Cálculo Proposicional

1. Breve historia de la lógica

En un principio la lógica no tuvo el sentido de estructura formal estricta que tiene ahora. La definición de lógica como ciencia formal en la cultura occidental es el resultado de un largo desarrollo histórico que empieza con las obras de algunos filósofos griegos y llega hasta la actualidad. Históricamente las áreas de aplicación más importantes de la lógica son la filosofía, las matemáticas y la informática. A continuación se presenta un pequeño resumen de los principales pasos que han llevado a la formulación de la lógica formal.

1.1. Lógica y filosofía

En el siglo cuarto antes de nuestra era *Aristóteles* fue el primero en tratar de formalizar el razonamiento humano para poder discernir en la discusiones filosóficas. Aristóteles se puede considerar el fundador de la denominada ***lógica clásica.*** Durante la Edad Media el proceso de sistematización de la lógica fue desarrollado por los filósofos árabes y los escolásticos. En el siglo XIII *Santo Tomás de Aquino* empleó la lógica en el contexto de las discusiones teológicas. Filósofos racionalistas como R. Descartes, B. Pascal y G. Leibniz aportaron a través del desarrollo de la matemática temas que van a marcar una importante evolución en la lógica. *Leibniz*, en el siglo XVII, fue el primero en formular la lógica como base del razonamiento matemático, pero sus estudios fueron abandonados hasta el siglo XIX, cuando finalmente se fundó la lógica matemática como ciencia.

1.2. Lógica y matemáticas

A mediados del siglo XIX, De Morgan publicó *Lógica formal*, donde introdujo las famosas leyes que llevan su nombre e intentó generalizar la noción de silogismo. En el 1854, el inglés G. Boole publicó el libro *Las leyes del pensamiento*. Influenciado por las teorías de los matemáticos De Morgan y Hamilton, Boole definió la lógica como sistema formal dirigido a un ámbito más amplio que sólo al estudio del lenguaje natural. Su obra proporciona un modelo algebraico de la lógica de proposiciones.

En el 1879 el alemán G. Frege publicó el libro *Fundamentos de Aritmética: Conceptualmente derivada*, en el cual se formaliza la lógica de predicados. También desarrolló la idea de un *lenguaje formal* y definió la noción de *prueba*. Estas ideas constituyeron una base teórica fundamental para el desarrollo de las computadoras y las ciencias de la computación.

A principios del siglo XX B. Russell y Whitehead, inspirados por el trabajo del matemático italiano G. Peano, publicaron los tres volúmenes de *Principia Mathematica*, un trabajo monumental en el que lograron plasmar gran parte de la matemática a partir de la lógica, evitando caer en paradojas. En 1920

Hilbert propuso el problema de la axiomatización completa de las matemáticas, pero en 1931 Gödel demostró el famoso *teorema de incompletitud* del enfoque axiomático, que contesta negativamente al problema de Hilbert. Y por el otro, A. Church y A. Turing resolvían en 1936 de manera negativa el *problema de decisión*: no es decidible determinar si una fórmula es un teorema de un sistema axiomático de primer orden. El resultado de todas estas obras fue la base teórica de la teoría axiomática y semántica.

Toda teoría matemática se construye a partir de unos *axiomas*, que definen las propiedades básicas de los objetos de la teoría que se consideran verdaderas y, sin embargo, no se demuestran. La geometría euclídea y la construcción de los números reales son dos ejemplos de este tipo de teorías axiomáticas. Otro ejemplo de éstas es el modelo matemático conocido como *Álgebra de Boole*, muy importante en informática ya que se utiliza en el diseño de circuitos lógicos y en las búsquedas booleanas en grandes colecciones de datos (indices de páginas Web, datos genéticos, etc.).

Los axiomas de toda teoría matemática tienen que ser:

1. ***Compatibles***: a partir de ellos no tiene que ser posible deducir una contradicción,
2. ***Independientes***: ningún axioma se debe poder deducir a partir del resto de ellos (habría redundancia de axiomas),
3. ***Suficientes***: a partir de ellos tiene que ser posible deducir todas las propiedades que necesitamos satisfagan los objetos de nuestra teoría.

1.3. Lógica e informática

Una nueva época para la lógica comienza en las décadas de 1950 y 1960 a causa de la aparición de los ordenadores. Surgió entonces la necesidad de determinar si era posible especificar formalmente programas y definir sistemas de demostración automática de teoremas. Estos tipo de problemas son los principales objetos de estudio de la lógica informática.

El nacimiento de la inteligencia artificial y del primer lenguaje declarativo (LISP) se puede fijar en el 1959, con el trabajo de Mc Carthy. A lo largo de los años sesenta se mejoran los primeros sistemas de demostración automática y en el 1965 aparece la la regla universal de resolución con unificación de Robinson. En los años setenta se desarrolló la programación lógica como herramienta de resolución de problemas. En 1972 Colmerauer creó el primer lenguaje de programación lógica: *Prolog*.

Además de la lógica proposicional y de predicados (*lógica clásica*), en el siglo XX se desarrollaron otros sistemas lógicos. Los sistemas lógicos clásicos son los más estudiados y utilizados, se caracterizan por incorporar principios tradicionales que otras lógicas rechazan, como el principio del tercero excluido, el principio de no contradicción, etc. A partir de los años ochenta se empiezan a utilizar nuevas lógicas no clásicas, algunas de las cuales permiten dar una interpretación probabilista de la incertidumbre, o rechazan uno o más principios de la lógica clásica. Por ejemplo, la *lógica difusa* rechaza el principio del tercero excluido y propone un número infinito de valores de verdad; diversas *lógicas modales* utilizan expresiones para calificar la verdad de los juicios.

Los métodos deductivos de la lógica matemática son la base de la demostración automática de teoremas. Por lo que la informática trata de buscar los sistemas de demostración más eficientes para su implementación en un ordenador. Definida la semántica de un lenguaje de programación, se pueden

usar los métodos de demostración de la lógica matemática para verificar (automáticamente) la corrección de programas y sus propiedades.

La *programación lógica* forma parte de la base de la inteligencia artificial y permite deducir nuevos conocimientos a partir de una base de conocimientos (los axiomas) y una serie de deducciones automáticas. Por tanto, algunas de las áreas de aplicación de la lógica en informática son:

- La minería de datos.
- La description de la semántica de los lenguajes de programación y la verificación de programas.
- La demostración automática de teoremas.
- La programación lógica y los sistemas basados en el conocimiento en la inteligencia artificial.

2. Sintaxis del Cálculo Proposicional

2.1. Introducción

Como se dijo, la lógica juega un papel básico en la informática, brinda una herramienta teórica para especificaciones formales en diferentes áreas: lenguajes de programación, bases de datos, diseño y verificación de sistemas de hardware y software, complejidad computacional, inteligencia artificial, etc., y es sin duda uno de los fundamentos que proporcionan la madurez y agilidad necesarias para asimilar los conceptos, lenguajes, técnicas y herramientas informáticas que vayan surgiendo en el futuro. Los informáticos necesitan analizar las propiedades lógicas de sus sistemas mientras los diseñan, desarrollan, verifican y mantienen.

La palabra ***lógica*** deriva del griego antiguo *logikĕ*, que significa dotado de razón, intelectual, dialéctico, argumentativo; este vocablo a su vez proviene de la palabra *lógos*, que significa palabra, pensamiento, idea, razón o principio. El objeto de estudio de la lógica son las formas de razonamiento.

El razonamiento es el proceso por el cual se derivan conclusiones a partir de premisas, apoyándose en verdades supuestas. El razonamiento también sirve para justificar ciertas verdades, es decir para determinar si una verdad es consecuencia (formal) de conocimientos previamente aceptados. Entonces, la lógica elemental tiene como objetivo determinar si nuestros razonamientos, independientemente de su contenido, son correctos o incorrectos.

Cuando se quieren examinar los mecanismos de razonamiento con precisión matemática, es necesario que el lenguaje que se utilice no dé lugar a confusiones. Esto se consigue mediante un lenguaje simbólico en el cual cada símbolo tenga un significado bien definido. El mismo posibilita la formalización del lenguaje natural, permitiendo analizar las proposiciones obtenidas independientemente de su significado.

Esto es lo que permite hacer el *cálculo proposicional* o lógica proposicional, también conocida como lógica de enunciados o lógica de orden cero.

El cálculo proposicional es un sistema formal cuyos elementos más simples representan proposiciones, y cuyos operadores, llamados conectivos, son capaces de modificar estas proposiciones o conformar otras de mayor complejidad. Este modelo matemático nos ayuda a comprender mejor las formas básicas

del pensamiento racional, y también nos permite simular esquemas de deducción por medio de computadoras.

Consideremos una frase en lenguaje natural, en primer lugar, podemos observar si se trata de una frase simple o de una frase compuesta. Una frase simple consta de un sujeto y un predicado. Por ejemplo:

- Java es un lenguaje de programación.
- Android es un sistema operativo moderno.

Una frase compuesta se forma a partir de frases simples por medio de algún término de enlace (o conectivo). Por ejemplo:

- Java es un lenguaje de programación y Java es compatible con Android.
- Si Android es un sistema operativo moderno entonces Android soporta Java.

En segundo lugar, se va a asumir que todas las frases simples pueden ser verdaderas o falsas. Pero al considerar el lenguaje natural encontramos expresiones de las que no tiene sentido preguntarse si son verdaderas o falsas, aunque las mismas representen algún concepto; ejemplo de ésto son expresiones como “perro", “biología", al igual que oraciones como las interrogativas, exclamativas o imperativas.

Por lo tanto, para definir una *proposición* diremos que es una expresión de la que tiene sentido decir si es verdadera o falsa; es decir, lo que en gramática se conoce como *oración declarativa*, aquellas expresiones que enuncian hechos o describen situaciones.

Dadas distintas proposiciones, podemos distinguir entonces dos tipos: simples y compuestas. Una proposición se dice *simple* si en ella no aparece ningún término de enlace o conectivo, son las que representan frases simples. En cambio, si dos o más proposiciones se relacionan por medio de algún término de enlace (*“o”*, *“y”*, *“si...,entonces...”*, *“no”*), se obtienen las proposiciones *compuestas*, son las que representan las frases compuestas del ejemplo. Los conectivos o términos de enlace pueden relacionar dos proposiciones simples, como es el caso de *“o”*, *“y”*, *“si...,entonces...”*, o pueden utilizarse sobre una sola proposición simple como el *“no”*.

Una propiedad básica de estas combinaciones es que la verdad o falsedad de la proposición compuesta resultante, depende de la verdad o falsedad de las proposiciones simples que la componen y de los conectivos que se utilizaron para armarla; mientras que la verdad o falsedad de una proposición simple sólo depende de si el hecho o situación que ésta describe es verdad o no.

Así, por ejemplo, la proposición compuesta *El aire contiene nitrógeno y los Beatles visitaron la Argentina en 1963* será verdadera si y solamente si cada una de las proposiciones que la componen es verdadera; es decir, solo será verdadera si “el aire contiene nitrógeno” es verdadera y “los Beatles visitaron la Argentina en 1963” también es verdadera.

En cambio, se sabrá si la proposición “el 25 de agosto de 1930 llovió en Buenos Aires” es verdadera o también la proposición “el 27 de marzo de 2003 llovió en San Luis” solamente verificando qué pasó en esos días en ambas ciudades.

Como se dijo, el cálculo proposicional es un sistema formal cuyos elementos más simples representan proposiciones, que pueden ser modificadas por sus operadores, llamados conectivos. Al referirnos a

proposiciones supondremos que hemos eliminado todo tipo de ambigüedad del lenguaje natural en la formalización de las mismas. Entonces, el primer paso es encontrar un sistema de notación que nos permita expresar simbólicamente las proposiciones simples y las que se obtienen al combinarlas por medio de las partículas lógicas *"o"*, *"y"*, *"si...,entonces..."*, *"no"*.

Por otro lado, al expresar una oración en un cierto lenguaje, oral o escrito, su forma dependerá de las leyes gramaticales del mismo. Por ejemplo no podemos decir

Mis amigos y yo voy al cine

Esta oración está mal formada porque no hay concordancia entre el número del sujeto (plural) y el número del predicado (singular). También en matemáticas hay reglas que nos indican qué combinaciones de símbolos podemos usar; si encontramos la expresión

$$\% = 4 + (85-)$$

no sabríamos cómo interpretarla porque la expresión no respeta las reglas de formación de las fórmulas matemáticas. En general, las leyes gramaticales de los lenguajes que usamos corrientemente no son suficientes para caracterizar completamente a las proposiciones. Por esto es necesario establecer junto con el conjunto de símbolos que nos permitan formalizar frases del lenguaje natural, las reglas sintácticas que nos indican qué combinaciones de símbolos son correctas y cuáles no lo son.

Para comprender la importancia de un sistema de notación, basta observar que los cálculos aritméticos (esto es suma, resta, multiplicación, división de números naturales) se pueden presentar en la forma simple que aprendemos en los primeros grados de la escuela primaria .[1] La numeración decimal no es más que un sistema de notación que nos permite escribir (y leer) cualquier número natural usando sólo las cifras 0, 1, 2, 3, 4, 5, 6, 7, 8, 9.

Las operaciones aritméticas se aprenden en la escuela como reglas para manipular estas listas de símbolos. Por ejemplo, dadas las listas "5643" y "97", si las combinamos con el símbolo "+" obteniendo la expresión "5643+97", aplicando las reglas para sumar obtenemos la lista "5740".

Con el tiempo se comprendió que la característica esencial de este sistema es su carácter posicional, y que el número 10 puede ser sustituido por cualquier número $b > 1$, obteniéndose los sistemas de numeración en base b, que permiten escribir cualquier número con b símbolos básicos. Por ejemplo, en el sistema binario ($b = 2$), todo número se puede escribir como una lista de ceros y unos.

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

Para intentar comprender mejor las analogías y diferencias entre los sistemas de numeración y los de escritura de lenguajes naturales, vamos a introducir los siguientes conceptos.

Definición 1 *Dado un conjunto finito y no vacío A, indicaremos con A^* al conjunto de todas las listas finitas de elementos de A. En particular, la lista vacía será indicada por λ. Llamaremos* ***lenguaje sobre el alfabeto*** *A a cualquier subconjunto $\mathcal{L} \subseteq A^*$. Los elementos de un lenguaje $\mathcal{L}$ son llamados* ***palabras****.*

Ejemplo 1:

[1] Gracias al sistema de numeración decimal, aparentemente inventado en la India entre los siglos IX y X de nuestra era, y difundido por los árabes.

Sea $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$. Si pensamos en un lenguaje que permita escribir los números naturales, se puede definir el lenguaje $\mathcal{L} = A^* - \{\lambda\}$. Como los ceros a la izquierda no son significativos en matemáticas, cada número estaría representado por infinitas palabras de $\mathcal{L}$; así: 5, 05, 005, 0005, ... matemáticamente representan el mismo número, el 5, aunque para el lenguaje $\mathcal{L}$ son todas palabras diferentes. □

Entonces, si buscamos un lenguaje que permita escribir los número naturales de manera tal que cada cadena represente a un número, podemos pensar en algo como:

Ejemplo 2:

Sea $A = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$ y sea $\mathcal{M}$ el conjunto de todas las listas de $A^* - \{\lambda\}$ *que no comienzan con 0*. Entonces $\mathcal{L} = \mathcal{M} \cup \{0\}$ es un lenguaje sobre A, en el que cada número puede ser escrito de una manera única. □

Ejemplo 3:

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

Definición 2 *Sea A un alfabeto. Para $s \in A^*$, llamaremos **longitud de s**, al número de símbolos de A que figuran en la lista s, contados tantas veces como aparezcan.*

Así si A es el alfabeto del Ejemplo 1, la longitud de "000430" es seis.

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

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

$$A^n \subseteq A^* \; y \; A^n = \{s \in A^* / log(s) = n\} \tag{1}$$

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

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

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

Así, si A es el alfabeto del el Ejemplo 3 y $s =$ "casa" y $t =$ "caballo", entonces $st =$ "casacaballo". Observemos que si bien s y t pertenecen al castellano, la lista st no pertenece a este lenguaje.

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

$$long(st) = long(s) + long(t) \tag{2}$$

Una primera diferencia que conviene destacar entre los lenguajes definidos en los Ejemplos 1, 2 y 3 es la siguiente: en los dos primeros, basta inspeccionar una lista s para saber si pertenece o no al

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

En cambio, para saber si una lista de símbolos s pertenece o no al lenguaje definido en el Ejemplo 3, debemos recurrir a un diccionario. No podemos saber si "ñaca" pertenece o no a este lenguaje, sin recurrir al Diccionario de la Real Academia, por como éste fue definido.

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

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

No siempre es posible dar una descripción tan directa de las listas de símbolos que forman un lenguaje como fue hecho en los casos de los Ejemplos 1 y 2. El método que seguiremos ahora para definir el lenguaje del cálculo proposicional sirve de modelo para la definición de muchos otros lenguajes.

2.2. El lenguaje del cálculo proposicional

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

Definición 3 *Teniendo en cuenta esto, definimos como* ***alfabeto del cálculo proposicional*** *al conjunto:*

$$A = \{\ p,\ |,\ \neg,\ \vee,\ \wedge,\ \ \rightarrow,\ (,\ )\ \} \tag{3}$$

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

Los símbolos $\neg, \vee, \wedge, \rightarrow$ son llamados *conectivos proposicionales* (o conectivos lógicos) y son los términos de enlace que permitirán armar proposiciones compuestas. El paréntesis izquierdo y el paréntesis derecho tienen como única función quitar la ambigüedad a ciertas expresiones (tal como se hace con las operaciones aritméticas sin una precedencia definida).

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

Ya tenemos el alfabeto, entonces definiremos las reglas sintácticas que nos indican cuales son las cadenas válidas del lenguaje $\mathbf{Form} \subseteq A^*$. Sus elementos son llamados *fórmulas* y constituyen las palabras del lenguaje del cálculo proposicional.

Definición 4 *Una lista de símbolos de A es una* ***fórmula*** *si y sólo si se la puede obtener aplicando un número finito de veces las siguientes reglas:*

(FP1) *Las variables proposicionales son fórmulas.*

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

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

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

Definición 5 *Una **cadena de formación de longitud n** es una sucesión finita $X_1, X_2, \ldots, X_n$ de elementos de A^* que satisface las condiciones siguientes:*

(CF) *Para cada i, con $1 \leq i \leq n$, se tiene que:*

* *o bien X_i es una variable proposicional,*
* *o bien existe un j tal que $1 \leq j \leq i-1$ y $X_i = \neg X_j$,*
* *o bien existen j y k, ambos entre $1 \leq j, k \leq i-1$, tales que*

$$X_i = (X_j \vee X_k) \quad o \quad X_i = (X_j \wedge X_k) \quad o \quad X_i = (X_j \to X_k)$$

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

Considerando este nuevo concepto, la Definición 4 puede ahora reescribirse de la siguiente forma:

Definición 6 *Una lista de símbolos $P \in A^*$ es una fórmula si y sólo si existe una cadena de formación $X_1, X_2, \ldots, X_n$ tal que $P = X_n$.*

Así diremos que $X_1, X_2, \ldots, X_n$ es *una cadena de formación para una fórmula P* si $X_n = P$. Evidentemente esto da finitud al proceso de generar una fórmula explicado en la Definición 4, cuando el último eslabón de una cadena de formación es igual a la lista de caracteres que se quiere escribir, el proceso termina y se puede asegurar que esa lista es una fórmula.

Observemos que si $X_1, X_2, \ldots, X_n$ es una cadena de formación, entonces para cada $i \leq n$, la secuencia $X_1, \ldots, X_i$ es una cadena de formación de longitud i, ya que satisface la Definición 5. Por lo tanto *todos* los eslabones de una cadena de formación son fórmulas.

A título de ejemplo consideremos la siguiente expresión:

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

La sucesión:

$$\begin{aligned} X_1 &= p_0 \\ X_2 &= p_1 \\ X_3 &= p_3 \\ X_4 &= \neg X_3 \\ X_5 &= (X_1 \vee X_2) \\ X_6 &= (X_5 \to X_4) \\ X_7 &= (X_6 \wedge X_1) \end{aligned}$$

es una cadena de formación de P, esto significa que P es una fórmula. Notar que si se decide cortar la cadena de formación en el eslabón 4, ese trozo de cadena también representa una fórmula, en este caso $Q = \neg p_3$.

$$\begin{aligned} X_1 &= p_0 \\ X_2 &= p_1 \\ X_3 &= p_3 \\ X_4 &= \neg X_3 \end{aligned}$$

Otro aspecto a considerar es que la cadena de formación de una fórmula no es única. Si se considera la sucesión:

$$\begin{aligned} Y_1 &= p_3 \\ Y_2 &= \neg Y_1 \\ Y_3 &= p_1 \\ Y_4 &= p_0 \\ Y_5 &= (Y_3 \vee Y_4) \\ Y_6 &= (Y_5 \to Y_2) \\ Y_7 &= (Y_6 \wedge Y_4) \end{aligned}$$

se puede observar que es otra cadena de formación para la fórmula P, ya que todos sus eslabones son válidos y $Y_7 = P$.

También es una cadena de formación para P la sucesión:

$$\begin{aligned} Z_1 &= p_0 \\ Z_2 &= p_1 \\ Z_3 &= p_2 \\ Z_4 &= p_3 \\ Z_5 &= \neg Z_1 \\ Z_6 &= \neg Z_4 \\ Z_7 &= (Z_1 \vee Z_2) \\ Z_8 &= (Z_1 \to Z_3) \\ Z_9 &= (Z_7 \to Z_6) \\ Z_{10} &= (Z_9 \wedge Z_1) \end{aligned}$$

Claramente $P = Z_{10}$, y cada Z_i se obtuvo aplicando la definición **CF**, por lo tanto es su cadena de formación. Esto muestra que, además de existir infinitas cadenas de formación para una misma fórmula, una cadena de formación de P puede tener eslabones superfluos, es decir, que no forman parte del eslabón final X_n, pero cumplen con la definición, como Z_3, Z_5 y Z_8.

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

Naturalmente, nos interesarán aquellas cadenas en las que no figuren más eslabones que los necesarios, esto es, que si le quitamos un eslabón, que no sea el último, a una cadena de formación para una fórmula P, la misma deja de ser cadena de formación. Esto sugiere la siguiente definición:

Definición 7 *Una cadena de formación $X_1, X_2, \ldots, X_n$ se dice **irredundante** (o también llamada **irreducible**) si para todo i tal que $1 \leq i \leq n-1$, la sucesión $X_1, \ldots, X_{i-1}, X_{i+1}, \ldots, X_n$ no es una cadena de formación.*

Comprobar que en el ejemplo anterior, de las cadenas de formación para la fórmula P, tanto X como Y son irreducibles mientras que Z no lo es.

Observemos que las cadenas de formación de longitud 1 son irredundantes ya que si se les saca el único eslabón que tienen *ya no son* cadenas de formación. Una cadena de formación X_1, X_2 es irredundante si y sólo si $X_2 = \neg X_1$, o $X_2 = (X_1 * X_1)$ con X_1 una variable proposicional y siendo $*$ cualquiera de los conectivos binarios $\wedge, \vee, \rightarrow$.

Una cadena de formación de longitud 3, X_1, X_2, X_3, es irredundante si y sólo si se da alguno de los siguientes casos:

a) $X_3 = (X_1 * X_2)$ o $X_3 = (X_2 * X_1)$, con X_1 y X_2 variables proposicionales distintas y $*$ uno de los conectivos $\vee$, $\wedge$, $\rightarrow$, o bien

b) $X_3 = (X_1 * X_2)$ o $X_3 = (X_2 * X_1)$, con X_1 una variable proposicional y $X_2 = \neg X_1$, y $*$ uno de los conectivos $\vee$, $\wedge$, $\rightarrow$, o bien

c) $X_3 = \neg X_2$, siendo que la secuencia X_1, X_2 era una cadena de formación irredundante.

d) $X_3 = (X_2 * X_2)$, siendo que la secuencia X_1, X_2 era una cadena de formación irredundante y $*$ uno de los conectivos $\vee$, $\wedge$, $\rightarrow$.

Queda para el lector caracterizar las cadenas de formación irredundantes de longitud 4, y también encontrar todas las cadenas de formación irredundantes de la fórmula P considerada anteriormente.

Conviene insistir con el hecho de que, por el momento, estamos considerando a las fórmulas como meras listas de símbolos. Por lo tanto cuando decimos que $P = Q$ queremos significar que en P y en Q aparecen exactamente los mismos símbolos y en el mismo orden. Si consideramos las expresiones $(p_0 \vee p_1)$ y $(p_1 \vee p_0)$ no son iguales, vistas como listas de símbolos, pues tienen los mismos símbolos pero en diferente orden.

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

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

Por lo tanto, afirmar que ciertas propiedades son comunes a *todas* las fórmulas puede resultar complicado, si se debe verificar que cada una de ellas las cumple. Nuestro próximo objetivo, entonces, es describir un método inductivo que nos permita asegurar que ciertas propiedades las tienen todas las fórmulas. Comenzaremos por la siguiente definición:

Definición 8 *Un subconjunto $S \subseteq A^*$ se dice **cerrado por los conectivos** si satisface las siguientes condiciones:*

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

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

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

Teorema 1 *Sea S un subconjunto de A^* cerrado por los conectivos. Si S contiene a todas las variables proposicionales, entonces S contiene a todas las fórmulas.*

Demostración: Sea S un subconjunto de A^* cerrado por los conectivos y que contiene a todas las variables proposicionales. Queremos ver que para toda fórmula P, se tiene que $P \in S$. Para ello razonaremos por inducción en la longitud de las cadenas de formación de P.

Supongamos primero que P tiene una cadena de formación de longitud $n = 1$. Esto sólo es posible si P es una variable proposicional, o sea que $P = p_k$ para algún número natural k, y en este caso $P \in S$ por hipótesis.

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

Supongamos que P admite una cadena de formación de longitud n, digamos $X_1, X_2, \ldots, X_n$, con $X_n = P$. Si X_n es una variable proposicional, entonces ya sabemos que $P \in S$. Si no lo es, se pueden presentar los siguientes casos:

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

(2) Existen i, j tales que $1 \leq i, j \leq n-1$ tales que $P = (X_i \vee X_j)$, $P = (X_i \wedge X_j)$ o $P = (X_i \to X_j)$.

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

Luego, por la hipótesis inductiva, $X_i \in S$, y por ser S cerrado por los conectivos cumple la propiedad **(C1)**, luego $P \in S$.

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

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

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

El teorema anterior nos da un método para probar que las fórmulas tienen una determinada propiedad. Basta con tomar un conjunto $B \subseteq A^*$, formado por todas las listas de símbolos que satisfacen una

propiedad, digamos "**propiedad-1**", y verificar que ese subconjunto de A^* es cerrado por los conectivos y contiene a todas las variables proposicionales, para asegurar que *todas* las fórmulas están en B, por lo tanto todas las fórmulas satisfacen la propiedad "**propiedad-1**".

A continuación veremos una primera aplicación de este método, que nos permitirá concluir que cada fórmula puede ser leída de una única manera. Para ello, primero debemos definir el siguiente concepto:

Definición 9 *Sea $X \in A^*$, una lista de símbolos de A. Llamaremos **peso de** X, y lo denotaremos por **peso(X)**, al número entero que se obtiene restando al número de paréntesis izquierdos "(" el de paréntesis derechos ")" que figuran en la lista X.*

Observemos, en primer lugar, que la lista X es una secuencia cualquiera de símbolos de A y no necesariamente debe ser una fórmula. Por otro lado, si en X no figuran ni paréntesis izquierdos ni paréntesis derechos, entonces $peso(X) = 0$. En particular, tenemos que:

Para toda variable proposicional p_n,

$$peso(p_n) = 0 \tag{4}$$

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

$$peso(\neg X) = peso(X) \tag{5}$$

y si $*$ denota uno cualquiera de los conectivos $\vee, \wedge$ o $\rightarrow$, entonces:

$$peso((X * Y)) = peso(X) + peso(Y) \tag{6}$$

Lema 1 *Para toda fórmula P valen las siguientes propiedades:*

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

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

Demostración: Comencemos por el punto **(i)**. Llamemos S al conjunto de todas las listas de A^* que tienen peso 0:

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

Del enunciado (4) resulta que todas las variables proposicionales pertenecen a S, y de (5) y (6) se deduce que S es cerrado por los conectivos. Luego por el Teorema 1, debe ser $\mathbf{Form} \subseteq S$, esto es, todas las fórmulas tienen peso cero, lo que prueba el enunciado **(i)**.

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

Esto es, una fórmula P pertenece al conjunto T si y sólo si asumiendo que P es de la forma $P = X * Y$, donde $X, Y \in A^*$, se cumple que el $peso\ (X) > 0$.

Como en las variables proposicionales sólo figuran los símbolos p y $|$, éstas pertenecen trivialmente al conjunto T. Por lo tanto, sabiendo que las variables proposicionales pertenecen a T, si demostramos que T es cerrado por los conectivos, sabremos que *todas* las fórmulas están en T.

Para probar que T es cerrado por los conectivos, suponemos que $P \in T$ y que $\neg P = X * Y$. Recordar que si bien P es fórmula, también es una lista de símbolos de A^*, por lo cual, tanto X como Y pueden verse solo como listas de A^*. Esta última condición implica que existe $Z \in A^*$ tal que $X = \neg Z$, y por lo tanto que $P = Z * Y$. Como $P \in T$, debe ser $peso(Z) > 0$, y por (5) se tiene que también $peso(X) > 0$. Luego $\neg P \in S$. Por lo tanto T satisface la primer condición (**(C1)**) de la definición de conjunto *cerrado por los conectivos* (Definición 8).

Para probar que también satisface la segunda condición, **(C2)**, de la misma definición, vamos a suponer que P y Q son fórmulas que pertenecen a T y sea $*$ uno de los símbolos $\vee, \wedge$ o $\rightarrow$ que figuran en la lista $(P \vee Q)$. Si $*$ figura a la izquierda de $\vee$, entonces deben existir X, Y en A^* tales que $P = X * Y$, y $(P \vee Q) = (X * Y \vee Q)$. Como $P \in T$, debe ser $peso(X) > 0$, y como $peso((X) = 1 + peso(X)$, resulta que la expresión a la izquierda de $*$ en $(P \vee Q)$ tiene peso mayor que cero.

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

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

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

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

Como T es un subconjunto de A^* que contiene todas las variables proposicionales y es cerrado por los conectivos, se cumple que $\mathbf{Form} \subseteq T$: es decir todas las fórmulas satisfacen **(ii)**, la condición que define al conjunto T, c.q.d. ■

Ejemplo:

Dada la siguiente fórmula del lenguaje:

$$P = (((p_0 \vee p_1) \rightarrow \neg p_3) \wedge (p_0 \rightarrow p_5))$$

podemos observar que $peso(P) = 0$ y además se puede verificar, que a la izquierda del conectivo $\vee$ el peso de la lista de símbolos $(((p_0$ es $3 > 0$; lo mismo ocurre con el peso de la lista a la izquierda de $\rightarrow$, *peso*$((((p_0 \vee p_1))= 2 > 0$. Queda para el lector calcular el peso de las listas de símbolos a la izquierda de los otros conectivos de la fórmula.

□

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

Teorema 2 (Unicidad de la lectura de las fórmulas) *Sean P, Q, R y S fórmulas, y supongamos que $*$ y $\circ$ representan a alguno de los conectivos $\vee$, $\wedge$ o $\rightarrow$.*

$$\textit{Si } (P * Q) = (R \circ S) \textit{ entonces debe ser } P = R,\ Q = S \textit{ y } * = \circ.$$

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

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

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

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

Se definirán algunos conceptos que serán de utilidad más adelante.

Definición 10 *Llamaremos **grado de complejidad de una fórmula** P, y lo denotaremos por **comp(P)**, al número de conectivos que figuran en P, contados tantas veces como aparezcan.*

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

Corolario 1 *Para toda fórmula P se tiene que:*

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

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

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

Demostración: La parte **(i)** del enunciado es obvia ya que las variables proposicionales no incluyen conectivos. La parte **(ii)** es una consecuencia inmediata de la definición de fórmula y del Teorema 2. ■

Reconocimientos

El presente apunte se realizó tomando como base el Apunte de *Cálculo Proposicional* del **Dr. Roberto Cignoli** de la Universidad de Buenos Aires, el libro *Lógica para informática* de **Claudia Pons, Ricardo Rosenfeld y Clara Smith** de Facultad de Informática de la Universidad Nacional de La plata (UNLP) y notas de clase del *Curso de Lógica y Computación* dictado por el **Dr. Guillermo Martínez** en la Universidad Nacional de San Luis.