el conectivo $\leftrightarrow$ que ya hemos mencionado). En los estudios de lógica, digamos abstracta, no se trabaja con otros conectivos. Sin embargo en lógica aplicada si que se trabaja con otros conectivos. En particular en informática son muy utilizadoslos conectivos NAND y NOR, que se representan con $|$ y $\downarrow$ respectivamente. Estas conectivas tienen una propiedad especial: tanto el conjunto $\{|\}$ como el conjunto $\{\downarrow\}$ forman conjuntos completos de conectivas. Esto se utiliza en los diseños de circuitos, puesto que cualquier circuito lógico (que no es más que una función de B^n en B) se puede construir exclusivamente con puertas NAND o con puertas NOR.

5. Consecuencia lógica. Compacidad

En el lenguaje común estamos acostumbrados a sacar conclusiones a partir de ciertos conocimientos que denominamos premisas. Decimos que la conclusión B se obtiene de unas premisas $A_1, \ldots, A_n$, si la verdad de las premisas garantiza la verdad de la conclusión. Intentamos traducir estas ideas al formalismo de la lógica.

Definición 5.1 *Una fórmula α es* `satisfacible` *si existe una valuación v tal que $v[\alpha] = 1$. Un conjunto $\Sigma \subset$ **Form** es satisfacible si existe una valuación tal que $v[\alpha] = 1$ para toda $\alpha \in \Sigma$.*

Para indicar que $v[\alpha] = 1$ para toda $\alpha \in \Sigma$, cometeremos el abuso de notación de escribir $v[\Sigma] = 1$.

Ejemplos.

- Si α es una tautología o una contingencia, entonces es satisfacible. Si α es una contradicción entonces no es satisfacible.
- El conjunto $\Sigma = \{p_0, \neg(p_0)\}$ es no satisfacible.
- El conjunto $\Sigma = \{p_0 \wedge p_1, p_0 \vee p_1\}$ es satisfacible. Basta considerar una valuación v que cumpla $v[p_0] = 1$ y $v[p_1] = 1$.
- Si el conjunto Σ tiene un único elemento, basta realizar la tabla de verdad de dicho elemento para comprobar si es o no satisfacible: si aparece algún 1 en la tabla, entonces es satisfacible. En caso contrario no es satisfacible.

En el caso de los conjuntos finitos podemos trabajar únicamente con fórmulas y olvidarnos de los conjuntos, como muesta la siguiente

Proposición 5.1 *Sea $\Sigma = \{\alpha_1, \ldots, \alpha_n\}$ un conjunto finito. Entonces*

$$\Sigma \text{ es satisfacible } \Leftrightarrow \ \alpha_1 \wedge \cdots \wedge \alpha_n \text{ es satisfacible}$$

Demostración.

En efecto, si existe v que cumple $v[\alpha_1] = \cdots = v[\alpha_n] = 1$, entonces, aplicando las propiedades de las valuaciones, $v[\alpha_1 \wedge \cdots \wedge \alpha_n] = 1$.

Recíprocamente, si $v[\alpha_1 \wedge \cdots \wedge \alpha_n] = 1$, necesariamente $v[\alpha_i] = 1$, pues lo contrario implica una contradicción. □

Si el conjunto es infinito, el resultado no tiene sentido. Debemos esperar al teorema de compacidad para tratar del mismo modo el caso de conjuntos infinitos.

Observación. En el caso de los conjuntos finitos existe un procedimiento efectivo (lo que técnicamente se llama un algoritmo) que comprueba si el conjunto es o no satisfacible. Por ejemplo, las tablas de verdad es un algoritmo válido para esta comprobación. Aunque no entraremos en ello, esto significa que el problema de comprobar si un conjunto finito es o no satisfacible es `decidible`. □

Definición 5.2 *Dado* $\Sigma \subset \mathbf{Form}$ *decimos que* α *es una* `consecuencia` *de* Σ *si* $v[\Sigma] = 1$ *implica* $v[\alpha] = 1$. *Denotaremos este hecho por* $\Sigma \vDash \alpha$.

El conjunto de consecuencias de Σ se denota $\mathbf{Con}(\Sigma)$.

$$\mathbf{Con}(\Sigma) = \{\alpha \in \mathbf{Form} \text{ tal que } \Sigma \vDash \alpha\}$$

Ejemplos.

- Supongamos que $\Sigma = \varnothing$. Todas valuaciones que cumplen $v[\varnothing] = 1$. Por lo tanto las consecuencias del conjunto vacío son precisamente las fbf α que cumplen $v[\alpha] = 1$ para toda valuación. Esto es, las tautologías. Hemos probado que
$$\mathbf{Con}(\varnothing) = \mathbf{Tau}$$
- Si Σ es un conjunto no satisfacible, entonces no existe ninguna valuación que cumpla $v[\Sigma] = 1$. Por ello $\Sigma \vDash \alpha$ para cualquier α. De un conjunto,

o de una premisa, contradictoria se puede derivar cualquier conclusión. Si Σ no es satisfacible hemos demostrado que

$$\mathbf{Con}(\Sigma) = \mathbf{Form}$$

- Si $\Sigma = \{\alpha\}$ tiene un único elemento se suele cometer el abuso de notación de escribir $\alpha \vDash \beta$ en lugar de $\{\alpha\} \vDash \beta$.
- Si $\alpha \equiv \beta$ entonces $\alpha \vDash \beta$ y $\beta \vDash \alpha$. Es más, en algunos libros esta caracterización se toma como definición de equivalencia, pues el recíproco también es cierto.
- Sea $\Sigma = \{\alpha, \alpha \to \beta\}$. Entonces $\Sigma \vDash \beta$, pues suponer lo contrario implica una contradicción. En efecto, si existe v que cumple $v[\alpha] = 1$, $v[\alpha \to \beta] = 1$ y $v[\beta] = 0$, entonces tendríamos que $v[\alpha \to \beta] = v[\alpha] \to v[\beta] = 1 \to 0 = 0$, produciendose la contradicción.

Las nociones de consecuencia y satisfacibilidad están intimamente ligadas.

Teorema 5.2 *Dados* $\Sigma \subset \mathbf{Form}$ *y* $\alpha \in \mathbf{Form}$

$$\Sigma \vDash \alpha \Leftrightarrow \Sigma \cup \{\neg\, \alpha\} \textit{ no es satisfacible}$$

Demostración.

Supongamos que $\Sigma \vDash \alpha$. Entonces si $v[\Sigma] = 1$, necesariamente $v[\alpha] = 1$. Esto implica que no puede existir ninguna valuación que cumpla $v[\Sigma] = 1$ y $v[\neg(\alpha)] = 1$. El conjunto $\Sigma \cup \{\neg\, \alpha\}$ es insatisfacible.

Si $\Sigma \cup \{\neg\, \alpha\}$ es insatisfacible, toda v que cumple $v[\Sigma] = 1$, cumple también $v[\neg(\alpha)] = 0$. Esto es, si $v[\Sigma] = 1$ necesariamente $v[\alpha] = 1$. Pero esto es decir que $\Sigma \vDash \alpha$. $\square$

Corolario 5.3 *Sea* $\Sigma = \{\alpha_1, \ldots, \alpha_n\}$ *un conjunto finito. Entonces*

$$\Sigma \vDash \beta \Leftrightarrow \alpha_1 \wedge \ldots \alpha_n \wedge \neg\beta \textit{ es una contradicción}$$

Pero ya hemos visto que las contradicciones se pueden detectar con las tablas de verdad. En el caso de que el conjunto de premisas Σ sea finito podemos averiguar si $\alpha \in \mathbf{Con}(\Sigma)$ simplemente haciendo tablas de verdad. El conjunto $\mathbf{Con}(\Sigma)$, si Σ es finito, es decidible.

También están muy relacionados la noción de consecuencia lógica y el conectivo $\rightarrow$.

Proposición 5.4 *Tenemos que* $\beta \in \mathbf{Con}(\Sigma \cup \{\alpha\}) \Leftrightarrow \alpha \rightarrow \beta \in \mathbf{Con}(\Sigma)$. *En otras palabras*

$$\Sigma \cup \alpha \vDash \beta \text{ si y solo si } \Sigma \vDash \alpha \rightarrow \beta$$

Demostración.

Sea $\alpha \in \mathbf{Con}(\Sigma \cup \{\beta\})$. Sea v tal que $v[\Sigma] = 1$. Pueden darse dos casos:

- Si $v[\beta] = 1$ entonces $v[\Sigma \cup \{\beta\}] = 1$. Esto implica que $v[\alpha] = 1$. Por lo tanto $v[\alpha \rightarrow \beta] = 1$ y por lo tanto $\beta \rightarrow \alpha \in \mathbf{Con}(\Sigma)$.
- Si $v[\beta] = 0$, entonces $v[\beta \rightarrow \alpha] = 1$, independientemente del valor de $v[\alpha]$. También se concluye que $\beta \rightarrow \alpha \in \mathbf{Con}(\Sigma)$.

Recíprocamente. Sea $\beta \rightarrow \alpha \in \mathbf{Con}(\Sigma)$. Tomemos una valuación v tal que $v[\Sigma \cup \{\beta\}] = 1$. Entonces necesariamente $v[\alpha] = 1$, pues lo contrario implica una contradicción. Concluimos que $\alpha \in \mathbf{Con}(\Sigma \cup \{\beta\})$. $\square$

Comenzaremos a tratar los conjuntos infinitos de premisas.

Definición 5.3 *Un conjunto* Σ *es* `finitamente satisfacible` *si cualquier subconjunto finito de* Σ *es satisfacible.*

La definición afirma que si tomamos cualquier subconjunto finito Σ_1 existe una valuación v_1 que cumple $v_1[\Sigma_1] = 1$. Si tomamos otro subconjunto finito Σ_2 existe otra valuación v_2 que cumple lo mismo. En principio estas

valuaciones pueden ser distintas. Si la misma valuación sirviese para cualquier conjunto finito, entonces Σ sería satisfacible. El teorema de compacidad precisamente afirma esto: si para cualquier suconjunto finito Σ_1 existe una valuación v_1 que cumple $v_1[\Sigma_1] = 1$, entonces existe una valuación v que cumple $v[\Sigma] = 1$. Este resultado no es trivial y requiere alguna maquinaria extra para su demostración.

Definición 5.4 *Un conjunto* $\Delta \subset$ **Form** *es* `completo` *si para toda fbf* α *se tiene o bien* $\alpha \in \Delta$ *o bien* $\neg\,\alpha \in \Delta$ *(pero no ambas a la vez).*

Es claro que todo conjunto completo es necesariamente infinito. En principio no está nada claro ni que exista un conjunto completo. Vamos a dar un procedimiento que permite construir conjuntos completos.

Procederemos de modo inductivo. Como el conjunto de fbf es numerable podemos escribir

$$\mathbf{Form} = \{\alpha_1, \alpha_2, \dots\}$$

Construimos $\Delta_1 = \{\alpha_1\}$. Para construir Δ_n hacemos una comprobación: si $\neg\,\alpha_n = \alpha_i$ para algún $i < n$, entonces $\Delta_n = \Delta_{n-1}$. En caso contrario $\Delta_n = \Delta_{n-1} \cup \{\alpha_n\}$. El conjunto unión

$$\Delta = \bigcup_{n=1}^{\infty} \Delta_i$$

es completo, puesto que o bien $\alpha_n \in \Delta_n$ o bien $\neg\,\alpha_n \in \Delta_i$ con $i < n$. Existe otro método de construcción, basado en el lema de Zorn, que es más elegante.

Lema 5.5 *Si* Σ *es finitamente satisfacible entonces no puede ocurrir que* α *y* $\neg\,\alpha$ *pertenezcan a* Σ*.*

Demostración.

Si α y $\neg\,\alpha$ perteneciesen a Σ, el conjunto finito $\{\alpha, \neg\,\alpha\}$ no podría ser satisfacible. $\square$

Aunque este lema se puede demostrar de modo inductivo, realizaremos la demostración por el lema de Zorn.

Lema 5.6 *Si Σ es finitamente satisfacible, existe un conjunto completo Δ tal que $\Sigma \subset \Delta$.*

Demostración.

Construimos el conjunto de partes de **Form**.

$$C = \{H \subset \textbf{Form} \text{ tales que } \Sigma \subset H \text{ y } H \text{ es finitamente satisfacible}\}$$

Por el lema de Zorn existe un elemento maximal, que denotamos Δ. Este conjunto es finitamente satisfacible por construcción. Además es completo por el siguiente razonamiento:

- Si $\Delta \cup \{\alpha\}$ es finitamente satisfacible, dicho conjunto pertenece a C y por la maximalidad de Δ debe estar contenido en Δ. Esto implica que $\alpha \in \Delta$.
- Si $\Delta \cup \{\alpha\}$ no es finitamente satisfacible, entonces debe ocurrir que el conjunto $\Delta \cup \{\neg\, \alpha\}$ si sea finitamente satisfacible. El mismo razonamiento prueba en este caso que $\neg\, \alpha \in \Delta$.

Hemos probado que Δ es completo. $\square$

Teorema 5.7 (de compacidad) *Si Σ es finitamente satisfacible, entonces es satisfacible.*

Demostración.

Existe un conjunto Δ completo que contiene a Σ. Definimos una función $v : \textbf{Form} \mapsto B$ con el siguiente criterio

$$v[\alpha] = 1 \text{ si y solo si } \alpha \in \Delta$$

Por lo tanto necesariamente $v[\alpha] = 0$ si $\alpha \notin \Delta$. Pero como el conjunto es completo, se tiene que $v[\alpha] = 0$ si y solo si $\neg\, \alpha \in \Delta$.

Un análisis detallado, aunque sencillo, de todos los casos, demuestra que efectivamente v es una valuación. Por construcción se tiene que $v[\Sigma] = 1$ y el conjunto es satisfacible. $\square$

Corolario 5.8 *Si* $\Sigma \vDash \alpha$*, existe un conjunto finito* Σ_1 *que cumple* $\Sigma_1 \vDash \alpha$*.*

Demostración.

Sabemos que $\Sigma \vDash \alpha$ si y solo si el conjunto $\Sigma \cup \{\neg\alpha\}$ es insatisfacible. Imaginemos que para todo conjunto finito Σ_1 sea imposible que $\Sigma_1 \vDash \alpha$. Entonces el conjunto finito $\Sigma_1 \cup \{\neg\alpha\}$ si es satisfacible. Pero como esto ocurre para todo conjunto finito, por compacidad, tenemos que $\Sigma \cup \{\neg\alpha\}$ es satisfacible. Esto entra en contradicción con la hipótesis $\Sigma \vDash \alpha$. $\square$

6. Sistemas deductivos

En Matemáticas estamos acostumbrados a demostrar teoremas. Si tenemos una teoría, por ejemplo, la teoría de grupos, partimos de unos conocimientos básicos, que denominamos axiomas, y tras una serie de razonamientos, llegamos a una conclusión, que es lo que se denomina teorema. En los razonamientos debemos emplear la «lógica» en algún sentido que no podemos todavía precisar. Nosotros sabemos que si un conjunto cumple los axiomas (esto es, si es un grupo), entonces el resultado obtenido es cierto para dicho conjunto. Es más, para proceder a demostrar más teoremas, podemos apoyarnos en los ya demostrados y siempre obtenemos resultados verdaderos. Esto que hemos esbozado es el concepto intuitivo de deducción.

Definición 6.1 *Una* `regla de inferencia` *es una función parcial simétrica de rango n sobre el lenguaje* **Form**.

Aclaremos esta definición. Una función de rango n sobre **Form** es una aplicación

$$f: \quad \mathbf{Form} \times \overset{n)}{\cdots} \times \mathbf{Form} \quad \mapsto \quad \mathbf{Form}$$

Que sea simétrica significa que si cambiamos el orden de las coordenadas, el resultado no cambia. Finalmente, parcial significa que f puede no estar definida en todo el conjunto $\mathbf{Form}^n$ sino solamente sobre un subconjunto propio.

Propiedades.

- Como la función es simétrica el valor de $f[\alpha_1, \ldots, \alpha_n]$ dependen solamente del conjunto $\{\alpha_1, \ldots, \alpha_n\}$ y no del orden particular de los elementos.
- Existe una forma relativamente estandar para denotar las reglas de inferencia. Consiste en colocar el conjunto de argumentos sobre una línea y debajo de ésta la imagen

$$\frac{\alpha_1 \quad \alpha_2 \ldots \alpha_n}{f[\alpha_1, \alpha_2, \ldots, \alpha_n]}$$

A veces separamos los argumentos con comas para facilitar la lectura. También se puede escribir cada elemento α_i en una fila.

Definición 6.2 *Dada una regla de inferencia, si* $\beta = f[\alpha_1, \ldots, \alpha_n]$ *decimos que* β *es una* `consecuencia` *de* $\alpha_1, \ldots, \alpha_n$ *aplicando la regla* f*.*

Para manejar el concepto de regla de inferencia, debemos pensar que los argumentos de la regla de inferencia son las premisas de las que se parte y que su imagen es la conclusión que se obtiene de dichas premisas. De esta manera, debemos ser capaces de probar la verdad de la consecuencia a partir de la verdad de las premisas. Por ello únicamente nos interesarán las reglas de inferencia que cumpla la siguiente

Definición 6.3 *Una regla de inferencia* f *es* `correcta` *si para toda valuación* v *que cumple* $v[\alpha_1] = \cdots = v[\alpha_n] = 1$ *se tiene también que* $v[\beta] = 1$*, siendo* $\beta = f[\alpha_1, \ldots, \alpha_n]$*.*

Ejemplos.

- Tal vez la regla de inferencia más importante es el *modus ponens*:

$$\frac{\begin{array}{l}\alpha \\ \alpha \to \beta\end{array}}{\beta}$$

  Sea v una valuación tal que $v[\alpha] = 1$ y que $v[\alpha \to \beta] = 1$. Entonces necesariamente $v[\beta] = 1$, puesto que si fuese nulo tendríamos una contradicción. Esta regla es correcta.

- Otras reglas también muy conocidas (y también correctas) son:

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{\begin{array}{l}\alpha \\ \beta\end{array}}{\alpha \wedge \beta} \qquad \frac{\begin{array}{l}\neg\alpha \\ \alpha \vee \beta\end{array}}{\beta}$$

- Algunas reglas de inferencia no necesitan premisas. Ello equivale a que la función f sea de rango cero. Una de ellas es

$$\frac{\varnothing}{\alpha \vee \neg\alpha}$$

  Esta regla es correcta (pues la consecuencia es una tautología) y normalmente se elimina el conjunto vacío de su expresión, escribiendo simplemente

$$\frac{}{\alpha \vee \neg\alpha}$$

Es evidente que existe una gran variedad de reglas de inferencia. Sin embargo nosotros estaremos interesados en las reglas de inferencia que se utilizan habitualmente en el trabajo matemático, que naturalmente son correctas.

Definición 6.4 *Dado un conjunto R de reglas de inferencia, y un subconjunto $\Delta \subset$* **Form***, llamamos deducción desde Δ mediante R a cualquier sucesión finita $\alpha_1, \ldots, \alpha_n$ que cumpla alguna de las condiciones:*

- *α_i es un elemento de Δ.*
- *α_i es consecuencia de un conjunto de eslabones anteriores de la cadena mediante alguna regla de inferencia de R.*

Si existe una deducción con $\alpha_n = \beta$ decimos que β es `deducible` de Δ mediante R. Todos los eslabones de una deducción son deducibles.

Para empezar a realizar deducciones necesitamos un conjunto de reglas de inferencia y un conjunto de fbf de las que partir para realizar deducciones.

Definición 6.5 *Un* `sistema deductivo` *D es un par formado por un conjunto $\Lambda \subset$* **Form** *cuyos elementos se llaman* `axiomas` *y por un conjunto R de reglas de inferencia.*

El conjunto de axiomas de los que se parte puede ser vacío (existen reglas de inferencia que no necesitan premisas), pero el conjunto de reglas de inferencia nunca puede ser vacío.

Definición 6.6 *Fijado un sistema deductivo $D = (\Lambda, R)$, y un subconjunto $\Gamma \subset$* **Form***, llamamos* `teorema` *de Γ a cualquier β que se deduce de $\Lambda \cup \Gamma$ mediante las reglas R. Indicaremos esto mediante $\Gamma \vdash_D \beta$. Si el sistema deductivo se supone conocido se indica $\Gamma \vdash \beta$.*

Los elementos de Γ se dice que son las `premisas` del teorema. Si β se puede deducir sin premisas, esto es, si $\varnothing \vdash \alpha$, decimos que β es un `teorema lógico`. Se suele denotar por $\vdash \beta$.

Hemos visto que el conjunto de teoremas de Γ se construye de modo recursivo y por ello presenta un principio de inducción asociado. Supongamos fijado el sistema deductivo D. Denotamos por **Teorema**(Γ) al conjunto de todos los teoremas de Γ.

Proposición 6.1 *Dado un sistema deductivo $D = (\Lambda, R)$ se cumple:*

- $(\Lambda \cup \Gamma) \subset$ **Teorema**(Γ).
- *Si f es una regla de inferencia de orden n y $\alpha_1, \ldots, \alpha_n \in$* **Teorema**$(\Gamma)$*, entonces $f[\alpha_1, \ldots, \alpha_n] \in$* **Teorema**$(\Gamma)$ *(siempre que la función parcial f se pueda aplicar a dichos elementos).*

Demostración.

El primer punto es claro, puesto que existe una deducción de longitud 1 de todos los elementos de $\Lambda \cup \Gamma$.

Sea $\beta_{1i}, \beta_{2i}, \ldots, \beta_{ni}$ una deducción de α_i, que existe pues es un teorema. Colocamos en orden todas las deducciones de $\alpha_1, \alpha_2, \ldots$ y le añadimos $f[\alpha_1, \ldots, \alpha_n]$. Es claro que ésta es una deducción que toma como premisas el conjunto $\Lambda \cup \Gamma$ y por lo tanto el último eslabón de la cadena es un teorema de Γ. $\square$

A pesar de que las reglas de inferencia pueden no estar definidas globalmente observamos que **Teorema**(Γ) está generado por el conjunto $\Lambda \cup \Gamma$ y por las reglas R. Ya podemos enunciar el principio de inducción.

Corolario 6.2 *Sea T un conjunto tal que:*

- $(\Lambda \cup \Gamma) \subset T$.
- *T es cerrado por las reglas de inferencia R.*

Entonces **Teorema**$(\Gamma) \subset T$.

De este corolario se obtienen una serie de resultados, todos ellos inmediatos, que se emplean habitualmente en las deducciones. Enunciamos algunos de ellos.

- Sea $\Gamma \subset \Delta$. Si $\Gamma \vdash \alpha$ entonces $\Delta \vdash \alpha$. Si aumentamos el conjunto de premisas el conjunto de teoremas aumenta.
- Si $\Gamma \vdash \alpha$ entonces existe un conjunto finito $\Gamma' \subset \Gamma$ tal que $\Gamma' \vdash \alpha$. Aunque se tenga un colección infinita de premisas, en cada deducción solamente se emplea un número finito de ellas. Lo mismo es cierto para el conjunto de axiomas.
- La regla de transitividad. Si $\Gamma \vdash \beta$ para todo elemento β de un conjunto B (lo que indicaremos $\Gamma \vdash B$) y si además $B \vdash \alpha$, entonces $\Gamma \vdash \alpha$. Si en la deducción de α utilizamos teoremas previamente demostrados, es posible deducir α directamente de las premisas iniciales. Para ello basta con concatenar las deducciones.

Ahora que ya tenemos la teoría general de los sistemas deductivos, vamos a aplicarlo a un caso particular. Antes de empezar debemos hacer una precisión.

En matemáticas, al final, lo que nos interesa es la semántica. Si queremos demostrar enunciado α, nos sentimos satisfechos si en vez de α demostramos β siempre que α y β sean (semánticamente) equivalentes. Como la semántica no necesita de todos los conectores, podemos realizar deducciones empleando solamente un conjunto completo de conectivas. Por ello supondremos que las fórmulas que queremos demostrar están construidas a partir de los conectores $\{\neg, \rightarrow\}$. Por ello no se pierde generalidad si llamamos **Form** al conjunto de fbf generadas únicamente con los conectivos $\{\neg, \rightarrow\}$. Haremos, sin mencionarlo, la siguiente suposición:

Llamamos **Form** al conjunto generado por $\mathcal{V}$ y las operaciones $E_\neg$ y $E_\to$.

Definición 6.7 *El sistema deductivo con el trabajaremos tiene como axiomas*

- $\alpha \to (\beta \to \alpha)$.
- $(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$
- $(\neg\beta \to \neg\alpha) \to ((\neg\beta \to \alpha) \to \beta)$

donde α, β y γ son fbf arbitrarias. La única regla de deducción es el modus ponens:

$$\frac{\alpha \qquad \alpha \to \beta}{\beta}$$

Aunque parezca que existen solamente tres axiomas en realidad existen infinitos. En cualquiera de las expresiones anteriores podemos sustituir α, β y γ por cualquier fbf, obteniendo un nuevo axioma.

Proposición 6.3 *Todos los axiomas son tautologías.*

Demostración.

Basta realizar las tabla de verdad. □

Corolario 6.4 *Todos los teoremas lógicos son tautologías.*

Demostración.

Como el *modus ponens* es correcto, entonces cualquier consecuencia es verdadera siempre que sean verdaderas las premisas. Como los axiomas siempre son ciertos, entonces las consecuencias también. □

Definición 6.8 *Decimos que un cálculo deductivo es* `correcto` *si para cualquier conjunto de premisas* Γ *y para cualquier deducción* β *se tiene*

$$\Gamma \vdash \beta \Rightarrow \Lambda \cup \Gamma \vDash \beta$$

Como en nuestro caso los axiomas son tautologías el añadir el conjunto de axiomas no aporta nada. Podemos decir entonces que el sistema deductivo es correcto si

$$\Gamma \vdash \beta \Rightarrow \Gamma \vDash \beta$$

Naturalmente la corrección del sistema deductivo tiene mucho que ver con la corrección de las reglas de inferencia.

Teorema 6.5 (de corrección) *Nuestro sistema deductivo es correcto.*

Demostración.

Supongamos que $\Gamma \vdash \beta$. Entonces existe una deducción

$$\alpha_1, \alpha_2, \ldots, \alpha_n = \beta$$

desde el conjunto $\Lambda \cup \Gamma$. Tomemos una valuación de verdad v tal que $v[\Gamma] = 1$. Probemos por inducción que todos los eslabones de la deducción son verdaderos bajo esta valuación. Necesariamente α_1 es un axioma o un elemento de Γ. En ambos casos se cumple $v[\alpha_1] = 1$ (recordar que los axiomas son tautologías). Supongamos que $v[\alpha_i] = 1$ para todo $i < r$. Si α_r es axioma o premisa entonces es verdadera. En caso contrario debe existir un eslabón anterior de la forma $\alpha_k = \alpha_i \rightarrow \alpha_r$, siendo i estrictamente menor que r. Como la regla de inferencia es correcta, de la verdad de las premisas (α_i y α_k) se deduce la verdad de la consecuencia (α_r). $\square$

Los sistemas deductivos que cumplen el recíproco del teorema anterior se llaman completos.

Definición 6.9 *Un sistema deductivo es* `completo` *si* $\Gamma \vDash \beta \Rightarrow \Gamma \vdash \beta$.

Nuestro sistema deductivo también es completo, pero la demostración no es tan sencilla como la de correción.

7. Completitud de la lógica proposicional

Realizaremos ahora unas cuantas deducciones, pero solo las estrictamente necesarias para demostrar el teorema de completitud. Además, escribiremos las deducciones en vertical, añadiendo comentarios a su lado para que sean más fáciles de seguir. Debemos recordar que en cualquier momento podemos cambiar cualquier fbf que aparezca en la demostración por otra o por cualquier combinación de ellas, siempre que lo hagamos en todas sus entradas.

Proposición 7.1 *Para cualquier* $\alpha \in$ **Form** *se tiene* $\vdash \alpha \to \alpha$.

Demostración.

Una posible demostración es

1	$\alpha \to (\beta \to \alpha)$	Ax. 1
2	$(\alpha \to (\beta \to \alpha)) \to ((\alpha \to \beta) \to (\alpha \to \alpha))$	Ax. 2
3	$(\alpha \to \beta) \to (\alpha \to \alpha)$	MP(1,2)
4	$(\alpha \to (\gamma \to \alpha)) \to (\alpha \to \alpha)$	β por $\gamma \to \alpha$ en 3
5	$\alpha \to (\gamma \to \alpha)$	Ax. 1
6	$\alpha \to \alpha$	MP(5,6)

y obtenemos el resultado deseado. □

Proposición 7.2 *Si* $\{\Gamma, \alpha\} \vdash \beta$ *entonces* $\Gamma \vdash (\alpha \to \beta)$.

Demostración.

La haremos por inducción. Llamemos D al conjunto

$$D = \{\beta \text{ tales que } \Gamma \vdash \alpha \to \beta\}$$

Para demostrar que este conjunto contiene a los teoremas de $\{\Gamma \cup \alpha\}$ debemos demostrar que contiene al conjunto y que es cerrado por la regla de inferencia.

- Supongamos que $\beta \in \Gamma$. Entonces es claro que $\Gamma \vdash \beta$. Ahora debemos demostrar que $\alpha \to \beta$ se puede deducir de $\Gamma \cup \alpha$

1	β	Puesto que $\Gamma \vdash \beta$
2	$\beta \to (\alpha \to \beta)$	Ax. 1
3	$\alpha \to \beta$	MP(1,2)

Esto prueba que $\beta \in D$.

- Por la proposición anterior $\alpha \to \alpha$. Entonces $\alpha \in D$.
- Supongamos que β y $\beta \to \gamma$ pertenecen a D. Debemos demostrar que γ también pertenece a D.

terminar

El recíproco del teorema anterior también es cierto y es muy utilizado en la realización de deducciones

Proposición 7.3 *Si* $\Gamma \vdash \alpha \to \beta$ *entonces* $\{\Gamma, \alpha\} \vdash \beta$.

Demostración.

Como $\Gamma \vdash \alpha \to \beta$, entonces, añadiendo una premisa tenemos también que $\{\Gamma, \alpha\} \vdash \alpha \to \beta$. Como también $\{\Gamma, \alpha\} \vdash \alpha$, aplicando el modus ponens tenemos que $\{\Gamma, \alpha\} \vdash \beta$. $\square$

El conectivo $\to$ presenta unas leyes de transitividad, análogas a las de la implicación

Corolario 7.4 *Se cumplen las propiedades:*

- $\{\alpha \to \beta, \beta \to \gamma\} \vdash \alpha \to \gamma$.
- $\{\alpha \to (\beta \to \gamma), \beta\} \vdash \alpha \to \gamma$.

Definición 7.1 *Un sistema deductivo es* `inconsistente` *si existe un fbf* β *tal que* $\vdash \beta$ *y también* $\vdash \neg\beta$*. Se dice que es* `consistente` *en caso contrario.*

Nuestro sistema deductivo es consistente, como se deduce del teorema de corrección. Si fuese inconsistente entonces tendríamos $\Gamma \vDash \beta$ y $\Gamma \vDash \neg\beta$, lo cual es imposible.

8. Lenguajes de primer orden. Términos

Una frase declarativa, a nivel proposicional, es una variable p_n y no se considera que tenga estructura. Ahora necesitamos un lenguaje que, aproximadamente, nos dé la estructura *sujeto-predicado* propia de las oraciones de los lenguajes naturales. Los lenguajes de primer orden permiten resolver este problema. Además, prácticamente cualquier expresión matemática se puede traducir a este lenguaje.

Definición 8.1 *El alfabeto A de un* `lenguaje de primer orden` *consta de varios suconjuntos disjuntos, que enumeramos a continuación:*

- `Variables`*: Un conjunto infinito numerable $\mathcal{V}$, cuyos elementos denotamos por x_i. También utilizaremos las letras $x, y, z, \ldots$ para denotarlas.*
- `Conectivos`*: $\{\neg, \wedge, \vee, \rightarrow\}$*
- `Cuantificadores`*: $\{\forall, \exists\}$. $\forall$ se lee «para todo» y $\exists$ se lee «existe al menos un».*
- `Paréntesis`*: $\{(,)\}$.*
- `Constantes`*: Un conjunto $\mathcal{C}$ cuyo cardinal puede ser arbitrario. Los elementos los denotaremos por c_i.*
- `Functores`*: Para cada $n \geq 1$ un conjunto $\mathcal{F}_n$ de cardinal arbitrario. Utilizaremos las letras f y g para denotarlos.*
- `Relatores`*: Para cada $n \geq 1$ un conjunto $\mathcal{R}_n$ de cardinal arbitrario. Utilizaremos letras latinas mayúsculas para denotarlos. Los relatores también se llaman* `predicados`*.*

Denotaremos por $\mathcal{F}$ al conjunto $\bigcup_{n=1}^{\infty} \mathcal{F}_n$ y por $\mathcal{R}$ al conjunto $\bigcup_{n=1}^{\infty} \mathcal{R}_n$. Estos conjuntos pueden tener cualquier cardinal. En particular, nada impide que el conjunto de funciones sea vacío. Del mismo modo puede ocurrir que el conjunto de constantes sea vacío. Pero siempre vamos a suponer que existe

un relator de orden 2. Lo denotaremos por $\approx$ y diremos que es el `símbolo de igualdad`.

Observación. Nosotros consideramos que no existen functores de orden cero. En otros libros se llaman functores de orden cero a lo que nosotros llamamos constantes. Veremos en la parte de semántica la razón de ello. $\square$

A diferencia del lenguaje proposicional, existen muchos lenguajes de primer orden. Llamaremos `alfabeto lógico` al conjunto formado por las variables, los conectivos, los cuantificadores, los paréntesis y el símbolo de igualdad. Todos los lenguajes de primer orden contienen al alfabeto lógico. Además, cada lenguaje en particular tiene sus propios símbolos. Este conjunto lo llamaremos `vocabulario` y consta de las constantes, los functores y los relatores (salvo el relator igualdad).

Ejemplos.

- El lenguaje de primer orden más simple es el que únicamente tiene símbolos lógicos y carece de vocabulario. Normalmente se llama `lenguaje de igualdad`.
- Como el conjunto de símbolos lógicos es numerable, el cardinal del alfabeto A es
$$\aleph_0 + |\mathcal{C}| + |\mathcal{F}| + |\mathcal{R}|$$
- Imaginemos que queremos hablar de aritmética elemental. Para ello necesitamos un símbolo que denote nuestro concepto de cero. No buscamos más y denotamos dicho elemento por 0. Del mismo modo debemos tener un símbolo para denotar a la unidad. Si queremos realizar operaciones, necesitamos un símbolo para la suma, y otro para la multiplicación. El vocabulario $\{0, 1, +, \cdot\}$ está formado por dos constantes y por dos functores de orden 2.
- El lenguaje de la teoría de conjuntos tiene un único relator binario $\in$, y no tiene ni constantes ni functores.

- Si queremos hablar de teoría de grupos necesitaremos un símbolo para la operación y otro símbolo para denotar el elemento neutro. El vocabulario es: $\mathcal{C} = \{e\}$, $\mathcal{F} = \{+\}$ y $\mathcal{R} = \varnothing$.

- Podemos utilizar otro lenguaje para estudiar la teoría de grupos. Simplemente añadimos al lenguaje anterior un functor de orden 1, $-$. El nuevo vocabulario es ahora: $\mathcal{C} = \{e\}$, $\mathcal{F} = \{+, -\}$ y $\mathcal{R} = \varnothing$. Se pueden utilizar otros lenguajes para la teoría de grupos. Por ejemplo el que tiene vocabulario $\mathcal{C} = \{e\}$, $\mathcal{F} = \varnothing$ y $\mathcal{R} = \{R_+\}$, donde R_+ es un relator de orden 3. Posteriormente veremos como su puede construir un relator de orden $n + 1$ a partir de un functor de orden n. En definitiva, el lenguaje que permite estudiar una teoría no es único.

- Para hablar de una relación de orden en un conjunto necesitamos un símbolo para indicar dicha relación. Normalmente se utiliza $\leq$, que es un relator binario. Este lenguaje no necesita constantes ni funciones.

- Si tenemos dos lenguajes de primer orden, decimos que L' es una `extensión` de L si se cumplen las siguientes tres condiciones evidentes:

$$\mathcal{C} \subset \mathcal{C}' \qquad \mathcal{F} \subset \mathcal{F}' \qquad \mathcal{R} \subset \mathcal{R}'$$

  También se dice que L es una `restricción` de L'.

Con el alfabeto de un lenguaje de primer orden pretendemos definir un conjunto de fórmulas bien formadas. El proceso de creación es similar al del cálculo proposicional, empleando un conjunto generador y unas reglas de formación. Existen dos diferencias fundamentales con el caso proposicional:

- El conjunto de reglas de formación es ahora un conjunto infinito.

- El conjunto generador, que en el caso proposicional era $\mathcal{V}$, no está dado directamente por el alfabeto. Lo tendremos que construir. Los elementos de este conjunto generador es lo que llamaremos fórmulas atómicas. Pero para construir estas fórmulas atómicas necesitamos primeramente el concepto de término.

Dado un lenguaje de primer orden, si el conjunto de functores no es vacío, a cada elemento $f \in \mathcal{F}_n$ le asociamos una operación interna de rango n, que denotamos T_f, definida en el conjunto de expresiones

$$\begin{array}{rccc} T_f: & A^* \times \overset{n)}{\cdots} \times A^* & \mapsto & A^* \\ & (X_1, \ldots, X_n) & \mapsto & fX_1 \ldots X_n \end{array}$$

donde la expresión $fX_1 \ldots X_n$ es la concatenación del functor f, seguido de la expresión X_1, etc. Naturalmente el conjunto $\{T_f\}$ puede ser infinito si el conjunto de functores lo es.

Observación. A veces, para que la escritura esté más próxima a los convenios que habitualmente se emplean en matemáticas, escribimos $f(t_1 \ldots t_n)$ en lugar de $ft_1 \ldots t_n$. □

Definición 8.2 *Llamamos* **Ter** *al subconjunto generado por* $\mathcal{V} \cup \mathcal{C}$ *y las funciones* T_f*. Los elementos de este conjunto se llamarán* `términos`*. En general emplearemos las letras* t *y* s *para referirnos a los términos.*

Por lo tanto toda variable y toda constante, caso de existir, son términos. Si no existen functores estos son los únicos términos. Pero en cuanto exista una functor f de orden n, la expresión $ft_1 \ldots t_n$ también es un término. Este proceso inductivo de generación da lugar al concepto de cadena de formación de términos. Este concepto es análogo al de fbf que vimos en el cálculo proposicional.

Definición 8.3 *Llamamos* `cadena de formación de términos` *a toda sucesión* $X_1, \ldots, X_n \in A^*$ *que cumple una de estas condiciones:*

- X_i *es una variable o una constante.*
- *Dado* i*, existen* $i_1, \ldots, i_k$ *índices menores que* i *y un elemento* $f \in \mathcal{F}_k$ *tal que* $X_i = fX_{i_1} \ldots X_{i_k}$*.*

Se dice que un elemento $X \in A^*$ admite una cadena de formación de términos si es el último eslabón de alguna de las cadenas definidas anteriormente. Del mismo modo que en el cálculo proposicional se demuestra la

Proposición 8.1 *Un elemento $t \in$* **Ter** *si y solo si posee una serie de composición de términos.*

Ejemplos.

- En el lenguaje de la igualdad los únicos términos son las variables.
- En el lenguaje aritmético podemos crear ya nuevos términos. Por ejemplo $+(11)$ es un término (recordemos lo dicho sobre los paréntesis). Normalmente esto no lo escribimos así. Empleamos la notación habitual de la aritmética y lo denotamos por $(1+1)$. Otro término es $\cdot(+(11)+(11))$. Es fácil hallar la traducción de este término y en general de cualquier otro.
- De la misma forma, en el lenguaje de los conjuntos ordenados se emplea un relator binario $\leq$. En lugar de escribir $\leq x_1x_2$, la notación habitual es $x_1 \leq x_2$.
- Si L' es una extensión de L, todo término de L se puede considerar también como un término de L'. Al hacer extensiones ampliamos el conjunto de términos.

Obsrvación. Hemos visto que el lenguaje de la aritmética es muy formal y que nosotros podemos traducirlo siempre a un lenguaje más próximo al que estamos acostumbrados. Esto es habitual en todos los lenguajes formalizados. Cada parte de la matemática, e incluso diría que cada matemático en particular, tiene sus propios convenios de notación. Nosotros empleamos muchos convenios. Algunos los explícitamos, pero otros los tenemos ya tan interiorizados que los empleamos sin darnos cuenta. □

El método empleado para construir el conjunto de términos es recursivo. Todas las construcciones recursivas tienen un principio de inducción asociado, que se utiliza para probar propiedades que poseen todos los términos.

Proposición 8.2 (Principio de inducción) *Si un subconjunto L cumple:*

- *Contiene a las variables y a las constantes.*

- *Es cerrado por las operaciones* $\{T_f\}$.

Entonces L *contiene a* **Ter**.

Hemos demostrado que todo término es o bien una variable, o bien una constante o bien una expresión que comienza con un functor. Es claro que estos tres tipos son disjuntos. Además de esto se tiene el siguiente resultado.

Proposición 8.3 (lectura única) *Sea* $t_1 \dots t_n = s_1 \dots s_k$ *donde* t_i, s_i *son términos. Entonces se tiene que* $n = k$ *y* $t_i = s_i$ *para todo índice.*

Demostración.

Realizaremos la demostración por inducción sobre la longitud modificada. Cometeremos el abuso de notación de escribir long para denotarla. Con esta notación se tiene

$$\mathsf{long}[ft_1 \dots t_n] = 1 + \mathsf{long}[t_1] + \cdots + \mathsf{long}[t_n]$$

Si la longitud es la unidad, entonces necesariamente es una variable o una constante y se tiene la igualdad. Supongamos, por hipótesis de inducción, que cualquier cadena de términos de longitud menor que r posee la propiedad.

Si la primera letra de la expresión $t_1 \dots t_n$ es una variable o una constante, entonces t_1 es una variable o una constante. Lo mismo le sucede a s_1. Eliminando este primer término, obtenemos una igualdad entre dos sucesiones de términos de menor longitud. Por hipótesis de inducción concluimos.

Si la primera letra de la expresión es un functor f, necesariamente la otra cadena también comienza por f. Si eliminamos esta primera letra, nos queda una igualdad entre dos sucesiones cuya longitud es menor. De nuevo concluimos por inducción. □

Corolario 8.4 *Las aplicaciones* $T_f : \mathbf{Ter} \times \overset{n)}{\cdots} \times \mathbf{Ter} \mapsto \mathbf{Ter}$ *son inyectivas. Además, el conjunto generador,* $\mathcal{V} \cup \mathcal{C}$*, y los conjuntos imagen de* T_f *son disjuntos dos a dos.*

Demostración.

Veamos que T_f es inyectiva. Si

$$T_f(t_1, \ldots, t_n) = T_f(s_1, \ldots, s_n)$$

entonces tenemos que

$$ft_1 \ldots t_n = fs_1 \ldots s_n$$

Por la proposición anterior, $t_i = s_i$ y la aplicación es inyectiva.

Es claro que el conjunto generador y los de tipo $\mathrm{Im}(T_f)$ son disjuntos, pues los elementos del conjunto generador comienzan por una variable o una constante y los del otro conjunto comienzan por el símbolo f. El mismo razonamiento prueba que los conjuntos imagen son disjuntos dos a dos. □

Observación. En Álgebra, si tenemos un conjunto generador G y unas operaciones generadoras $\{f_i\}$, se dice que el conjunto $\langle G \rangle$ está `libremente generado` si los conjuntos G y las imágenes de las aplicaciones generadoras son disjuntos dos a dos. Con esta notación, nosotros hemos probado que el conjunto de términos está libremente generado por el conjunto de variables y constantes. □

Los resultados anteriores permiten utilizar el `método recursivo` para definir funciones sobre el conjunto de términos. La ideal general es la siguiente: tomamos un conjunto B como conjunto imagen. A cada elemento del conjunto generador le asociamos un elemento de B. Construimos así una función $\phi : \mathcal{V} \cup \mathcal{C} \mapsto B$. Para extender dicha función a términos compuestos definimos $\phi[ft_1 \ldots t_n]$ en función de $\phi[t_1], \ldots, \phi[t_n]$. Esta construcción es correcta debido a la lectura única. Como cualquier término compuesto tiene una serie de formación, podemos emplear el procedimiento anterior en cada eslabón de una cadena de formación, obteniendo finalmente la imagen de cualquier término compuesto.

Ejemplos.

- Decimos que t es un `subtérmino` de s si $s = \alpha t \beta$ y t es a la vez un

término. Llamamos subter a la única función

$$\mathsf{subter} : \mathbf{Ter} \mapsto \mathcal{P}(\mathbf{Ter})$$

que cumple:

- $\mathsf{subter}[c_i] = \{c_i\}$
- $\mathsf{subter}[x_i] = \{x_i\}$
- $\mathsf{subter}[ft_1 \dots t_n] = \{ft_1 \dots t_n\} \cup \mathsf{subter}[t_1] \cup \dots \cup \mathsf{subter}[t_1]$

La imagen $\mathsf{subter}[t]$ da el conjunto de todos los subtérminos de t. Es habitual utilizar árboles para encontrar todos los subtérminos. En cada paso se utiliza el resultado de lectura única para crear nuevas ramas. El proceso se detiene cuando se llega a una variable o a una constante.

- El `grado de complejidad` de un término es el número de functores que posee, contados tantas veces como aparece. Si la denotamos por $\mathsf{comp} : \mathbf{Ter} \mapsto \mathbb{N}$, es la única función que cumple:

  - $\mathsf{comp}[x_i] = 0$.
  - $\mathsf{comp}[c_i] = 0$.
  - $\mathsf{comp}[ft_1 \dots t_n] = 1 + \mathsf{comp}[t_1] + \dots + \mathsf{comp}[t_n]$.

- Denotamos por $\mathsf{var}[t]$ al conjunto de variables que tienen entrada en t. La definición recursiva de esta función es:

  - $\mathsf{var}[c_i] = \varnothing$.
  - $\mathsf{var}[x_i] = \{x_i\}$.
  - $\mathsf{var}[ft_1 \dots t_n] = \mathsf{var}[t_1] \cup \dots \cup \mathsf{var}[t_n]$.

  Si un término t cumple que $\mathsf{var}[t] \subset \{x_1, \dots, x_n\}$, lo denotaremos por $t[x_1, \dots, x_n]$. Esta notación quedará clara cuando estudiemos la semántica.

9. Lenguajes de primer orden. Fórmulas

De momento solamente hemos empleado las variables, las constantes y los functores. Para construir las fbf necesitamos ya los predicados.

Definición 9.1 *Llamamos* `fórmula atómica` *a cualquier expresión de la forma* $Rt_1 \dots t_n$ *donde* R *es un relator de orden* n *y* t_i *son términos. En particular* $t_1 \approx t_2$ *es siempre una fórmula atómica. El conjunto de todas las fórmulas atómicas lo denotamos* **Atm**.

Ejemplos.

- En el lenguaje de la igualdad $\approx x_1x_2$ es una fórmula atómica. En este lenguaje todas las fórmulas atómicas son similares, puesto que los únicos términos son las variables y el único relator es el de igualdad. La escritura más habitual de esta fbf es $x_1 = x_2$.
- En aritmética $\approx x_10$ es una fórmula atómica. Normalmente escribiremos $x_1 \approx 0$. Si no hay peligro de confundir el signo $\approx$ con el que a veces empleamos en el castellano, escribiremos $x_1 = 0$. Otra fórmula atómica, ya traducida, es $(1 + 1) \cdot (0 + 1 + 1) = (x_1 + 1)$.
- En el lenguaje de la teoría de grupos, si denotamos por un punto a la operación interna y por e al «elemento neutro», $e \cdot x_7 = (e \cdot x_1) \cdot e$ es una fórmula atómica.

Ya hemos construido el conjunto que nos permitirá generar inductivamente el conjunto de fbf. Ahora necesitamos construir operaciones internas que nos sirvan de reglas de formación. Algunas de ellas son exactamente iguales que las del cálculo proposicional y las denotaremos $E_\neg$, $E_\wedge$, $E_\vee$ y $E_\to$. Para cada natural n tendremos otras dos funciones de rango 1. Sus definiciones son:

$$\begin{array}{rccc} F_i: & A^* & \mapsto & A^* \\ & X & \mapsto & \forall x_i(X) \end{array} \qquad\qquad \begin{array}{rccc} G_i: & A^* & \mapsto & A^* \\ & X & \mapsto & \exists x_i(X) \end{array}$$

Definición 9.2 *Llamamos* **Form** *al conjunto generado por* **Atm** *con las reglas de formación E, F_i y G_i. Sus elementos se llaman* `fórmulas` *o abreviadamente* `fbf`*.*

De nuevo se tienen los conceptos de cadena de formación y principio de inducción.

Ejemplos.

- En el lenguaje de la igualdad $\forall x_1(x_7 = x_1)$ es una fórmula `compuesta` (compuesta es lo contrario de atómica). También es una fbf $\neg(\approx x_1 x_2)$ que normalmente escribimos $x_1 \neq x_2$.
- En el lenguaje de la teoría de grupos $\forall x \forall y \forall z\big(x \cdot (y \cdot z) = (x \cdot y) \cdot z\big)$ es una fbf. Nuestra intuición de matemáticos nos sugiere que esta fórmula expresa la propiedad asociativa, y que como estamos en un grupo, debe de ser cierta. De momento no podemos pensar así. Esto es simplemente una expresión en un lenguaje de primer orden. Cuando hablemos de semántica ya podremos asociar un valor de verdad a dicha fórmula.
- En el lenguaje de los conjuntos ordenados, donde $\leq$ es el único elemento del vocabulario, $\forall x(\exists y(x \leq y))$ es una fbf, donde hemos escrito $x \leq y$ en lugar de $\leq xy$.

La siguiente proposición se prueba del modo habitual por inducción.

Proposición 9.1 *El conjunto de fórmulas está libremente generado por las fórmulas atómicas.*

Al estar libremente generado podemos hablar del `tipo` de cada fbf. Cada fbf γ es de alguno de los siguientes tipos:

- Fórmula atómica.
- $\gamma = \neg\alpha$.
- $\gamma = \alpha \wedge \beta$.

- $\gamma = \alpha \vee \beta$.
- $\gamma = \alpha \to \beta$.
- $\gamma = \forall x_i(\alpha)$ para cierta variable x_i.
- $\gamma = \exists x_i(\alpha)$ para cierta variable x_i.

Al estar libremente generado se tiene un principio de lectura única que nos permite definir funciones recursivamente. Para ello debemos asociar a cada fórmula atómica la imagen de la función y definir recursivamente la función para el resto de tipos.

Ejemplos.

- El `grado de complejidad` de una fórmula es el número de conectivos y cuantificadores que aparecen en su escritura. Recursivamente se define:
  - $\mathsf{grado}[\alpha] = 0$ si α es atómica.
  - $\mathsf{grado}[\neg\alpha] = 1 + \mathsf{grado}[\alpha]$.
  - $\mathsf{grado}[\alpha \star \beta] = 1 + \mathsf{grado}[\alpha] + \mathsf{grado}[\beta]$.
  - $\mathsf{grado}[\forall x_i(\alpha)] = 1 + \mathsf{grado}[\alpha]$.
  - $\mathsf{grado}[\exists x_i(\alpha)] = 1 + \mathsf{grado}[\alpha]$.
- Tenemos asimismo el concepto de `subfórmula`. Las fórmulas atómicas no tienen subfórmulas y la construcción recursiva de la función $\mathsf{subfor} : \mathbf{Form} \mapsto \mathcal{P}(\mathbf{Form})$ es:
  - $\mathsf{subfor}[\alpha] = \{\alpha\}$ si α es atómica.
  - $\mathsf{subfor}[\neg\alpha] = \{\neg\alpha\} \cup \mathsf{subfor}[\alpha]$.
  - $\mathsf{subfor}[\alpha \star \beta] = \{\alpha \star \beta\} \cup \mathsf{subfor}[\alpha] \cup \mathsf{subfor}[\beta]$.
  - $\mathsf{subfor}[\forall x_i(\alpha)] = \{\forall x_i(\alpha)\} \cup \mathsf{subfor}[\alpha]$.
  - $\mathsf{subfor}[\exists x_i(\alpha)] = \{\exists x_i(\alpha)\} \cup \mathsf{subfor}[\alpha]$.

- Ayudado con el concepto de subfórmula se pueden crear sucesiones de formación de fbf. Por ejemplo, si c es una constante la sucesión

$$\begin{aligned} X_1 &= (x = y) \\ X_2 &= (x = c) \\ X_3 &= (x = y) \wedge (x = c) \\ X_4 &= \forall x((x = y) \wedge (x = c)) \end{aligned}$$

  es una serie de formación de $\forall x((x = y) \wedge (x = c))$. Aquí aparecen dos signos $=$. Uno forma parte el lenguaje de primer orden y otro forma parte del argot matemático. No debemos confundirlos.

Observación. Al aplicar el concepto de recursión estamos a la vez dando un algoritmo de cálculo de todos los conceptos recursivos. En un número finito de pasos, cada uno de ellos perfectamente determinado, podemos construir la imagen de cualquier elemento. □

Definición 9.3 *Dado un símbolo $a \in A$, decimos que tiene una* `entrada` *en una fbf α si $\alpha = \beta a \gamma$.*

Un mismo símbolo puede tener distintas entradas en una fbf. También puede ocurrir que dicho símbolo no entre en la fórmula.

Definición 9.4 *Si $\forall$ es una entrada en α, llamamos* `alcance` *del cuantificador a la única subfórmula β, tal que $\forall x_i(\beta)$ es también subfórmula. Análoga definición para el otro cuantificador.*

Decimos que una entrada de una variable $y \in \mathcal{V}$ de una fbf α está `ligada` si pertenece a alguna subfórmula β tal que o bien $\forall y(\beta)$ o bien $\exists y(\beta)$ es una subfórmula. En otras palabras, una variable está ligada si pertenece al alcance de un cuantificador (seguido de la variable en cuestión). Las otras variables que aparecen en α se llaman `libres`.

Observación. Aunque en rigor debemos hablar de entrada libre de una variable, normalmente hablaremos de `variables libres`. Esto puede inducir

a error puesto que una misma variable puede tener entradas libres y ligadas. Si decimos que α tiene como variable libre y, queremos decir que alguna (o varias) entrada de y en α es libre. $\square$

Denotaremos por $\mathsf{lib}[\alpha]$ al conjunto de variables libres de α. Las fórmulas atómicas tienen como variables libres las variables de sus términos. La definición recursiva de la función lib es:

- Si $\alpha = Rt_1 \dots t_n$ es atómica entonces $\mathsf{lib}[\alpha] = \mathsf{var}[t_1] \cup \dots \cup \mathsf{var}[t_n]$.
- $\mathsf{lib}[\neg\alpha] = \mathsf{lib}[\alpha]$.
- $\mathsf{lib}[\alpha \star \beta] = \mathsf{lib}[\alpha] \cup \mathsf{lib}[\beta]$.
- $\mathsf{lib}[\forall y(\alpha)] = \mathsf{lib}[\alpha] - \{y\}$.
- $\mathsf{lib}[\exists y(\alpha)] = \mathsf{lib}[\alpha] - \{y\}$.

Ejemplos.

- Si R es un relator binario, Rx tiene una variable libre, que es x. Sin embargo $\forall x(Rx)$ no tiene variables libres, pues la entrada libre de x, se ve «ligada» por el cuantificador.
- $\forall x(Rx) \wedge Px$ tiene una variable libre. La segunda entrada de x (la de Rx) está ligada. Sin embargo la tercera entrada no está en el alcance del cuantificador. Basta que exista una entrada libre de x para que digamos que la fórmula posee una variable libre.
- Si la fórmula no tiene cuantificadores, todas las variables que entren en la fórmula son libres.
- Dada una fbf α denotaremos por $\alpha[x_1, \dots, x_n]$ si $\mathsf{lib}[\alpha] \subset \{x_1, \dots, x_n\}$. Debemos esperar también a la parte de semántica para comprender la notación.

Definición 9.5 *Si una fbf tiene alguna variable libre se dice que es* `abierta`. *Si no posee variables libres se denomina* `cerrada`. *Las fbf cerradas también*

se llaman `sentencias`. *El conjunto de todas las sentencias de un lenguaje se denota* **Sent**.

Nosotros estamos fundamentalmente interesados en las sentencias, puesto que cuando estudiemos la semántica, le asociaremos a cada sentencia un valor de verdad. A las fórmulas abiertas la semántica no le asociará un valor de veracidad. Por ello es importante construir sentencias a partir de fbf libres. Existen dos métodos fundamentales para «cerrar» una sentencia. Imaginemos que α tiene una única variable libre y (lo que hemos notado $\alpha[y]$). Entonces $\forall y(\alpha)$ ya no posee variables libres. Lo mismo puede realizarse con el otro cuantificador. Si tiene más de una variable libre se pueden «cuantificar» una a una. En principio el orden de cuantificación puede ser arbitrario.

La otra forma de hacer que desaparezcan las variables libres es sustituirlas por constantes. Supongamos que una fbf tiene una única variable libre y. Si en todas las entradas libres de dicha variable sustituimos dicha variable por una constante c obtenemos una nueva expresión. No es difícil comprobar que esta nueva expresión es también una fbf. Debemos hacer notar que cuando decimos «sustituir» la variable x por la constante c, debemos hacerlo solamente en las entradas libres de la variable. Las entradas ligadas no se pueden sustituir por constantes puesto que si una fórmula es $\forall y(\alpha)$, si sustituimos y por una constante obtenemos $\forall c(\dots)$, que no es una fbf. Es sencillo dar una definición recursiva de este concepto.

10. Sistemas

Los lenguajes de primer orden se utilizan para estudiar sistemas. Los sistemas son estructuras algebraicas dotadas de operaciones y relaciones internas. Muchas de las estructuras algebraicas que aparecen habitualmente en el trabajo matemático son casos especiales de esta definición. Cada lenguaje de primer orden tiene una clase de sistemas asociado. Cada elemento de esa clase es lo que llamaremos una *interpretación* del lenguaje. El proceso también es reversible en cierto sentido: si tenemos un sistema particular, podemos crear un lenguaje de primer orden especificamente adaptado. Sin embargo los sistemas se pueden estudiar independientemente de los lenguajes de primer orden. Esto es lo que hace el *álgebra universal*. Cuando utilizamos también los lenguajes de primer orden, entonces nos introducimos en la *teoría de modelos*.

Definición 10.1 *Dado un conjunto M, una* `operación` *de orden n es una aplicación de M^n en M. Una* `relación` *de orden n es un subconjunto de M^n.*

En realidad una operación f de orden n se puede pensar como una relación R_f de orden $n+1$: dada la operación, el conjunto R_f de la relación está formado por los puntos de la forma $(a_1, \ldots, a_n, f[a_1, \ldots, a_n])$. El conjunto R_f es el gráfico de la función. Sin embargo el recíproco no es cierto. Existen relaciones en M que no son el gráfico de ninguna función.

Una función de orden 0 es una aplicación de M^0 en M. Como por convenio tenemos que $M^0 = \varnothing$, una tal función tiene como dominio un conjunto con un único elemento. Se puede identificar, en este caso, la función con su imagen. Por ello las funciones de orden cero se llaman también `constantes` o `elementos destacados` de A.

Definición 10.2 *Un* `sistema` *es una colección $\mathscr{M} = \{M, \{f_i\}_{i\in I}, \{R_j\}_{j\in J}\}$ donde M es un conjunto, f_i son funciones de cualquier orden (en M) y R_j son relaciones de cualquier orden.*

El conjunto soporte del sistema M se llama `universo`. Nada impide que el conjunto de funciones y el de relaciones sea vacío. Si el conjunto de funciones y de relaciones es finito, denotamos al sistema por

$$\mathscr{M} = \{M, f_1, \ldots, f_n, R_1, \ldots, R_m\}$$

A cada función f_i le corresponde un orden $\delta(f_i)$ determinado. Se construye de esta forma una función $\delta : I \mapsto \mathbb{N}$. Análogamente se construye una función que asigna su orden a cada relación. La denotamos por μ. Es costumbre colocar a las funciones y a las relaciones siguiendo un orden natural. Esto implica que supondremos siempre que $\delta[i] \leq \delta[k]$ si $i < k$ y análogamente con las relaciones. En particular, las constantes (si existen) son las primeras que se mencionan. Llamaremos `tipo` o `signatura` del sistema a $(\delta; \mu)$. En caso de que los conjuntos sean finitos, el tipo de un sistema se denota $(i_1, \ldots, i_n; j_i \ldots, j_m)$, lo que indica que la función f_k tiene orden i_k,... Decimos que dos sistemas son `similares` si tienen el mismo tipo.

Ejemplos.

- Un grupo es un conjunto junto con una operación interna y binaria. Además en un grupo existe un elemento destacado, el elemento neutro. Un grupo es un sistema de tipo $(0, 2; \varnothing)$. Un semigrupo también tiene una estructura del mismo tipo. Incluso si la operación no es asociativa, se tiene el mismo tipo.

- Un anillo es un conjunto, junto con dos operaciones y un elemento neutro para la suma. Su tipo es $(0, 2, 2; \varnothing)$. En un anillo con unidad o en un cuerpo, debemos destacar también la unidad. Trabajaríamos entonces con un sistema de tipo $(0, 0, 2, 2; \varnothing)$.

- Un conjunto ordenado es un par formado por un conjunto y una relación de orden. Es un sistema de tipo $(\varnothing; 2)$.

- Consideremos en $\mathbb{R}$ su estructura como cuerpo ordenado. En este caso trabajamos con dos operaciones, dos elementos neutros y una relación de orden.

- Sea el conjunto **Form** de la lógica proposicional. En este conjunto tenemos las cuatro operaciones internas $E_\neg, E_\wedge, E_\vee, E_\to$. Este es un sistema de tipo $(1, 2, 2, 2; \varnothing)$.

Observación. Lo mismo que en el estudio de las estructuras algebraicas vamos a cometer muchos abusos de notación. En álgebra, por ejemplo, hablamos del anillo A, sobrentendiendo las operaciones y elementos neutros y nombrando únicamente al conjunto. En la teoría de sistemas esta necesidad es aun mayor si queremos que la notación no sea terriblemente tediosa. Por ello muchas veces hablaremos del sistema M, sobreentendiendo que sus operaciones las denotamos f_i y a sus relaciones R_j. $\square$

En muchas teorías de tipo algebraico observamos que el comienzo del estudio de una estructura es siempre similar. Primeramente se define la estructura (por ejemplo la estructura de grupo o anillo). El siguiente paso es el estudio de las subestructuras (subgrupos y subanillos). Después se pasa a estudiar las aplicaciones entre estructuras (lo que llamamos morfismos) y ciertas características de ellos.

Definición 10.3 *Decimos que un sistema* (M, f_i, R_j) *es un* `subsistema` *de* (N, g_i, S_j) *si se cumple:*

- $M \subset N$.
- *La operación* f_i *es la restricción de la operación* g_i *al subconjunto.*
- *La relación* R_j *es la restricción de la relación* S_j *a* M.

Denotaremos el concepto de subsistema por $M \subset N$ *cometiendo un abuso de notación.*

Restringir relaciones a un subconjunto no presenta nunca problemas. Si R es una relación en B y $A \subset B$, la relación restringida es $R \cap A^n$. Sin embargo no toda operación, por ejemplo binaria, se puede restringir, pues puede ocurrir que $\alpha, \beta \in A$ y sin embargo $f[\alpha, \beta] \notin A$. El siguiente resultado es elemental.

Proposición 10.1 *Dado un sistema N y un subconjunto $M \subset N$, tenemos que M es un subsistema si y solo si M es invariante por todas las operaciones del sistema. Si el sistema carece de funciones cualquier subconjunto es subsistema.*

En particular las constantes, que son las funciones de orden nulo, deben pertenecer al subconjunto.

Corolario 10.2 *Sea (M, f_i, R_j) un sistema y N un subconjunto cualquiera del conjunto soporte M. Si denotamos por $\langle N \rangle$ al conjunto generado por N y las funciones f_i, se tiene que $\langle N \rangle$ es un subsistema.*

También es habitual en matemáticas «olvidarnos» de alguna estructura que posee un conjunto. Sabemos que en $\mathbb{R}$ podemos definir una estructura de cuerpo y una estructura de orden. Sin embargo a veces consideramos únicamente la estructura de cuerpo o incluso solamente la estructura de grupo. Dejamos el mismo conjunto pero eliminamos algunas operaciones o relaciones. Cuando hacemos esto en sistemas, decimos que `reducimos` el sistema. De la misma forma, si añadimos más funciones o relaciones a una estructura, decimos que realizamos una `ampliación`.

Ejemplos.

- Sea G un conjunto con una estructura de grupo. Si G' es un subgrupo es claro que es también un subsistema. Pero un conjunto puede ser subsistema sin ser subgrupo. La definición de subsistema obliga a que el neutro pertenezca al subsistema y que éste sea cerrado por la operación, pero no nos dice nada sobre el inverso. Por ejemplo $(\mathbb{N}, +)$ es subsistema de $(\mathbb{Z}, +)$ pero no es subgrupo.

- Si A es un subsistema de B y B es un subsistema de C, entonces también A es un subsistema de C.

- Consideremos la estructura $(\mathbb{N}, 0, S)$ donde S es la función $S(n) = n+1$. Esta estructura carece de subsistemas no triviales. En efecto, si $A \subset \mathbb{N}$ es subsistema, entonces $0 \in A$ y además, por ser cerrado por S, tenemos

que si $n \in A$ entonces $n+1 \in A$. Pero el principio de inducción nos dice que en este caso necesariamente $A = \mathbb{N}$.

- Sea ahora $(\mathbb{R}, 0, S)$. Este sistema si que presenta subsistemas no triviales. Por ejemplo $\mathbb{Z}$ es subsistema. También $\mathbb{R}^+ \cup \{0\}$ es subsistema. Dado el conjunto $\{-10\}$ el subsistema generado por este conjunto es $\{-10, -9, -8, \dots\}$. Sin embargo el subsistema generado por $1/2$ es $\{1/2, 3/2, \dots\} \cup \mathbb{N}$.

Definición 10.4 *Dadas dos estructuras del mismo tipo, llamamos* `morfismo` *de* (M, f_i, R_j) *en* (N, g_i, S_j) *a toda aplicación* $\phi : M \mapsto N$ *que cumpla:*

- *Conserva las operaciones: si* f_i *y* g_i *denotan la* i*-ésima operación en cada estructura*

$$\phi[f_i[x_1, \dots, x_n]] = g_i[\phi[x_1], \dots, \phi[x_n]]$$

  En particular para las constantes se tiene que $\phi(c) = c$.

- *Conserva las relaciones: si* R_j *y* S_j *denotan las relaciones* j*-ésimas de ambas estructuras*

$$\textit{Si } (x_1, \dots, x_n) \in R_j \textit{ entonces } (\phi[x_1], \dots, \phi[x_n]) \in S_j$$

El ejemplo más inmediato de morfismo es la inyección natural de un subsistema dentro del sistema. Los morfismos de grupos, de anillos, etc. también son morfismos de las correspondientes estructuras.

Propiedades.

- La composición de morfismos de sistemas es de nuevo un morfismo.
- Llamamos morfismos `inyectivos` a aquellos cuya función asociada sea inyectiva. La inyección canónica es un morfismo inyectivo.
- Si $\phi : M \mapsto N$ es un morfismo, la aplicación ϕ es biyectiva y la aplicación inversa $\phi^{-1} : N \mapsto M$ es también morfismo, hablaremos de un `isomorfismo`.

- La relación de isomorfía es de equivalencia. A todos los efectos dos sistemas isomorfos serán considerados iguales.

- Sea (M, f_i, R_j) un sistema y sea N un conjunto que tenga una biyección ϕ con el conjunto soporte M. En N se puede construir una estructura de sistema que haga de ϕ un isomorfismo.

El estudio del álgebra nos presenta muchos ejemplos de sistemas y morfismos. A su vez el estudio de los sistemas nos permite tener una visión general y observar que ciertos resultados que utilizamos en cada estructura algebraica son casos particulares de un resultado más general.

Ejemplos.

- Consideremos los sistemas que no tienen ni funciones ni relaciones destacados (los de tipo $(\varnothing; \varnothing)$). Los morfismos entre sistemas son aplicaciones entre conjuntos. Si un conjunto A tiene menos elementos que un conjunto B entonces existe un morfismo inyectivo de A en B. Para que dos sistemas sean isomorfos es necesario y suficiente que tengan el mismo cardinal.

- Sea $(A, \leq)$ un conjunto ordenado. Este es un sistema de tipo $(\varnothing; 2)$. Un morfismo entre dos conjuntos ordenados es un morfismo de este tipo de sistemas. En particular dos conjuntos ordenados son isomorfos si tienen el mismo ordinal. El estudio de los ordinales es una parte del estudio de los sistemas de este tipo.

11. Semántica

Nuestra intención es asociar a cada fbf un valor de verdad. Para ello basta con dar una «interpretación». Una interpretación consiste en asociar a cada símbolo no lógico un objeto matemático. Así el símbolo se transforma en algo más tangible. Con ello se consigue asociar un valor de verdad a las fbf cerradas. Para conseguir extender esta construcción a las fórmulas abiertas debemos dar también una interpretación a las variables.

El concepto de interpretación de un lenguaje se puede adaptar a varios esquemas. Se puede definir perfectamente sin saber lo que es un sistema y así se hace en muchos libros. Sin embargo el concepto de sistema ayuda a clarificar el concepto de interpretación. La definición clásica de interpretación se adapta a la

Definición 11.1 *Dado un lenguaje de primer orden L una* `interpretación` *M de L consta de los siguientes ingredientes:*

- *Un conjunto M llamado* `universo de la interpretación`*.*
- *A cada constante $c_i \in \mathcal{C}$ se le asocia un elemento $\mathbf{c_i} \in M$.*
- *A cada functor $f \in \mathcal{F}_n$ se le asocia una operación $\mathbf{f}$ de orden n en M.*
- *A cada relator $R \in \mathcal{R}_n$ se le asocia una relación $\mathbf{R}$ de orden n en M.*

Al predicado $\approx$ siempre se le debe asociar la relación binaria de igualdad, independientemente de la interpretación. Por ello este relator binario es en cierto sentido especial y no se incluye en el vocabulario, sino en los símbolos lógicos del lenguaje.

Observación. En gran parte de la literatura lo que nosotros hemos llamado functores se denominan `símbolos de función` y los relatores se denominan `símbolos de relación`. Con ayuda de esta definición podemos entender mejor dicha nomenclatura. De momento hemos denotado por el mismo símbolo al elemento del lenguaje y a su interpretación. Simplemente hemos puesto negrita a las interpretaciones. Este convenio lo iremos desechando poco a

poco y denotaremos exactamente igual al símbolo y a su interpretación. El contexto debe ser suficiente para discriminar qué significa en cada caso. □

Pero nosotros observamos que una interpretación de un lenguaje L es un sistema. La estructura de sistema creada guarda una relación muy estrecha con el lenguaje en cuestión.

Definición 11.2 *Dada un lenguaje L, llamamos L-*`sistema` *a un sistema $\mathscr{M} = (M, f_i, R_j)$ que tenga tantas operaciones como functores tenga el lenguaje y tantas relaciones como relatores tenga el lenguaje. Además los órdenes de los functores y operaciones, así como los de relatores y relaciones, deben coincidir. En particular a las constantes del lenguaje le corresponden elementos destacados del sistema.*

Por lo tanto podemos definir nuevamente el concepto de interpretación de un lenguaje. Una interpretación es un L-sistema. Esta claro que dos L-sistemas son siempre similares.

Observación. Recíprocamente, si tenemos un sistema (M, f_i, R_j) podemos crear un lenguaje de primer orden asociado. Por cada operación de orden n creamos un functor de orden n y análogamente con las relaciones. Esto permite construir el vocabulario del lenguaje. □

Ejemplos.

- Consideremos el lenguaje de la aritmética con vocabulario $\{0, 1, +, \cdot\}$. La interpretación estandard tiene como conjunto soporte $\mathbb{N}$ y las definiciones de los símbolos son las operaciones habituales de la aritmética. Tomemos un término sin variables, por ejemplo $(1 + 1) + (1 \cdot (1 + 0))$. Nuestra intuición de matemáticos nos indica que, fijada la interpretación, esta cadena de símbolos nos está indicando un elemento del conjunto $\mathbb{N}$. Sin embargo si tomamos un término con una variable, por ejemplo $(1 + x_1) + 1$ vemos que no le podemos asociar un elemento de $\mathbb{N}$. Sin embargo, si sustituimos la variable por un elemento de $\mathbb{N}$ ya podemos asociarle un número natural.

- Tomemos una estructura algebraica particular, por ejemplo el cuerpo $\mathbb{R}$. A este cuerpo se le asocia un lenguaje con vocabulario $\{0, 1, +, \cdot\}$ y ese lenguaje tiene un interpretación natural en $\mathbb{R}$. Sin embargo $\mathbb{R}$ puede ser soporte de otras interpretaciones distintas sin más que variar la definición de alguna de las operaciones o de alguna de las constantes. Puede ocurrir que bajo alguna de esas interpretaciones $\mathbb{R}$ no sea un cuerpo.

- Generalizando el ejemplo anterior, prácticamente a cualquier estructura algebraica se le asocia un lenguaje y la estructura algebraica es precisamente una interpretación de dicho lenguaje.

- Consideremos la interpretación estandard del cuerpo $\mathbb{R}$. Nuestra intuición nos dice que la fórmula $\forall x \exists y(y + y = x)$ se puede traducir por la frase: «Para todo número real x existe un número real y tal que $y + y = x$». Sabemos que esta afirmación es cierta en $\mathbb{R}$. Sin embargo la afirmación $\exists y(y \cdot y = x)$ no sabemos si es cierta. Si x «es» un número positivo sabemos que si es cierta, pero si x es negativo es falsa. Esto es debido a que la fórmula posee una variable libre y para saber si es cierta o falsa debemos asignarle un cierto valor.

- En $\mathbb{R}$ la fórmula $\exists x(x + x = y)$ es cierta para cualquier valor que le asignemos a y. Por lo tanto, en este caso, a pesar de tener una variable libre, podemos decir que la fbf es cierta. Nótese el gran parecido entre la fórmula anterior y $\forall y \exists x(x + x = y)$, que también es cierta en $\mathbb{R}$.

Proposición 11.1 *Dada una interpretación M de un lenguaje, a cada término sin variables le corresponde, de modo canónico, un elemento de M.*

Demostración.

Si denotamos por **TerCerr** al conjunto de términos cerrados (es decir sin variables), vamos a construir una función ϕ que tenga ese dominio y cuya imagen sea el conjunto A. Como siempre la definición será recursiva.

- Si c_i es una constante $\phi[c_i] = \mathbf{c_i}$, que es elemento que le corresponde a la constante por la interpretación dada.

- Si el término es de la forma $t = ft_1 \dots t_n$ entonces

$$\phi[t] = \mathbf{f}[\phi[t_1], \dots, \phi[t_n]]$$

  siendo $\mathbf{f}$ la operación que se asocia con el relator f en la interpretación dada.

Una inducción sobre el grado de complejidad nos garantiza la corrección del método de construcción. El elemento que le corresponde a t lo denotaremos $\mathbf{t}$. □

El método para calcular el elemento asociado a cada término es claro. En la expresión del término, sustituimos cada constante y cada functor por su interpretación y «operamos». Esto permite demostrar el siguiente

Corolario 11.2 *Dadas dos interpretaciones en un mismo universo M. Si las interpretaciones coinciden sobre las constantes y sobre los functores, los términos sin variables tienen la misma interpretación.*

En definitiva, para interpretar un término sin variables basta con conocer la interpretación de los símbolos lo componen. Por ello se dice que la interpretación de los términos es una `propiedad local`.

Si el término tiene una variable, la idea intuitiva es que al sustituir la variable por un elemento del universo se obtiene otro elemento del universo. Un término t con una variable induce una función $t : M \mapsto M$. Análogamente si tiene n variables se induce una función $t : M^n \mapsto M$. Sin embargo esto no se puede hacer directamente, puesto que estamos sustituyendo una variable por un elemento del universo que, en principio, no pertenece al lenguaje. Para poder realizar lo anterior debemos ampliar el lenguaje, añadiendo tantas constantes como elementos tenga el universo de la interpretación.

Definición 11.3 *Dado un lenguaje L y una interpretación M llamamos $L(M)$ al lenguaje que tiene como conjunto de constantes $\mathcal{C} \cup M$ (unión disjunta). Los conjuntos de functores y relatores coinciden con los de L.*

Este nuevo lenguaje es una ampliación de L. Todo término y toda fbf del lenguaje L es también un término o una fbf del nuevo lenguaje. La ventaja de trabajar en el nuevo lenguaje es que se pueden sustituir variables por elementos de M. De esta manera, si sustituimos todas las variables en un término, conseguimos «cerrar» dicho término y éste nuevo término ya tiene una interpretación. De la misma forma, si sustituimos todas las variables libres de una fbf cerramos la fórmula. Como veremos posteriormente esto también nos permitirá interpretar dicha fbf.

Proposición 11.3 *Dada una interpretación M de un lenguaje L se induce una interpretación canónica de $L(M)$ en el mismo conjunto.*

Demostración.

La interpretación de los functores y de los relatores coincide con la interpretación de L. Asimismo en las constantes que pertenezcan a L la interpretación coincide. Las nuevas constantes son símbolos de la forma $a \in M$. La interpretación natural es asociar al símbolo a el elemento $a \in M$. Esta es una interpretación de $L(M)$ que «extiende» a la interpretación de L. $\square$

Corolario 11.4 *La interpretación de los términos cerrados coincide en L y en $L(M)$.*

Demostración.

Es consecuencia de la localidad de la interpretación, ya que ambas interpretaciones asignan los mismos elementos a todos los símbolos que forman parte de los términos cerrados de L. $\square$

Con este lenguaje ampliado ya se puede realizar la idea anterior.

Proposición 11.5 *Cada término t con n variables induce una función $t : M^n \mapsto M$.*

Demostración.

Sea t un término en L con n variables libres. También se puede considerar como un término en $L(M)$. Sea $(a_1, \ldots, a_n)$ un elemento de M^n. Como el término t pertenece al lenguaje $L(M)$ podemos sustituir sus variables por constantes de este lenguaje. Sustituimos la primera variable por a_1, la segunda por a_2, etc. Así conseguimos un término cerrado en $L(M)$. Por un resultado anterior a este término cerrado le corresponde canónicamente un elemento de M. Así se construye la función t asociada. □

La notación habitual para un término con n variables es $t[x_1, \ldots, x_n]$. Después de esta proposición se entiende mejor la notación funcional que se suele adoptar.

Observación. Cuando tengamos una interpretación M de un lenguaje siempre podemos construir el lenguaje $L(M)$ y su interpretación. Por ello muchas veces se comete el abuso de notación de trabajar con $L(M)$ sin mencionarlo explícitamente. Esto no es problemático, pues todas las construcciones que hagamos serán independientes de que trabajemos con un lenguaje o con otro. □

Nuestra idea es ahora asociar un valor de verdad a cada fórmula cerrada de L. Denotaremos el valor de verdad de α por $v_M[\alpha]$ o simplemente $v[\alpha]$ si se sobreentiende la interpretación. Nosotros vamos a ir más allá y asociaremos un valor de verdad a cualquier sentencia cerrada del lenguaje $L(M)$. Denotaremos por $\alpha(x/a)$ a la sentencia que se obtiene sustituyendo la variable libre x por la constante a. Recordemos también que a cualquier término sin variables de $L(M)$ se le asocia un elemento de M. Como siempre el proceso será recursivo.

- Si α es atómica, sin variables y de la forma $t_1 \approx t_2$, donde los términos no pueden tener variables, entonces

$$v[\alpha] = 1 \text{ si y solo si } \mathbf{t_1} = \mathbf{t_2}$$

- Sea α otra fórmula atómica sin variables. Entonces $\alpha = Rt_1 \ldots t_n$ donde

los términos no pueden tener variables. En este caso

$$v[\alpha] = 1 \text{ si y solo si } (\mathbf{t_1}, \ldots, \mathbf{t_n}) \in \mathbf{R}$$

- Si $\alpha = \neg\beta$ entonces $v[\alpha] = 1$ si y solo si $v[\beta] = 0$.
- $v[\alpha \vee \beta] = 1$ si y solo si $v[\alpha] = 1$ o $v[\beta] = 1$ (o ambas valen 1).
- $v[\alpha \wedge \beta] = 1$ si y solo si $v[\alpha] = 1$ y $v[\beta] = 1$.
- $v[\alpha \to \beta] = 0$ si y solo si $v[\alpha] = 0$ o $v[\beta] = 1$ (o ambas cosas a la vez). Recordar que en el cálculo proposicional $\alpha \to \beta \equiv \neg\alpha \vee \beta$.
- Si $\alpha = \forall x(\beta)$ entonces $v[\alpha] = \sup_{a \in M}(v[\beta(x/a)])$. La definición tiene sentido, pues β tiene como única variable libre x.
- Si $\alpha = \exists x(\beta)$ entonces $v[\alpha] = \text{ínf}_{a \in M}(v[\beta(x/a)])$.

Observación. La sentencia $\forall x(\beta)$ es verdadera si y solo si todas las sentencias de la forma $\beta(x/a)$ son verdaderas. En definitiva, si «para todo» $a \in M$ la sentencia $\beta(x/a)$ es verdadera. La sentencia $\exists x(\beta)$ es verdadera «si existe algún» $a \in M$ que haga verdadera a la sentencia $\beta(x/a)$. Esto explica los nombres dados a los cuantificadores. □

Observación. No es difícil probar inductivamente la `propiedad de localidad` de la interpretación de fbf . Esto es, que el valor de verdad de α depende solo y exclusivamente de la interpretación de los símbolos que forman parte de α. Da lo mismo trabajar con el lenguaje L que con el $L(M)$ o con el lenguaje más restrigido que tiene como vocabulario los símbolos que entran en la sentencia. □

Hemos visto que la interpretación solamente asocia valores de verdad a fbf cerradas. Si las fórmulas son abiertas debemos cerrarlas de algún modo y entonces ya estamos en el caso anterior. Retomaremos nuestra vieja notación $\alpha[x_1, \ldots x_n]$ para indicar que α posee n variables libres. Si sustituimos x_i por $a_i \in M$ la notación habitual que hemos empleado es $\alpha[x_1/a_1, \ldots, x_n/a_n]$.

Relajaremos un poco la notación y escribiremos simplemente $\alpha[a_1, \ldots a_n]$, para indicar que sustituimos las variables libres.

Definición 11.4 *Dada una interpretación M y una fbf abierta $\alpha[x_1, \ldots, x_n]$ decimos que $(a_1, \ldots, a_n) \in M^n$* `satisface` *$\alpha$ si $v[\alpha[a_1, \ldots, a_n]] = 1$.*

Tomando una sentencia abierta y analizando que vectores la satisface, se pueden crear nuevas relaciones en M. Esto es lo que afirma la siguiente

Definición 11.5 *Dada una interpretación M de L, una relación $R \subset M^n$ es* `definible` *si existe una sentencia abierta α con n variables libres que cumple:*

$$(a_1, \ldots, a_n) \in R \text{ si y solo si } v[\alpha[a_1, \ldots, a_n]] = 1$$

La fórmula que sirve para definir la relación R no tiene necesariamente que ser única. Este concepto se debe interpretar de la siguiente manera: una formula abierta es una afirmación sobre un sistema que en principio no es ni cierta ni falsa. Al sustituir las variables por elementos de la interpretación se transforma ya en una sentencia y necesariamente es cierta o falsa. Los elementos de la relación son aquellos que hacen que la propiedad enunciada en la fórmula sea cierta.

En general no toda relación es definible, como prueba el siguiente argumento de cardinalidad. Imaginemos un lenguaje numerable. El conjunto de fórmulas es numerable. Si el conjunto M no es finito, el conjunto de sus relaciones siempre es innumerable. Necesariamente algunas de esas relaciones no pueden ser definibles.

En general a las fórmulas abiertas no se les puede asociar un valor de verdad, pues al hacer una sustitución de variables puede ser verdadera y al hacer otra puede ser falsa. Sin embargo si al realizar cualquier sustitución de variables resulta que la sentencia es siempre cierta (o siempre falsa) tiene sentido asociarle un valor de verdad.

Definición 11.6 *Dado M. Una fórmula abierta $\alpha[x_1, \ldots, x_n]$ es* `verdadera` *si $v[\alpha[a_1, \ldots, a_n]] = 1$ para cualquier vector $(a_1, \ldots, a_n) \in M^n$. Lo denotaremos por $v[\alpha] = 1$.*

Una fórmula abierta es verdadera (en una interpretación) si al realizar cualquier sustitución de sus variables libres obtenemos una fbf cerrada verdadera. Pero esto mismo se puede expresar de otro modo, utilizando el cuantificador universal, que también sirve para cerrar sentencias.

Proposición 11.6 *Dada una fbf abierta*

$$v[\alpha] = 1 \text{ } si \text{ } y \text{ } solo \text{ } si \text{ } v[\forall x_1 \ldots \forall x_n(\alpha[x_1, \ldots, x_n])] = 1$$

Observación. Para dar sentido a los términos y a las fbf con variables libres hemos hecho una extensión del lenguaje. De esta forma podemos sustituir variable libres por elementos de M. Sin embargo en otros libros no se procede de esta forma. Fijado un L-sistema M, se llama `asignación de variables` a toda función $s : \mathcal{V} \mapsto M$. Se cambian ahora el enfoque del problema y se define un nuevo concepto de interpretación. Una interpretación es un par (M, s) formado por un L-sistema M y una asignación de variables s. La asignación de variables se utiliza para sustituir las variables libres y de este modo, fijada una interpretación (M, s), se puede asociar un elemento de M a cualquier término, tenga o no tenga variables. Del mismo modo se puede asociar un valor de verdad a cualquier fbf, sea o no sea cerrada. Aunque el desarrollo de la semántica es ligeramente distinto con estas nuevas definiciones, los resultados en ambos enfoques son los mismos. □

12. Ejemplos semánticos

Antes de seguir conviene abordar un buen número de ejemplos, sacados todos ellos de la matemática habitual, para manejar con soltura la semántica de los lenguajes de primer orden.

Lenguaje de igualdad.

El lenguaje que no tiene vocabulario tiene como estructuras asociadas simples conjuntos.

- Sea la fórmula $\forall x(x = x)$. Esta fórmula es válida en cualquier interpretación, pues si sustituimos x por cualquier elemento $a \in M$ se obtiene la afirmación $a = a$ que es verdadera. Las fórmulas que son verdaderas en todas las interpretaciones son el análogo de las tautologías. Otra fórmula que es siempre válida es

  $$((x = y) \wedge (y = z) \rightarrow (x = z))$$

  Observemos que no hemos colocado los cuantificadores universales, pues es cierta para cualquier sustitución de las variables. Este es un abuso de notación bastante común en matemáticas.

- Veamos ahora la sentencia

  $$\varphi_3 = \exists x \exists y \exists z(x \neq y \wedge x \neq z \wedge y \neq z)$$

  Fijada una interpretación M, habla de la existencia de tres elementos que no sean iguales entre si. Si M tiene tres o más elementos esta sentencia es verdadera. En otro caso es falsa. No es difícil escribir (o al menos imaginar) una sentencia que afirme que el conjunto tiene n o más elementos.

- Estudiemos la sentencia

  $$\psi_2 = \forall x \forall y \forall z(x = y \vee x = z)$$

Afirma que dados tres elementos arbitrarios siempre dos, al menos, son iguales. Esto solamente lo cumplen los conjuntos que tienen 2 o menos elementos. Del mismo modo se puede estudiar la sentencia ψ_n que es cierta si y solamente si la interpretación tiene n o menos elementos.

- Finalmente la sentencia $\varphi_n \wedge \psi_n$ es cierta si ambas son ciertas. La primera afirma que el conjunto tiene n o más elementos y la segunda que tiene n o menos elementos. Esta sentencia solamente es cierta en las estructuras que posean n elementos.

Lenguaje de grupos.

- La sentencia (o fórmula pues la escribiremos sin cuantificadores)

$$(x + y) + z = x + (y + z)$$

es cierta si la operación es asociativa. Basta con la existencia de un trío de elementos de M que no la cumplan para que la fórmula sea falsa.

- Análogamente $x + y = y + x$ es cierta si la operación es conmutativa. La afirmación $x + 0 = x$ es cierta si la constante es el elemento neutro. La sentencia que habla del elemento neutro es

$$\forall x \exists y (x + y = 0)$$

Si hacemos la conjunción de las cuatro sentencia, la sentencia obtenida es cierta si y solo si M está dotado de una estructura de grupo conmutativo.

Lenguaje de las relaciones binarias.

- Denotamos por R la relación binaria y la situamos entre los términos. La sentencia xRx es cierta si la relación es reflexiva. No es difícil escribir las sentencias de las relaciones simétrica y transitiva.

- También es sencillo dar sentencias que nos permitan hablar sobre conjuntos ordenados. Ahora denotamos la relación por $\leq$. Algunas sentencias son:

$$x \leq x$$
$$(x \leq y) \wedge (y \leq z) \rightarrow (x = y)$$
$$(x \leq y) \wedge (y \leq z) \rightarrow (x \leq z)$$

13. Modelos

Es habitual empezar a cambiar notaciones y llamar `modelo` a lo que antes llamabamos interpretación, sobre todo si estudiamos la semántica con ayuda de los sistemas. También escribimos $M \vDash \alpha$ en lugar de $v_M[\alpha] = 1$. Con estas nuevas notaciones tenemos:

$M \vDash (t_1 = t_2)$	si y solo si $\mathbf{t_1} = \mathbf{t_2}$
$M \vDash Rt_1 \dots t_n$	si y solo si $(\mathbf{t_1}, \dots, \mathbf{t_n}) \in \mathbf{R}$
$M \vDash \neg(\alpha)$	si y solo si no es cierto que $M \vDash \alpha$
$M \vDash (\alpha \wedge \beta)$	si y solo si $M \vDash \alpha$ y $M \vDash \beta$
$M \vDash (\alpha \vee \beta)$	si y solo si $M \vDash \alpha$ o $M \vDash \beta$
$M \vDash (\alpha \to \beta$	si y solo si $M \vDash \neg(\alpha) \vee \beta$
$M \vDash \forall x(\alpha)$	si y solo si para todo $a \in M$ se tiene $M \vDash \alpha[a]$
$M \vDash \exists x(\alpha)$	si y solo si si existe un $a \in M$ se tiene $M \vDash \alpha[a]$

Definición 13.1 *Una sentencia α es* `satisfacible` *si existe un modelo M tal que $M \vDash \alpha$. Si Σ es un conjunto de sentencias (finito o infinito) decimos que es satisfacible si existe un modelo M tal que $M \vDash \alpha$ para cualquier elemento $\alpha \in \Sigma$. Cometeremos el abuso de notación de escribir $M \vDash \Sigma$ en este último caso. Si las fórmulas son abiertas se emplea el cuantificador universal para cerrarlas.*

Si $M \vDash \alpha$ también decimos que M es un `modelo` de α y lo mismo para conjuntos de sentencias.

Ejemplos.

- Sea L el lenguaje de los grupos. Consideremos el conjunto de tres sentencias:

$$\Sigma_{gr} = \begin{cases} \forall x \forall y \forall z((x + y) + z = x + (y + z)) \\ \forall x(x + e = x) \\ \forall x \exists y(x + y = e) \end{cases}$$

Las estructuras M que cumplen estas tres sentencias son precisamente los grupos, pues en estas tres sentencias está recogida la definición de grupo.

- Sea L el lenguaje con una única relación binaria R. Si un modelo M cumple la sentencia $\forall x(Rxx)$ decimos que la relación es reflexiva. Los conjuntos dotados de una relación de equivalencia son los modelos del siguiente conjunto de sentencias:

$$\begin{cases} \forall x(Rxx) \\ \forall x \forall y(Rxy \to Ryx) \\ \forall x \forall y \forall z((Rxy \wedge Ryz) \to Rxz) \end{cases}$$

Vemos la gran analogía entre este concepto y el correspondiente del cálculo proposicional. Lo que allí era una valuación aquí es un modelo. Prácticamente todas las definiciones y conceptos asociados se «calcan» del caso proposicional.

Definición 13.2 *Dos sentencias α y β son* `equivalentes` *(*`semánticamente equivalentes`*) si para todo modelo M se cumple*

$$M \vDash \alpha \text{ si y solo si } M \vDash \beta$$

Lo denotaremos por $\alpha \equiv \beta$.

Esto quiere decir que independientemente de la interpretación ambas son verdaderas o falsas a la vez. A nivel semántico dos sentencias equivalentes son exactamente iguales. Son dos maneras distintas de expresar lo mismo. En semántica se puede trabajar con el conjunto cociente de **Form** módulo esta relación de equivalencia. En particular tenemos las siguientes equivalencias:

- $\alpha \wedge \beta \equiv \neg(\alpha \to \neg\beta)$.
- $\alpha \vee \beta \equiv \neg\alpha \to \beta$.
- $\exists x(\alpha) \equiv \neg(\forall x(\neg\alpha))$

En el conjunto cociente todas las proposiciones se pueden crear a partir de los símbolos lógicos $\{\neg, \rightarrow, \forall\}$. Estos forman el análogo a un conjunto completo de conectivas. Por ello podríamos haber hecho todo el desarrollo de los lenguajes de primer orden utilizando únicamente estos tres símbolos lógicos. Naturalmente existen otros conjuntos completos de signos lógicos.

Definición 13.3 *Dado un conjunto Σ de sentencias y α una sentencia, decimos que α es* `consecuencia lógica` *de Σ si todo modelo que cumpla $M \vDash \Sigma$ cumple también $M \vDash \alpha$. Lo denotamos por $\Sigma \vDash \alpha$.*

Vemos que hemos empleado el mismo símbolo ($\vDash$) para la noción de satisfación y la de consecuencia. No puede existir confusión pues en un caso a la izquierda figura un modelo y el otro caso un conjunto de sentencias.

Si α no es consecuencia de un conjunto Σ lo denotaremos $M \nvDash \alpha$. Esto implica que existe al menos un modelo M que cumple $M \vDash \Sigma$ y que sin embargo no cumple $M \vDash \alpha$.

Ejemplos.

- En el lenguaje de la igualdad tenemos la sentencia

$$\alpha_2 = \exists x \exists y (x \neq y)$$

  Los modelos de esta sentencia deben de tener al menos dos elementos. De manera análoga se pueden definir la sentencia α_n cuya interpretación es cierta si el modelo tiene n o más elementos. Consideremos el conjunto de sentencias $\Sigma = \{\alpha_2, \alpha_3, dots\}$. Los modelos de este conjunto son los conjuntos infinitos. Como el lenguaje de la igualdad está contenido en todo lenguaje, este resultado es válido en cualquier lenguaje.

- Consideremos el conjunto Σ_{gr}. Entonces $M_{gr} \Vdash \alpha$ siendo

$$\alpha = \forall x \forall y \forall z ((x + y) = (x + z) \rightarrow (y = z))$$

  pues todo aquel que haya estudiado teoría de grupos ha demostrado que esta última sentencia es verdadera en todo grupo (es la llamada

ley de cancelación). Sin embargo $\Sigma_{gr} \nvDash \beta$ siendo

$$\beta = \forall x \forall y (x + y = y + x)$$

pues es conocido que existen grupos no abelianos. Si G es un grupo no abeliano entonces $G \vDash \Sigma_{gr}$ y sin embargo $G \nvDash \beta$.

- Si una sentencia α es satisfacible se escribe $\mathsf{Sat}\ \alpha$.
- Las consecuencias lógicas del conjunto vacío son las sentencias que son válidas en cualquier modelo. Es el análogo de la tautologías:

  > Una sentencia α es `universalmente válida` o es una `verdad lógica` si para todo modelo M se tiene $M \vDash \alpha$.

Los siguientes resultados se prueban igual que en el cálculo proposicional.

Corolario 13.1 *Un conjunto finito $\Sigma = \{\alpha_1, \ldots, \alpha_n\}$ es satisfacible si y solo es satisfacible la sentencia $\alpha_1 \wedge \cdots \wedge \alpha_n$.*

Corolario 13.2 *Si Σ no es satisfacible entonces $\Sigma \vDash \alpha$ para cualquier sentencia.*

Corolario 13.3 *$\alpha \equiv \beta$ si y solo si $\alpha \vDash \beta$ y $\beta \vDash \alpha$.*

Corolario 13.4 *$\alpha \equiv \beta$ si y solo si $\alpha \to \beta$ y $\beta \to \alpha$ son universalmente válidos.*

Corolario 13.5 *Sea Σ un conjunto de sentencias. Para toda sentencia α*

$$\Sigma \vDash \alpha \text{ si y solo si } \Sigma \cup \{\neg\alpha\} \text{ es insatisfacible}$$

En la teoría de sistemas, vista desde el punto de vista del álgebra universal, se tenía un concepto de isomorfismo. A todos los efectos dos sistemas isomorfos pueden ser considerados iguales. Pero ahora utilizamos los sistemas como modelos para estudiar la semántica. A efectos semánticos dos modelos que tengan las mismas sentencias verdaderas pueden ser considerados iguales.

Definición 13.4 *Dado un lenguaje L, decimos que dos modelos M y N son* `elementalmente equivalentes` *si*

$$M \vDash \alpha \text{ si y solo si } N \vDash \alpha$$

Es claro que esta noción es una relación de equivalencia en el conjunto de sistemas. No es difícil intuir que dos sistemas isomorfos son elementalmente equivalentes. Pero lo sorprendente es que el recíproco no es cierto: dado un lenguaje L, pueden existir modelos que sean elementalmente equivalentes y que sin embargo no sean isomorfos. Estos son los llamados `modelos no estandard`. La situación habitual es la siguiente. Se tiene una estructura matemática suficientemente conocida, por ejemplo, la aritmética elemental. Se crea un lenguaje asociado a esa estructura y se extraen todas las sentencias verdaderas de esa estructura. Pero resulta que existen otros modelos, no standard, de la aritmética que cumplen exactamente las mismas sentencias.

14. Teorías

En el conjunto de sentencias de un lenguaje tenemos definida la noción de consecuencia. Dada cualquier colección de sentencias se puede ampliar dicha colección añadiendole todas las posibles consecuencias. Obtenemos de esta forma, en general, un nuevo subconjunto que contiene al anterior. En principio podemos repetir el proceso con el nuevo subconjunto, sin embargo veremos que una nueva repetición no amplía el conjunto.

En toda esta sección trabajaremos con un lenguaje de primer orden fijo. Los modelos que aparezcan tendrán la misma signatura que el lenguaje.

Definición 14.1 *Un subconjunto* $T \subset \mathbf{Sent}$ *es una* `teoría` *si para todo* $\alpha \in \mathbf{Sent}$ *tal que* $T \vDash \alpha$ *se tiene* $\alpha \in T$*.*

Observación. En virtud del teorema de completitud de la lógica de primer orden sabemos que son equivalentes la noción de consecuencia $\Sigma \vDash \alpha$ y la de noción de deducción $M \vdash \alpha$. Se puede utilizar una u otra para definir el concepto de teoría. □

Las teorías son subconjuntos cerrados por la noción de consecuencia. Dado un subconjunto arbitrario de sentencias siempre se puede construir de un modo natural una teoría asociada. Recordemos que llamamos consecuencias de un conjunto Σ a

$$\mathbf{Con}(\Sigma) = \{\alpha \in \mathbf{Sent} \text{ tales que } \Sigma \vDash \alpha\}$$

Proposición 14.1 *Si* $\Sigma \subset \mathbf{Sent}$*, entonces* $\mathbf{Con}(\Sigma)$ *es una teoría.*

Demostración.

Dada una sentencia α, si $\mathbf{Con}(\Sigma) \vDash \alpha$ necesariamente, por la definición de consecuencia, $\Sigma \vDash \alpha$ y $\alpha \in \mathbf{Con}(\Sigma)$. □

Es evidente que dado cualquier conjunto Σ se tiene que $\Sigma \subset \mathbf{Con}(\Sigma)$. Además si T es una teoría se tiene la inclusión contraria. El siguiente corolario se puede tomar también como una nueva definición del concepto de teoría.

Corolario 14.2 *$T \subset \mathbf{Sent}$ es un teoría si y solo si $T = \mathbf{Con}(T)$.*

Dada cualquier teoría T, todo conjunto Σ tal que $\mathbf{Con}(\Sigma) = T$ se dice que `genera` la teoría. Naturalmente dada una teoría pueden existir muchos subconjuntos distintos que sean generadores. Los elementos del conjunto Σ se llaman `axiomas` de la teoría. Si variamos el conjunto generador variamos los axiomas.

Ejemplos.

- Sea $T = \mathbf{Sent}$. Entonces T es una teoría. Cualquier teoría está incluida en T. Dada una sentencia arbitraria α tenemos que el conjunto $\Sigma = \{\alpha, \neg\alpha\}$ genera el conjunto, puesto que ya hemos visto que de una contradicción podemos extraer cualquier consecuencia. En general, si Σ no es satisfacible (esto es, si carece de modelos), entonces genera el conjunto de todas las sentencias.

- $\mathbf{Con}(\varnothing)$ (el conjunto de verdades lógicas) es un teoría. Cualquier teoría T contiene a esta teoría. En efecto, dada una teoría T, como $\varnothing \subset T$ tenemos que $\mathbf{Con}(\varnothing) \subset \mathbf{Con}(T) = T$. Como el conjunto de verdades lógicas es infinito, cualquier teoría tiene cardinal infinito.

- La intersección de teorías es una teoría. Si $\{T_i\}_{i\in I}$ es una colección de teorías, sea $T = \bigcup_{i\in I} T_i$. Supongamos que $T \vDash \alpha$. Como $T \subset T_i$ tenemos que $T_i \vDash \alpha$. Como T_i es teoría, entonces $\alpha \in T_i$. Al ser cierto para todo índice, α pertenece a la intersección.

- Sea L el lenguaje de la teoría de grupos. Consideremos el conjunto

$$\Sigma = \begin{cases} \forall x \forall y \forall z((x + y) + z = x + (y + z)) \\ \forall x(x + e = x) \\ \forall x \exists y(x + y = e) \end{cases}$$

  Sea $\alpha \in \mathbf{Con}(\Sigma)$, entonces todo modelo M tal que $M \vDash \Sigma$ cumple $M \vDash \alpha$. Por lo tanto todas las sentencias de la teoría son ciertas en

todos los grupos. A la inversa, si α es una sentencia que es verdadera en todos los grupos, entonces pertenece a la teoría. Los axiomas de esta teoría constituyen hoy en día la definición natural de grupo. Ello no implica que no existan otros axiomas que generan la misma teoría. Denotaremos esta teoría como $\mathbf{Th}(Grupos)$.

- El ejemplo anterior se generaliza a muchas otras teorías que se estudian en matemáticas. Según la escuela formalista este es el prototipo de las teorías matemáticas: se parte de un lenguaje, se toma un conjunto (finito o infinito) de axiomas y se deducen de estos axiomas todas las posibles consecuencias.

El método anterior de generar teorías es, posiblemente, uno de los mejores, adaptado sobre todo a la mentalidad formalista. Sin embargo otros matemáticos prefieren ver el asunto desde otro punto de vista. Se imaginan que existen unas estructuras, por ejemplo el conjunto de los números naturales. Entonces crean un lenguaje adaptado a esta estructura y estudian todas las sentencias que son verdad en este modelo. Denotaremos por $\mathcal{M}$ al conjunto de todas las estructuras del tipo del lenguaje del que hablamos.

Definición 14.2 *Dado un conjunto* $\Sigma \subset \mathbf{Sent}$ *llamamos* `modelos de` Σ *al conjunto*

$$\mathbf{Mod}(\Sigma) = \{A \in \mathcal{M} \textit{ tales que } A \vDash \Sigma\}$$

Si el conjunto es satisfacible, entonces tiene al menos un modelo y este conjunto no es vacío. Los conjuntos insatisfacibles carecen de modelos, lo que hace poco interesantes.

Por la definición de consecuencia lógica tenemos

$$\mathbf{Mod}(\Sigma) = \mathbf{Mod}(\mathbf{Con}(\Sigma))$$

por lo que basta estudiar los modelos de las teorías.

Observación. El teorema de Löwenheim-Skolem afirma que si Σ es consistente y el lenguaje es numerable, entonces Σ tiene un modelo A de cardinal

numerable. Existen muchas generalizaciones de este teorema, fundamentalmente debidas a Tarski, que afirman la existencia de modelos con ciertos cardinales. □

Definición 14.3 *Sea L un lenguaje y A una estructura adaptada a este lenguaje. Llamamos* `teoría de` *A al conjunto*

$$\mathbf{Th}(A) = \{\alpha \in \mathbf{Sent} \textit{ tales que } A \vDash \alpha\}$$

Casi por definición se puede comprobar que efectivamente $\mathbf{Th}(A)$ es una teoría. Los elementos de la teoría son las sentencias verdaderas en el modelo particular que hemos elegido. Del mismo modo se puede partir de una colección $K \subset \mathcal{M}$ de modelos y considerar las sentencias que son válidas en todos los modelos

$$\mathbf{Th}(K) = \{\alpha \text{ tales que } A \vDash \alpha \text{ para todo modelo } A \in K\}$$

Ejemplos.

- Si $K_1 \subset K_2$ entonces $\mathbf{Th}(K_2) \subset \mathbf{Th}(K_1)$. También se cumple la propiedad $\mathbf{Th}(K_1 \cup K_2) = \mathbf{Th}(K_1) \cap \mathbf{Th}(K_2)$. La formación de teorías invierte el orden de las inclusiones. Si deseamos que la teoría abarque más estructuras el precio a pagar es reducir el conjunto de sentencias que son válidas en todas las estructuras.

- Sea K la clase de todos los grupos. La teoría que generan coincide con la teoría que generan los axiomas de grupo. Por ello antes hemos denotado por $\mathbf{Th}(Grupos)$ a dicha teoría. El mismo esquema es válido para muchas otras estructuras algebraicas.

- Consideremos el conjunto de los números naturales con sus dos operaciones y su relación de orden. La teoría que genera esta estructura es lo que comunmente se ha llamado `aritmética`. Se suele denotar por $\mathbf{Th}(\mathbb{N})$.

- Sean A y B dos estructuras isomorfas. Entonces ambas generan la misma teoría. Esta restricción se puede relajar un poco y pedir que únicamente sean elementalmente equivalentes. Es más, se tiene la siguiente caracterización:

  > A es elementalmente equivalente a B si y solo si $\mathbf{Th}(A) = \mathbf{Th}(B)$

Una vez planteados estos dos procesos de construcción de teorías surgen dos preguntas básicas:

- Dados unos axiomas Σ, generamos la teoría $T = \mathbf{Con}(\Sigma)$. ¿Existe una colección de estructuras $K \subset \mathcal{M}$ tal que $T = \mathbf{Th}(K)$?
- Dada una colección de estructuras $K \subset \mathcal{M}$ creamos la teoría $T = \mathbf{Th}(K)$. Esta teoría tiene axiomas puesto que $T = \mathbf{Con}(T)$. ¿Pero existe un conjunto finito de axiomas?

Recordemos que un conjunto $\Delta \subset \mathbf{Sent}$ es consistente si tiene un modelo y que es completo si para toda sentencia α o bien α o bien $\neg\alpha$ pertenecen al conjunto. Con estos resultados en mente, tenemos el siguiente resultado.

Corolario 14.3 *Una teoría T es consistente si y solo si $T \neq \mathbf{Sent}$. La única teoría inconsistente es* **Sent**.

Dada cualquier colección de modelos K, la teoría generada siempre es consistente, puesto que sabemos que es imposible que $A \vDash \alpha$ y a la vez $A \vDash \neg\alpha$. La teoría generada no puede contener a todas las sentencias.

A partir de ahora consideramos el conjunto de todas las teorías consistentes de un cierto lenguaje. La única teoría que debemos quitar es el total, según se deduce del corolario anterior. Este conjunto está ordenado por inclusión y hemos visto que tiene un elemento mínimo, que es $\mathbf{Con}(\varnothing)$. Los elementos maximales, que siempre existen en virtud del lema de Zorn, tienen unas características especiales.

Proposición 14.4 $\Delta \subset \mathbf{Sent}$ *es consistente maximal si y solo si es una teoría completa (y consistente).*

Demostración.

Supongamos que Δ es maximal y sea $\Delta \vDash \alpha$. Si α no pertenece al conjunto entonces $\Delta \cup \{\alpha\}$ sería consistente, contradicciendo la maximalidad. Concluimos que Δ es una teoría.

Tomemos una sentencia arbitraria α y construimos el conjunto $\Delta \cup \{\alpha\}$. Pueden darse dos casos:

- Si $\Delta \cup \{\alpha\} = \Delta$ entonces $\alpha \in \Delta$.
- Si $\Delta \cup \{\alpha\}$ es estrictamente mayor que Δ, por la maximalidad, debe ser inconsistente. Se concluye en este caso que $\neg\alpha \in \Delta$

Hemos visto que necesariamente la teoría es completa.

Recíprocamente, si Δ es una teoría completa (y consistente), necesariamente es maximal. Supongamos que $\alpha \notin \Delta$. Por ser teoría no es posible que $\Delta \vDash \alpha$. Como es completa, tenemos que $\Delta \vDash \neg\alpha$. Entonces $\Delta \cup \{\alpha\} \vDash \alpha \wedge \neg\alpha$ y el conjunto $\Delta \cup \{\alpha\}$ no es consistente. $\square$

Ejemplos.

- Sea A una estructura. Entonces $\mathbf{Th}(A)$ es siempre consistente, puesto que es imposible que $A \vDash \alpha$ y que también se tenga $A \vDash \neg\alpha$. Además esta teoría es completa, pues si $A \nvDash \alpha$ entonces, por la definición de satisfación, se tiene que $A \vdash \neg\alpha$.
- La teoría de grupos no es completa. La sentencia

  $$\alpha = \forall x \forall y (x + y = y + x)$$

  es cierta en algunos modelos (los grupos conmutativos) y falsa en otros. Por lo tanto no puede pertenecer a la teoría de grupos. Lo mismo le sucede a su negación. Ni α ni $\neg\alpha$ pertenecen a la teoría.

- El famoso teorema de incompletitud de Gödel afirma que $\mathbf{Con}(\Delta)$ es una teoría incompleta, siendo Δ el conjunto de axiomas de Peano. La demostración de este resultado sobrepasa lo pretendido en estos apuntes.

Corolario 14.5 *Sean A y B estructuras tales que $\mathbf{Th}(A) \subset \mathbf{Th}(B)$. Entonces se tiene la igualdad.*

Demostración.

Como $\mathbf{Th}(A)$ es consistente y completa, entonces es maximal. Como la teoría de B es consistente, se tiene la igualdad. □

Corolario 14.6 *Sea K una colección de sistemas. La teoría $\mathbf{Th}(K)$ es completa si y solo si para cada par de estructuras $A, B \in K$ son elementalmente equivalentes.*

Demostración.

Si todas los elementos de K son elementalmente equivalentes, entonces se tiene que $\mathbf{Th}(K) = \mathbf{Th}(A)$ siendo A un elemento arbitrario. Ya hemos visto que este tipo de teorías son completas.

Si A es un miembro de K se tiene la inclusión $\mathbf{Th}(K) \subset \mathbf{Th}(A)$. Si la teoría de K es completa se tiene la igualdad. Para todo miembro B de K se tiene el mismo resultado. Esto prueba que $\mathbf{Th}(A) = \mathbf{Th}(B)$ y esto es lo mismo que decir que las dos estructuras son elementalmente equivalentes. □

En una teoría T es completa y consistente todos sus modelos son elementalmente equivalentes. Sin embargo ello no implica que todos los modelos sean isomorfos. Ello requiere una nueva

Definición 14.4 *Una teoría es* `categórica` *si cada si todos sus modelos son isomorfos.*

Necesariamente una teoría categórica es completa, pero no existe una caracterización tan simple de las teorías categóricas. Una de las mayores sorpresas de la teoría de modelos radica en la existencia de teorías de la forma $\mathbf{Th}(A)$ que no son categóricas. Demos algunos ejemplos, aunque omitimos las demostraciones.

Ejemplos.

- La aritmética $\mathbf{Th}(\mathbb{N})$ no es categórica. Existen modelos, numerables A que dan lugar a la misma teoría y que no son isomorfos. Sonlos llamados modelos no estandard de la aritmética.
- Análogamente $\mathbf{Th}(\mathbb{R})$ no es categórica. Existen modelos, con el mismo cardinal que los reales y no isomorfa.

15. Cálculo deductivo

En esta sección pretendemos formalizar el concepto de demostración matemática. Dado un conjunto de sentencias Σ definiremos cuando una sentencia arbitraria β se puede deducir a partir de Σ. Para ello crearemos una serie de reglas. La sentencia β se puede deducir a partir de Σ si podemos llegar a β partiendo de las sentencias de Σ y aplicando únicamente las reglas.

Definición 15.1 *Una deducción es una sucesión finita de lineas, donde cada línea es una sucesión finita de sentencias.*

Si denotamos por $\alpha_1 \dots \alpha_n \beta$ a una de estas líneas, diremos que $\alpha_1 \dots \alpha_n$ son las `premisas` y que β es la `conclusión`. Normalmente denotaremos por $\Gamma = \alpha_1 \dots \alpha_n$ y escribiremos la línea en la forma $\Gamma\beta$.

Las `reglas del cálculo` permiten construir nuevas líneas a partir de otras ya creadas. La colocación de las líneas a las que se va a aplicar la regla es indiferente.

Para enunciar las reglas situaremos sobre la recta las líneas que deben existir previamente y debajo de la recta la nuva línea que nos permite crear la regla en cuestión. Hay una regla que nos permite crear una línea a pesar de que no exista ninguna línea creada. Otras necesitan una única línea para crear una nueva y finalmente otras reglas necesitan dos líneas ya creadas para dar origen a otra línea.

- (IH) Regla de introducción de hipótesis

$$\frac{}{\Gamma \quad \beta} \quad \text{si } \beta \in \Gamma$$

- (M) Regla de monotonía

$$\frac{\Gamma \quad \beta}{\Gamma' \quad \beta} \quad \text{si } \Gamma \subset \Gamma'$$

- (PC) Regla de la prueba por casos

$$\dfrac{\Gamma \quad \beta \quad \gamma \qquad \Gamma \quad \neg\beta \quad \gamma}{\Gamma \quad \gamma}$$

- (NC) Regla de no contradicción

$$\dfrac{\Gamma \quad \neg\alpha \quad \beta \qquad \Gamma \quad \neg\alpha \quad \neg\beta}{\Gamma \quad \alpha}$$

- (IDA) Regla de introducción del disyuntor en el antecedente

$$\dfrac{\Gamma \quad \alpha \quad \gamma \qquad \Gamma \quad \beta \quad \gamma}{\Gamma \quad \alpha \vee \beta \quad \gamma}$$

- (IDC) Reglas de introducción del disyuntor en la conclusión

$$\dfrac{\Gamma \quad \alpha}{\Gamma \quad \alpha \vee \beta} \qquad \dfrac{\Gamma \quad \alpha}{\Gamma \quad \beta \vee \alpha}$$

16. Consistencia

Definición 16.1 *Un subconjunto* $\Sigma \subset$ **Sent** *es* `consistente` *(sintácticamente) si no existe ninguna fórmula* α *tal que* $\Sigma \vdash \alpha$ *y* $\Sigma \vdash \neg\alpha$.

Un subconjunto es `inconsistente` *o* `contradictorio` *si existe una fórmula* α *tal que* $\Sigma \vdash \alpha$ *y* $\Sigma \vdash \neg\alpha$.

Proposición 16.1 *Un subconjunto es inconsistente si y solo si* $\Sigma \vdash \beta$ *para cualquier sentencia* β.

Demostración.

Si es contradictorio, entonces existe una sentencia tal que $\Sigma \vdash \alpha$ y también $\Sigma \vdash \neg\alpha$. Esto es, existe un subconjunto finito $\Gamma_1 \subset \Sigma$ tal que $\Gamma_1 \vdash \alpha$. Del mismo modo existe un subconjunto finito Γ_2 tal que $\Gamma_2 \vdash \neg\alpha$.

Partimos de las premisas

$$\begin{array}{ll} \Gamma_1 & \alpha \\ \Gamma_2 & \neg\alpha \end{array}$$

Añadimos premisas y reordenamos

$$\begin{array}{lll} \Gamma_1 & \Gamma_2 & \alpha \\ \Gamma_2 & \Gamma_1 & \neg\alpha \end{array}$$

Luego de $\Gamma_1 \cup \Gamma_2$ se puede derivar cualquier fórmula β. Como $\Gamma_1 \cup \Gamma_2$ es finito y está contenido en Σ, esto implica que $\Sigma \vdash \beta$. $\square$

Corolario 16.2 *Un conjunto* Σ *es consistent si existe una fórmula que no se deduce de* Σ.

Lema 16.3 *Un conjunto* Σ *es consistente si y solo si son consistentes todos sus subconjuntos finitos.*

Lema 16.4 *Si* Σ *tiene un modelo, entonces es consistente.*

Demostración.

Si fuese inconsistente entonces existiría α tal que $\Sigma \vdash \alpha$ y $\Sigma \vdash \neg\alpha$. Por el teorema de corrección $\Sigma \vDash \alpha$ y $\Sigma \vDash \neg\alpha$. Pero esto es imposible puesto que tiene un modelo. $\square$

Lema 16.5 *Se cumplen las siguientes propiedades:*

- $\Sigma \vdash \alpha$ *si y solo si* $\Sigma \cup \{\neg\alpha\}$ *es contradictorio.*
- $\Sigma \vdash \neg\alpha$ *si y solo si* $\Sigma \cup \{\alpha\}$ *es contradictorio.*
- *Si* Σ *es consistente, entonces o bien* $\Sigma \cup \{\alpha\}$ *o bien* $\Sigma \cup \{\neg\alpha\}$ *es consistente.*

17. Teorema de completitud

Dado un conjunto de sentencias Σ tenemos una noción semántica de deducción, $\Sigma \vDash \alpha$, y una noción sintáctica, asociada al cálculo secuencial, $\Sigma \vdash \alpha$. El teorema de corrección afirma que la noción semántica engloba a la sintáctica:

$$\text{si } \Sigma \vdash \alpha \text{ entonces } \Sigma \vDash \alpha$$

El teorema de completitud pretende probar el recíproco de este resultado. Veremos que probar este resultado es equivalente a encontrar un modelo para cada conjunto consistente de fórmulas.

Proposición 17.1 *Si cada conjunto consistente tiene un modelo (es consistente) entonces el cálculo secuencial es completo.*

Demostración.

Supongamos que $\Sigma \vDash \alpha$ y sin embargo no sea cierto que $\Sigma \vdash \alpha$. Entonces el conjunto $\Sigma \cup \{\neg\alpha\}$ es consistente (si no lo fuese, entonces Σ tampoco lo sería), pero no puede tener un modelo, puesto que es imposible que un modelo satisfaga una sentencia y su negación. $\square$

Consideremos la familia de todos los conjuntos consistentes de un lenguaje determinado. Respecto a la inclusión conjuntista, tenemos la noción de elemento maximal. El lema de Zorn afirma la existencia de conjuntos consistentes y maximales. Es más, todo conjunto Σ está contenido en un conjunto consistente maximal.

Definición 17.1 *Un conjunto Σ es* `consistente maximal` *si no existe ningún conjunto consistente que lo contenga estrictamente.*

A partir de ahora cometeremos el abuso de notación de hablar de conjuntos maximales, sobreentendiendo el calificativo de consistente.

El siguiente resultado es otra posible definición de conjunto consistente maximal.

Corolario 17.2 *Un conjunto Σ es consistente maximal si y solo si para cada $\alpha \notin \Sigma$ se tiene que el conjunto $\Sigma \cup \{\alpha\}$ es contradictorio.*

Demostración.

En efecto, $\Sigma \cup \{\alpha\}$ contiene estrictamente al conjunto Σ. Como este es maximal, necesariamente el conjunto que lo contiene no puede ser consistente. Entonces $\Sigma \cup \{\alpha\}$ es contradictorio.

Recíprocamente. Sea Σ un conjunto consistente, tal que al añadirle una sentencia se transforme en contradictorio. Supongamos que existe un conjunto Δ tal que $\Sigma \subset \Delta$ de modo estricto. Entonces debe existir una sentencia α que pertenezca a Δ y que sin embargo no esté en Σ. Pero entonces $\Sigma \cup \{\alpha\} \subset \Delta$ es contradictorio, lo que implica que también es contradictorio Δ. Todos los conjuntos que contienen a Σ son contradictorios. Luego Σ es consistente y maximal. $\square$

El siguiente resultado es otra nueva caracterización de los conjuntos consistentes maximales. Es muy usada en razonamientos de todo tipo.

Proposición 17.3 *Si Σ es consistente maximal, entonces para toda sentencia α se tiene que o bien $\Sigma \vdash \alpha$ o bien $\Sigma \vdash \neg\alpha$.*

Demostración.

Supongamos que no es cierto que $\Sigma \vdash \alpha$. Entonces, por la maximalidad, tenemos terminar. $\square$

Definición 17.2 *Un conjunto Σ es* `ejemplificado` *si para cada sentencia de la forma $\exists x(\alpha) \in \Sigma$ existe un término t (no necesariamente único) tal que $\alpha[t/x] \in \Sigma$.*

A nivel sintáctico esta definición parece un poco extraña. Sin embargo a nivel semántico es más clara: en un modelo M, $\exists x(\alpha)$ se interpreta como la existencia de un elemento $m \in M$ que hace verdadera a la sentencia α. Esto es, que existe un elemento m, tal que $\alpha[m]$ es cierta. A nivel sintáctico lo más que podemos afirmar es la existencia de un término que verifique lo mismo.

A partir de ahora consideraremos un conjunto de sentencias consistente maximal y ejemplificado. Veremos que a partir de estos conjuntos se puede construir, de un modo efectivo, un modelo. El procedimiento a seguir que utilizaremos se debe a Leon Henkin. Para construir modelos de otros conjuntos consistente simplementente tendremos que incluirlos en conjuntos ejemplificados y maximales.

Teorema 17.4 (Henkin) *Si Σ es maximal y ejemplificado, entonces posee un modelo.*

Demostración.

Debemos construir la estructura del modelo. Primeramente construiremos el conjunto y después las funciones y las relaciones.

En el conjunto de términos del lenguaje consideramos la siguiente relación de equivalencia

$$t_1 \sim t_2 \text{ si y solo si } \Sigma \vdash t_1 = t_2$$

El cálculo secuencial nos informa de que efectivamente esta es una relación de equivalencia. La clase de equivalencia de t la denotaremos por $\bar{t}$. Llamemos M al conjunto cociente. Tomemos ahora una functor f de orden n. Construimos la función

$$\begin{array}{rccc} f_M: & M \times \cdots \times M & \mapsto & M \\ & (\bar{t}_1, \ldots, \bar{t}_n) & \mapsto & \overline{ft_1 \ldots t_n} \end{array}$$

Como hemos definido la función a partir de un representante de la clase de equivalencia, debemos mostrar que la definición es independiente del representante elegido. De nuevo el cálculo secuencial nos garantiza la corrección del procedimiento.

18. Bibliografía

[1] J. Sancho San Román. *Lógica Matemática y Computabilidad*, Díaz de Santos, 1990

[2] M. Manzano & A. Huertas. *Lógica para principiantes*, Alianza Editorial, 2004

[3] H.D. Ebbinghaus & J. Flum & W. Thomas *Mathematical Logic*, Springer, 1984

[4] C. Ivorra. *Lógica*, Internet.

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