Matemáticas

8° Semestre

Lógica matemática

Unidad 1. Lógica de enunciados o proposiciones

Clave:
05144845

Universidad Abierta y a Distancia de México

Índice

Unidad 1. Lógica de enunciados o de proposiciones

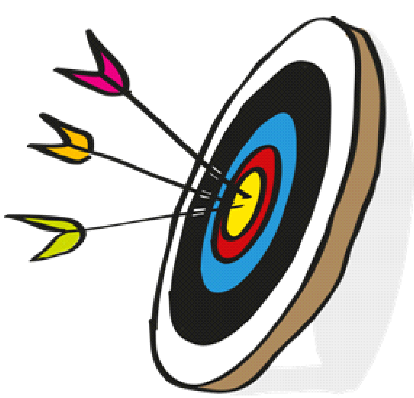
Presentación de la unidad

En esta unidad se introducirán los conceptos de lenguajes formales, símbolos de lógica de enunciados, reglas y el principio de inducción, que permitirán generar procesos lógicamente estructurados en la búsqueda de soluciones de problemas en distintas áreas, para lo cual se contará con las herramientas, el lenguaje y el pensamiento lógico; para determinar si un enunciado o solución está estructurado correctamente y, por lo tanto, si es válido.

Competencia específica

Aplica el lenguaje de la lógica de proposiciones mediante el cálculo proposicional para comprender el concepto de sistema formal.

Logros



- Usar procesos lógicos para comprender los lenguajes formales.
- Utilizar los símbolos de la lógica para formar enunciados.
- Utilizar la lógica matemática en fórmulas conocidas.
- Utilizar el principio de inducción para validar enunciados.

1.1. El lenguaje de la lógica de enunciados

La lógica de enunciados o de proposiciones permite traducir proposiciones enunciadas en lenguaje natural a símbolos lógicos o matemáticos para analizarlas y asignarles un valor de verdad o falsedad.

En la lógica de enunciados (o lógica proposicional), se estudia la composición de enunciados mediante los conectivos (“y”, “o”, “si…, entonces…”, etcétera), y se fundamenta en **el principio de bivalencia,** en el cual todo enunciado en el lenguaje natural tiene un valor de verdad bien definido, de verdadero o falso, no ambos a la vez.

Para empezar, considera que cada proposición tiene una forma lógica a la que se le da un nombre, unas se denominan proposiciones atómicas y otras proposiciones moleculares.

Proposición atómica: es una proposición simple sin términos de enlace.
Proposición molecular: se forma de una o varias proposiciones atómicas y términos de enlace (o conectivos).

En lo siguiente, se definirá más precisamente el lenguaje que se va a usar y cómo se determinan los valores.

1.1.1. Introducción a los lenguajes formales

El uso del lenguaje formal permite escapar de las ambigüedades e imprecisiones de los lenguajes naturales; sin embargo, los lenguajes formales tienen un grado de expresividad limitado.

A diferencia de los lenguajes naturales (como el español o el alemán, por mencionar algunos), éste será un lenguaje formal, con reglas de precisión y características de este lenguaje, para explicarlo considera el siguiente ejemplo:

Ejemplo

El enunciado “Se observaron rastros de potasio” se puede traducir al lenguaje formal, usando, digamos, el símbolo K, para el enunciado relacionado “No se observaron rastros de potasio” se puede utilizar $(\neg K)$, el símbolo $\neg$ es el símbolo para la negación. Para un enunciado que no está relacionado "La muestra contenía cloro”, se elige el símbolo C, a partir de estos dos enunciados, es posible definir enunciados compuestos y representarlos de forma simbólica de la siguiente manera:

El enunciado compuesto “Si se observaron rastros de potasio, entonces la muestra no contenía cloro” se puede representar como $(K \rightarrow (\neg C))$, donde la flecha es traducción de “si…, entonces…”.

En el enunciado “La muestra contenía cloro y se observaron rastros de potasio”, se usará el símbolo ∧ como traducción de “y”, es decir:

$$(\mathrm{C} \wedge \mathrm{K})$$

En el siguiente enunciado se usará el símbolo ∨ para la disyunción “o”
“No se observaron rastros de potasio, o la muestra no contenía cloro”

$$((\neg K) \vee (\neg C))$$

En el siguiente enunciado hay dos formas alternativas para la traducción del enunciado que en el lenguaje natural son entendidas como “lo mismo”.
“Ni la muestra contenía cloro, ni se observaron restos de potasio”

$$(\neg(\mathrm{C} \wedge \ \mathrm{K})) \text{ o bien } ((\neg \mathrm{C}) \wedge (\neg \mathrm{K}))$$

Un aspecto importante de la descomposición de los enunciados compuestos es que sabiendo la verdad o falsedad de las partes atómicas, se puede saber la verdad o falsedad del compuesto. Considera entonces la siguiente situación: “El químico sale de su laboratorio y anuncia que observó rastros de potasio pero que la muestra no contenía cloro”; de esta manera, puedes saber la verdad o falsedad respectivamente de los cuatro enunciados mencionados, es posible construir la siguiente tabla (Tabla I) para ejemplificar los cuatro resultados experimentales posibles, la descripción de este método se dará más adelante.

Tabla 1

$\boldsymbol{K}$	$\boldsymbol{C}$	$(\neg(\boldsymbol{C} \vee \boldsymbol{K}))$	$((\neg \boldsymbol{C}) \wedge (\neg \boldsymbol{K}))$
$\boldsymbol{F}$	F	V	V
$\boldsymbol{F}$	V	F	F
$\boldsymbol{V}$	F	F	F
$\boldsymbol{V}$	V	F	F

Generalmente se incluyen tres tipos de datos en la descripción de un lenguaje formal:

1. Especificar el tipo de símbolos (**el alfabeto**) en la lógica de enunciado. Algunos símbolos son $(,), \rightarrow, \neg, A_1, A_2, \ldots$

2. Especificar las reglas de formación para la sucesiones finitas de símbolos “gramaticalmente correctas” (éstas se llaman **fórmulas**). Por ejemplo:
   Las sucesiones de símbolos $(A_1 \rightarrow (\neg \mathrm{A}_2))$ es una fórmula, mientras que $)) \rightarrow \mathrm{A}_3$ no lo es.

3. Indicar las traducciones permisibles entre el español y el lenguaje formal. Los símbolos $A_1, A_2, \ldots$ pueden ser traducciones de enunciados declarativos del español.

La tercera parte es en la que se da significado a las fórmulas. Este proceso de asignación de significado es lo que motiva a hacer todo esto. Aunque también es posible que se realicen manipulaciones en las fórmulas sin tomar en cuenta cualquier significado posible. Si una persona sólo estuviera al tanto de los dos primeros datos de la descripción del lenguaje, puede que realice cualquiera de las cosas que se harán, pero esto no tendría ningún sentido para ella.

Brevemente se examinan algunas clases de lenguajes formales con una importancia muy grande en la vida moderna.

Una clase de lenguaje formal de interés generalizado hoy en día son los usados por las computadoras digitales (o que al menos están en conexión con ellas); existen muchos de estos lenguajes, una fórmula típica en uno de ellos es:

$$01101011010100011111000100000111010$$

En otro, una fórmula típica es:

$$STEP\#ADDIMAZ, A$$

Un lenguaje de programación muy conocido es C++ que tiene fórmulas como ésta:

$$\text{while}(*s++)$$

En cada lenguaje formal existe un procedimiento para traducir las fórmulas al español y (para una clase restringida de oraciones del español) un procedimiento para traducir del lenguaje natural al formal.

1.1.2. Símbolos de la lógica de enunciados

Se asume que se tiene una sucesión infinita de objetos distintos llamados símbolos, a los cuales se les da nombre en la tabla siguiente (basada en la que se encuentra en *A Mathematical Introduction to Logic*, p. 14). Se asume también que ninguno de estos símbolos es una sucesión finita de otros.

Tabla #

Símbolo	Nombre largo	Observaciones
(	Paréntesis izquierdo	Puntuación
)	Paréntesis derecho	Puntuación
$\neg$	Símbolo de negación	Español: no
$\wedge$	Símbolo de conjunción	Español: y

$\vee$	Símbolo de disyunción	Español: o (inclusivo)
$\rightarrow$	Símbolo condicional	Español: si ___, entonces __
$\leftrightarrow$	Símbolo bicondicional	Español: si y sólo si
$\boldsymbol{A_1}$	Primer símbolo de enunciado	
$\boldsymbol{A_2}$	Segundo símbolo de enunciado	
$\vdots$	$\vdots$	
$\boldsymbol{A_n}$	Enésimo símbolo de enunciado	

Considera las siguientes observaciones:

1. Los símbolos $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ son llamados **conectivos de enunciados**. La forma en que se usan es sugerida por la traducción que se presenta en la tabla. Los conectivos, junto con los paréntesis, son **símbolos lógicos**. Los símbolos de enunciado son los **parámetros** (o símbolos no lógicos), su traducción no está fija; más bien estará abierta a interpretación.

2. Se introdujeron una cantidad numerable de símbolos de enuncuado. La mayor parte de las cosas que se presentan en este tema se aplican bien, si se considera un conjunto arbitrario de símbolos de enunciado.

3. Algunos autores prefieren llamar a A_n el enésimo símbolo de **proposición** (y hablar de lógica proposicional). Esto porque se quiere que la palabra “enunciado” se refiera a un tipo particular de oración y que una proposición sea lo que un enunciado afirma.

4. Se denominaron “símbolos”, pero esto es, siendo neutrales, acerca del *status* ontológico de los símbolos. Por ejemplo A_{43} es un símbolo, a saber el cuadragésimo tercero de enunciado, (por otra parte, “A_{43}” es el nombre de éste). El símbolo condicional puede o no contar con la propiedad geométrica de estar conformado como una flecha, aunque su nombre “$\rightarrow$” sí la tiene). Los símbolos pueden ser conjuntos, números, canicas u objetos pertenecientes a un universo de objetos lingüísticos. En este último caso se puede tener que sean de hecho las mismas cosas que los nombres que usamos para ellos. Otra posibilidad es que los símbolos de enunciados sean fórmulas de otro lenguaje.

5. Al suponer que ningún símbolo es una sucesión finita de otros símbolos se tiene que los símbolos de la lista son distintos (por ejemplo: $A_3 \neq \leftrightarrow$) y se exige que $A_3 \neq \langle \neg, A_4, (\rangle$. El propósito de esta descomposición es asegurar que las sucesiones finitas de símbolos tengan una descomposición única. Es decir, si:

$$\langle a_1, \dots, a_m \rangle = \langle b_1, \dots, b_n \rangle$$

con a_i y b_j símbolos, entonces $m = n$ y $a_i = b_i$.

Una **expresión** es una sucesión finita de símbolos. Se puede especificar una expresión concatenando los nombres de los símbolos; así, $(\neg A_1)$ es la sucesión $\langle (, \neg, A_1,) \rangle$. Esta notación se extiende: si α y β son sucesiones de símbolos, entonces $\alpha\beta$ es la sucesión que consiste en los símbolos de la sucesión α seguidos de los símbolos de la sucesión β. Por ejemplo, si α y β son las expresiones dadas por las ecuaciones:

$$\alpha = (\neg A_1)$$
$$\beta = A_2$$

Entonces, la expresión $(\alpha \rightarrow \beta)$ es la expresión $\left((\neg A_1) \rightarrow A_2\right)$.

A continuación, se presentan dos ejemplos de traducciones de enunciados del lenguaje natural a expresiones del lenguaje formal. Sean $A, B, \ldots, Z$ los primeros 27 símbolos de enunciado.

1. Español: El sospechoso debe ser liberado. Traducción: R.
   Español: La evidencia obtenida es admisible. Traducción: E.
   Español: La evidencia obtenida es inadmisible. Traducción: $(\neg E)$.
   Español: La evidencia obtenida es admisible y el sospechoso puede no ser liberado. Traducción: $\left(E \wedge (\neg R)\right)$.
   Español: La evidencia obtenida es admisible o el sospechoso debe ser liberado (o posiblemente ambas cosas). Traducción: $(E \vee R)$.
   Español: O bien la evidencia obtenida es admisible o el sospechoso debe ser liberado, pero no ambas cosas. Traducción: $\left((E \vee R) \wedge \left(\neg(E \wedge R)\right)\right)$. Siempre se usará el símbolo $\vee$ como traducción de la palabra “o” en su sentido inclusivo “y/o”.
   Español: La evidencia obtenida es inadmisible pero el sospechoso puede no ser liberado. Traducción: $\left((\neg E) \wedge (\neg R)\right)$. Por otra parte, la expresión $\left((\neg E) \vee (\neg R)\right)$ se traduce al español como “La evidencia obtenida es inadmisible, o el sospechoso puede no ser liberado”.

2. Español: Si mi abuela tuviera ruedas, sería bicicleta. Traducción: $(R \rightarrow B)$.
   Español: Mi abuela tiene ruedas si y sólo si es bicicleta. Traducción: $(R \leftrightarrow B)$.

Con este respecto ten mucho cuidado, no debes confundir un enunciado del lenguaje natural (Las rosas son rojas) con una traducción de dicho enunciado al lenguaje formal (por ejemplo R) pues son diferentes; el enunciado en español es presuntamente verdadero o falso, pero la expresión formal es sólo una sucesión de símbolos (si bien es posible darle interpretación ésta depende del contexto).

También considera que algunas expresiones no se pueden obtener como traducciones de ningún enunciado en lenguaje natural, tal como…

$$((\rightarrow A_3.$$

A continuación se verá cómo hacer que una fórmula esté bien estructurada o tenga sentido para lo que se está haciendo.

1.1.3. Reglas de formación de fórmulas

Las reglas de formación permiten definir fórmulas que son expresiones "gramaticalmente correctas", excluyendo las que no tienen sentido.

La definición para la formación de fórmulas tendrá las siguientes consecuencias:

(a) Todo símbolo de enunciado es una fórmula.
(b) Si α y β son fórmulas, entonces también lo son $(\neg\, \alpha), (\alpha \wedge \beta), (\alpha \vee \beta), (\alpha \rightarrow \beta), (\alpha \leftrightarrow \beta)$.
(c) Ninguna expresión es fórmula a menos que (a) y (b) obliguen a ello.

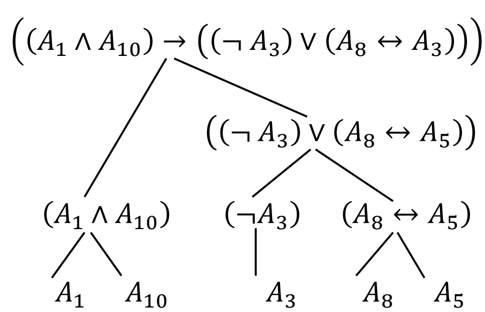
Se quiere precisar la tercera propiedad. Una fórmula bien formada es una expresión que se construye a partir de símbolos de enunciados y de la aplicación, un número finito de veces, de las operaciones de construcción de fórmulas (sobre las expresiones) definidas por las ecuaciones.

$$\begin{aligned} \mathcal{E}_{\neg} &= (\neg\, \alpha), \\ \mathcal{E}_{\wedge}(\alpha,\beta) &= (\ \alpha \wedge \beta), \\ \mathcal{E}_{\vee}(\alpha,\beta) &= (\alpha \vee \beta), \\ \mathcal{E}_{\rightarrow}(\alpha,\beta) &= (\ \alpha \rightarrow \beta), \\ \mathcal{E}_{\leftrightarrow}(\alpha,\beta) &= (\alpha \leftrightarrow \beta). \end{aligned}$$

Por ejemplo, la siguiente expresión es una fórmula que se construye a partir de cuatro símbolos de enunciados aplicando cinco veces operaciones de construcción de fórmulas:

$$\Big((A_1 \wedge A_{10}) \rightarrow \big((\neg\, A_3) \vee (A_8 \leftrightarrow A_3)\big)\Big)$$

El árbol muestra cómo se construye la expresión:



Se puede desarrollar la idea de "construcción" como sigue: Defina una **sucesión de construcción** como una sucesión finita $\langle \varepsilon_1, \ldots, \varepsilon_n \rangle$ de expresiones que, para cada $i \leq n$, se tiene al menos uno de los siguientes hechos:

ε_i es un símbolo de enunciado

$\varepsilon_i = \mathcal{E}_{\neg}(\varepsilon_j)$ para algún $j < i$

$\varepsilon_i = \mathcal{E}_{\square}(\varepsilon_j, \varepsilon_k)$ para algunos $j < i, k < i$

Donde □ es uno de los conectivos binarios $\wedge, \vee, \rightarrow, \leftrightarrow$. De esta manera, las fórmulas pueden caracterizarse como las expresiones α, tales que alguna sucesión de construcción termina con α. Puedes pensar ε_i como la expresión en el estado i del proceso de construcción.

Por ejemplo, para la expresión:

$$\Big((A_1 \wedge A_{10}) \rightarrow \big((\neg\, A_3) \vee (A_8 \leftrightarrow A_5)\big)\Big)$$

Se obtiene una sucesión de construcción al comprimir su árbol genealógico en un orden lineal.

Una característica en este tipo de construcción es que genera un tipo de inducción.

1.1.4. Principio de inducción

1.1.4.1. Definición
Un conjunto S es cerrado bajo una función f de dos argumentos si cada vez que $x, y \epsilon S$, entonces $f(x, y) \in S$, y análogamente para funciones de un argumento, etcétera.

1.1.4.1. Teorema. Principio de inducción
Sea S un conjunto de fórmulas que contiene todos los símbolos de enunciado y es cerrado bajo las cinco operaciones de construcción de fórmulas, entonces S es el conjunto de todas las fórmulas.

Primera demostración
Sea α una fórmula cualquiera, entonces α se construye mediante símbolos de enunciados al aplicar un número finito de veces las operaciones de construcción de fórmulas.
Si se explora el árbol genealógico hacia arriba, se encontrará que cada expresión que existe en el árbol pertenece a S; después de un número finito de pasos se encontrará, en la punta del árbol, que $\alpha \in S$.

■

Segunda demostración
Se usa la misma idea que en la primera demostración, sólo que no se usarán los árboles. Considérese una fórmula α cualquiera, entonces α es el último miembro de alguna sucesión de construcción $\langle \varepsilon_1, \ldots, \varepsilon_n \rangle$, por inducción numérica fuerte ordinaria sobre el número i se observa que cada $\varepsilon_i \in S$, para $i \leq n$.

Es decir, supóngase como hipótesis de inducción que $\varepsilon_j \in S$ para toda $j < i$. Entonces se verifica que $\varepsilon_i \in S$, considerando los diferentes casos. Así, por inducción fuerte sobre i, se sigue que $\varepsilon_i \in S$ para cada $i \leq n$. En particular, el último miembro α pertenece a S.

■

El principio de inducción permite saber si ciertas expresiones no son fórmulas.

Ejemplo
Ninguna expresión con más paréntesis izquierdos que paréntesis derechos es una fórmula.

Demostración
La idea es que, al comenzar a construir fórmulas, se empieza con símbolos de enunciados (que tienen cero paréntesis izquierdos y cero paréntesis derechos), luego aplicando las operaciones de construcción de fórmulas, las cuales agregan paréntesis sólo por parejas, uno izquierdo y uno derecho. El argumento se puede reformular de la siguiente manera: el conjunto de las fórmulas “balanceadas” (las que tienen igual número de paréntesis izquierdos que derechos) tiene como elementos a todos los símbolos de enunciado y es cerrado bajo las operaciones de construcción de fórmulas. Es decir, el conjunto de fórmulas balanceadas es inductivo. El principio de inducción asegura, entonces, que todas las fórmulas son balanceadas.

■

Una característica especial de las operaciones de construcción de fórmulas particulares es que van hacia arriba y nunca hacia abajo. Esto es: las expresiones $\mathcal{E}_{\square}(\alpha,\beta)$ siempre incluyen la sucesión completa α (y la sucesión completa β) más otros símbolos. Por ende, es más largo que α y β.

Esta característica especial simplifica el problema de determinar para cada fórmula φ cómo fue construida exactamente.

Todos los bloques de construcción, por decirlo así, están incluidos como segmento en la sucesión de construcción de φ.

1.2. Asignación de verdad

Se quiere definir lo que significa que una fórmula del lenguaje se siga lógicamente de otras fórmulas. Por ejemplo, A_1 deberá seguirse de $(A_1 \wedge A_2)$, ya que, independiente de cómo se traduzcan los parámetros A_1 y A_2 al español, si la traducción de $(A_1 \wedge A_2)$ es verdadera, entonces la traducción de A_1 debe ser verdadera. Pero la noción de todas las posibles traducciones al español es demasiado vaga; sin embargo, es posible expresar esta idea en una forma simple y precisa.

Se fija un conjunto $\{V, F\}$ de **valores de verdad** que consta de dos puntos distintos:

$$\begin{array}{ll} V, & llamado\ verdad \\ F, & llamado\ falsedad \end{array}$$

Al fijar este conjunto no importa qué sean estos puntos, también podrían ser, por ejemplo, números $1, 0$. Entonces una **asignación de verdad** para un conjunto $\mathcal{S}$ de símbolos enunciados es una función:

$$v: \mathcal{S} \to \{V, F\}$$

que asigna V o F a cada símbolo en $\mathcal{S}$. Estas asignaciones de verdad se usarán en vez de las traducciones al lenguaje natural.

Con lo anterior se está diciendo que se usará una lógica **bivalente**, por supuesto, esto deja la idea de poder usar tres valores verdad, o quizá un conjunto más grande de 1000 valores de verdad, $\aleph_0$ valores de verdad o tomar como conjunto de valores de verdad el intervalo $[0,1]$, todo lo anterior es posible; sin embargo, la lógica bivalente es la que siempre ha sido de mayor significación y en este curso será nuestro objeto de estudio.
Sea $\bar{\mathcal{S}}$ el conjunto de las fórmulas generado a partir de $\mathcal{S}$ por las cinco operaciones de construcción de fórmulas. Se quiere una extensión $\bar{v}$ de v,

$$v : \bar{\mathcal{S}} \to \{V, F\}$$

que asigne el valor correcto de verdad a cada fórmula de $\bar{\mathcal{S}}$ y debe satisfacer las siguientes condiciones:

a) Para cualquier $A \in \mathcal{S}$, $\bar{v}(A) = v(A)$.
b) Para cualesquiera α y β en $\bar{\mathcal{S}}$:
   1. $\bar{v}\big((\neg\alpha)\big) = \begin{cases} V \ si \ \bar{v}(\alpha) = F, \\ F \ en \ los \ demás \ casos. \end{cases}$
   2. $\bar{v}\big((\alpha \wedge \beta)\big) = \begin{cases} V \ si \ \bar{v}(\alpha) = V \ y \ \bar{v}(\beta) = V, \\ F \ en \ los \ demás \ casos. \end{cases}$
   3. $\bar{v}\big((\alpha \vee \beta)\big) = \begin{cases} V \ si \ \bar{v}(\alpha) = V \ o \ \bar{v}(\beta) = V \ (o \ ambos), \\ F \ en \ los \ demás \ casos. \end{cases}$
   4. $\bar{v}\big((\alpha \to \beta)\big) = \begin{cases} F \ si \ \bar{v}(\alpha) = V \ y \ \bar{v}(\beta) = F, \\ V \ en \ los \ demás \ casos. \end{cases}$
   5. $\bar{v}\big((\alpha \leftrightarrow \beta)\big) = \begin{cases} V \ si \ \bar{v}(\alpha) = \bar{v}(\beta), \\ F \ en \ los \ demás \ casos. \end{cases}$

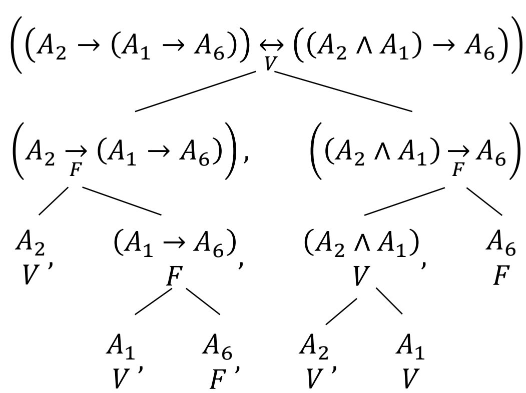
Como ejemplo de la forma de calcular $\bar{v}$, sea α la fórmula:

$$\Big(\big(A_2 \to (A_1 - A_6)\big) \leftrightarrow \big((A_2 \wedge A_1) \to A_6\big)\Big)$$

y sea v la asignación de verdad para $\{A_1, A_2, A_6\}$ tal que:

$$v(A_1) = V$$
$$v(A_2) = V$$
$$v(A_6) = F$$

Se quiere calcular $\bar{v}(\alpha)$. Mira el siguiente árbol que muestra la construcción de α:



Si te desplazas de abajo hacia arriba puedes asignar a cada vértice β del árbol el valor $\bar{v}(\beta)$. Se puede escribir esto de manera más concisa y en una sola línea:

$$((A_2 \to (A_1 \to A_6)) \leftrightarrow ((A_2 \wedge A_1) \to A_6))$$
$$V \;\; F \;\; V \;\; F \;\; F \;\; V \;\; V \;\; V \;\; V \;\; F \;\; F$$

De esta forma se tiene una asignación de verdad a las fórmulas de $\bar{\mathcal{S}}$, el siguiente teorema habla de la unicidad de esta extensión para la función asignación de verdad.

Teorema
Si v es una asignación de verdad para el conjunto $\mathcal{S}$, entonces existe una única función $\bar{v}: \bar{\mathcal{S}} \to \{V, F\}$ que satisface las condiciones anteriores en los incisos a) y b).

Este teorema se sigue del teorema de recursión, que se dará en temas siguientes, y del teorema de unicidad de la lectura de fórmulas, pero te debe parecer claro después del ejemplo. Al demostrar la existencia de $\bar{v}$, el punto importante es, esencialmente, la unicidad de los árboles mencionados en el ejemplo.

Se dice que una asignación de verdad $\boldsymbol{v}$ **satisface** $\boldsymbol{\varphi}$ sii $\bar{v}(\varphi) = V$. Evidentemente, esto significa que todos los símbolos de enunciado de φ deben pertenecer al dominio de v.

1.2.1. Implicación tautológica

Considera ahora un conjunto Σ de fórmulas que juegan el papel de hipótesis y otra fórmula τ, la posible conclusión.

1.2.1.1. Definición
$\boldsymbol{\Sigma}$ implica tautológicamente a $\boldsymbol{\tau}$ ($\boldsymbol{\Sigma} \vDash \boldsymbol{\tau}$) ssi (si y sólo si) toda asignación de verdad para los símbolos de enunciado que ocurren en Σ y en τ, que satisfaga a todos los elementos de Σ también satisface τ.

Esta definición está relacionada con la idea que se tiene de que una conclusión se sigue de un conjunto de hipótesis, que si se suponen verdaderas esto garantiza la verdad de la conclusión.

Se mencionan ahora algunos casos especiales de concepto de implicación tautológica. Primero considera el caso especial en el que Σ es el conjunto $\emptyset$. En este caso se tiene que es cierto por vacuidad que toda asignación de verdad satisface a todos los elementos de $\emptyset$ (esto puedes verlo preguntándote ¿cómo podría ser esto falso? La respuesta es que existiera algún elemento en $\emptyset$ que no satisfaga la propiedad, lo cual es absurdo). Por tanto, se tiene: $\emptyset \vDash \tau$ ssi toda asignación de verdad (para los símbolos de enunciado que ocurren en τ) satisface τ. En este caso se dice que τ es **tautología** ($\vDash \boldsymbol{\tau}$). En el ejemplo que se presentó antes, la fórmula:

$$\Big((A_2 \to (A_1 - A_6)\big) \leftrightarrow \big((A_2 \wedge A_1) \to A_6)\Big)$$

se satisface con cualquiera de los ocho posibles valores de verdad para $\{A_1, A_2, A_6\}$, es decir, sin importar los valores de verdad que se le asignen a los elementos de $\{A_1, A_2, A_6\}$ el enunciado siempre es verdadero, por tanto es una tautología.

Otro caso especial es aquel en que ninguna asignación de verdad satisface a todos los elementos de Σ. Entonces, para toda τ es cierto por vacuidad que $\Sigma \vDash \tau$. Por ejemplo, considera:

$$\{A, (\neg A)\} \vDash B.$$

Se presenta el siguiente ejemplo sobre implicación tautológica.

Ejemplo

Considera $\{A, (A \rightarrow B)\} \vDash B$. Existen cuatro posibles asignaciones de verdad para $\{A, B\}$. Puedes verificar, al calcular los valores de verdad de cada parte, que sólo una de estas cuatro posibilidades satisface tanto a A como a $(A \rightarrow B)$; ésta es la v para la cual $v(A) = v(B) = V$. Y claramente esta v también satisface B.

Si Σ es un conjunto unitario de la fórmula σ, $\Sigma = \{\sigma\}$, entonces se escribe “$\sigma \vDash \tau$” en lugar de “$\{\sigma\} \vDash \tau$”.

1.2.2. Equivalencia tautológica

1.2.2.1 Definición

Si a la vez $\sigma \vDash \tau$ y $\tau \vDash \sigma$, entonces σ y τ se llaman **tautológicamente equivalentes** ($\boldsymbol{\sigma} \vDash\Dashv \boldsymbol{\tau}$). Por ejemplo, cuando se tienen las fórmulas $\big(\neg(C \vee K)\big)$ y $\big((\neg C) \wedge (\neg K)\big)$ como traducciones equivalentes de una oración en español, se afirma que entonces son tautológicamente equivalentes.

Se puede enunciar sin demostración, por ahora, un hecho realmente importante y no trivial.

1.2.2.1. Teorema [teorema de compacidad]

Sea Σ un conjunto infinito de fórmulas tal que para todo subconjunto finito Σ_0 de Σ existe una asignación de verdad que satisface a todos los elementos de Σ_0. Entonces existe una asignación de verdad que satisface a todos los elementos de Σ.

Se puede reformular el teorema anterior de una manera más sencilla como sigue: Si todo subconjunto finito de Σ es satisfactible, entonces Σ es satisfactible.

1.2.3. Tablas de verdad

Existe un procedimiento sistematico para verificar, dadas las fórmulas $\sigma_1, \ldots, \sigma_k$ y τ, si $\{\sigma_1, \ldots, \sigma_k\} \vDash \tau$ o no. En particular, cuando $k = 0$, el procedimiento decidirá, dada una fórmula, si es tautología o no. Como primer ejemplo, se verá que

$$\big(\neg(A \wedge B)\big) \vDash \big((\neg A) \vee (\neg B)\big).$$

Para esto, considera todas las asignaciones de verdad para $\{A, B\}$, puedes ver que para un conjunto de n símbolos hay 2^n asignaciones de verdad, las cuatro asignaciones en este caso se pueden escribir en una tabla:

$\boldsymbol{A}$	$\boldsymbol{B}$
V	V
V	F
F	V
F	F

Esta tabla se puede extender para que incluya a $\big(\neg(A \wedge B)\big)$ y $\big((\neg A) \vee (\neg B)\big)$. Para cada fórmula se calculan los valores de V y F, como se dijo, escribiendo el valor de verdad debajo de la conectiva correcta, esto se presenta en la siguiente tabla:

$\boldsymbol{A}$	$\boldsymbol{B}$	$\big(\neg(\boldsymbol{A} \wedge \boldsymbol{B})\big)$	$\big((\neg \boldsymbol{A}) \vee (\neg \boldsymbol{B})\big)$
V	V	$F\ V\ V\ V$	$F\ V\ \ F\ \ F\ V$
V	F	$V\ V\ F\ F$	$F\ V\ \ V\ \ V\ F$
F	V	$V\ F\ F\ V$	$V\ F\ \ V\ \ F\ V$
F	F	$V\ F\ F\ F$	$V\ F\ \ V\ \ V\ F$

Puedes ver que todas las asignaciones de verdad que satisfacen a $\big(\neg(A \wedge B)\big)$, y que son tres, tambien satisfacen a $\big((\neg A) \vee (\neg B)\big)$. También es cierta la conversa y, por tanto, son tautológicamente equivalentes $\big(\neg(A \wedge B)\big) \vDash\Dashv \big((\neg A) \vee (\neg B)\big)$.

Para probar que $\big(\neg(A \wedge B)\big) \nvDash \big((\neg A) \wedge (\neg B)\big)$ puedes construir una tabla de la misma manera. Pero sólo una línea de la tabla es necesaria para establecer que realmente existe una asignacion de verdad que satisface a $\big(\neg(A \wedge B)\big)$ y no a$\big((\neg A) \wedge (\neg B)\big)$.

Este método suele ser aplicable pero puede ser un procedimiento poco eficiente, pues para probar una fórmula con varios símbolos, la cantidad de renglones en la tabla crece demasiado. Pero haciendo observaciones puedes reducir el número de operaciones que debes hacer. Existen algunos valores de verdad para los símbolos que “dominan” los valores del resto de la fórmula. Como ejemplo de lo anterior, considera:

$$\begin{array}{l}\Big(\big(\boldsymbol{A} \vee (\boldsymbol{B} \wedge \boldsymbol{C})\big) \leftrightarrow \big((\boldsymbol{A} \vee \boldsymbol{B}) \wedge (\boldsymbol{A} \vee \boldsymbol{C})\big)\Big) \\ \quad V\ \ V \qquad\qquad\quad V \quad\ V\ \ V \quad\ \ V\ \ V\ \ V\end{array}$$

$F \quad F \quad F \quad F \qquad V \qquad F \quad F \quad F \quad F \quad F$

$F \quad V \quad V \quad V \quad V \qquad V \qquad F \quad V \quad V \quad V \quad F \quad V \quad V$

1.2.4. Tautologías

A continuación se presenta una lista de las tautologías comunes y útiles en demostraciones.

1. **Leyes asociativas y conmutativas para** $\wedge, \vee, \leftrightarrow$**.**

2. **Leyes distributivas:**

$$\big((\boldsymbol{A} \wedge (\boldsymbol{B} \vee \boldsymbol{C})) \leftrightarrow ((\boldsymbol{A} \wedge \boldsymbol{B}) \vee (\boldsymbol{A} \wedge \boldsymbol{C}))\big)$$
$$\big((\boldsymbol{A} \vee (\boldsymbol{B} \wedge \boldsymbol{C})) \leftrightarrow ((\boldsymbol{A} \vee \boldsymbol{B}) \wedge (\boldsymbol{A} \vee \boldsymbol{C}))\big)$$

3. **Negación:**

$$\big((\neg(\neg \boldsymbol{A})) \leftrightarrow \boldsymbol{A}\big)$$
$$\big((\neg(\boldsymbol{A} \to \boldsymbol{B})) \leftrightarrow (\boldsymbol{A} \wedge (\neg \boldsymbol{B}))\big)$$
$$\big((\neg(\boldsymbol{A} \leftrightarrow \boldsymbol{B})) \leftrightarrow ((\boldsymbol{A} \wedge (\neg \boldsymbol{B})) \vee ((\neg \boldsymbol{A}) \wedge \boldsymbol{B}))\big)$$

4. **Leyes de De Morgan:**

$$\big((\neg(\boldsymbol{A} \wedge \boldsymbol{B})) \leftrightarrow ((\neg \boldsymbol{A}) \vee (\neg \boldsymbol{B}))\big)$$
$$\big((\neg(\boldsymbol{A} \vee \boldsymbol{B})) \leftrightarrow ((\neg \boldsymbol{A}) \wedge (\neg \boldsymbol{B}))\big)$$

5. **Otras**
   Tercero excluido: $(\boldsymbol{A} \vee (\neg \boldsymbol{A}))$**.**
   Contradicción: $(\neg(\boldsymbol{A} \vee (\neg \boldsymbol{A})))$**.**
   Contraposición: $((\boldsymbol{A} \to \boldsymbol{B}) \leftrightarrow ((\neg \boldsymbol{B}) \to (\neg \boldsymbol{A})))$**.**
   Exportación: $(((\boldsymbol{A} \wedge \boldsymbol{B}) \to \boldsymbol{C}) \leftrightarrow (\boldsymbol{A} \to (\boldsymbol{B} \to \boldsymbol{C})))$**.**

1.3. Inducción y recursión

A continuación se presentaran dos métodos de construcción usados en la lógica y en varias áreas de la matemática que son de vital importancia por presentar una manera de obtener, en un principio, conjuntos que cumplen ciertas propiedades: inducción y recursión. Debes tener cuidado, pues aunque los nombres de estos métodos de construcción son conocidos fuera del contexto matemático, el significado aquí es distinto al usual.

1.3.1. Teoremas generales de inducción y recursión

Inducción

Hay un tipo especial de construcción que es usando en la lógica y en otras áreas de las matemáticas. Se puede querer construir un cierto subconjunto de un conjunto U, empezando con algunos elementos iniciales de U y aplicando ciertas operaciones una y otra vez. El conjunto buscado será el más pequeño que contenga a los elementos iniciales y se cerrará bajo las operaciones. Los elementos serán elementos de U que puedan ser construidos por los elementos iniciales aplicando las operaciones un número finito de veces.

En particular cuando U es el conjunto de las expresiones, los elementos iniciales son los símbolos de enunciado y las operaciones $\mathcal{E}_\neg$, $\mathcal{E}_\wedge$, etcétera. El conjunto resultante es el conjunto de las fórmulas. Se encontrarán, posteriormente, otros casos especiales y será de ayuda estudiar de manera abstracta esta construcción.

Para simplificar la discusión, considera un conjunto inicial $B \subseteq U$ y una clase $\mathcal{F}$ de funciones con sólo dos elementos f y g, donde:

$$f: U \times U \to U, \qquad g: U \to U.$$

Así, f es una operación binaria en U y g es una operación unaria (por el momento $\mathcal{F}$ está siendo considerada finita, pero esto no es necesario y se verá por qué el argumento puede ser extendido a una situación más general).

Si a y b pertenecen a B, entonces el conjunto C que se quiere construir tendrá, por ejemplo, los elementos:

$$b, f(b,b), g(a), f\big(g(a), f(b,b)\big), g\Big(f\big(g(a), f(b,b)\big)\Big).$$

Por supuesto, estos elementos pueden no ser distintos.

Al definir C más formalmente, se puede escoger entre dos definiciones como sigue:

1. De “arriba hacia abajo”: se dice que un subconjunto S de U es **cerrado** bajo f y g sii siempre que los elementos x, y pertenezcan a S, entonces también $f(x,y)$ y $g(x)$ pertenecen a S. El conjunto S **inductivo** sii $B \subseteq S$ y S es cerrado bajo f y g. Sea C^* la intersección de todos loa subconjuntos inductivos de U; así $x \in C^*$ sii x pertenece a todo

subconjunto inductivo de U. Se puede ver que C^* es inductivo, y que C^* es el conjunto inductivo más pequeño, pues está contenido en todos los demás conjuntos inductivos.

2. De "abajo hacia arriba": esta definición es equivalente a la primera. En este caso se quiere que los elementos de C_* sean todas las cosas que pueden ser alcanzadas desde B aplicando f y g un número finito de veces. Se define una **sucesión de construcción** como una sucesión finita $\langle x_0, \dots x_n \rangle$ de elementos de U tales que para cada $i \leq n$ se cumple por lo menos una de las siguientes condiciones:

$$x_i \in B,$$
$$x_i = f(x_j, x_k) \text{ para algunos } j < i, k < i$$
$$x_i = g(x_j) \text{ para algún } j < i.$$

En otras palabras, cada miembro de la sucesión está en B, o resulta de los elementos anteriores aplicando f o g.

Sean C_* el conjunto de todos los puntos x tales que alguna sucesión de construcción termina con x y C_n el conjunto de todos los puntos x tales que alguna sucesión de construcción de longitud n termina con x. Entonces $C_1 = B$,

$$C_1 \subseteq C_2 \subseteq C_3 \subseteq \cdots,$$

y $C_* = \cup_n C_n$. Por ejemplo, $g\Big(f\big(a, f(b,b)\big)\Big)$ pertenece a C_5 y por lo tanto a C_*, como puedes ver en el siguiente árbol:

$$g\Big(f\big(a, f(b,b)\big)\Big)$$
$$f\big(a, f(b,b)\big)$$
$$a, \qquad f(b,b)$$
$$b, \qquad b$$

Al aplanar este árbol hasta convertirlo en un orden lineal se obtiene una sucesión de construcción de $g\Big(f\big(a, f(b,b)\big)\Big)$.

Ejemplo

1. Los números naturales

Sea U el conjunto de todos los números reales y sea $B = \{0\}$. Considera una operación (sucesor) S, donde $S(x) = x + 1$. Entonces:

$$C_* = \{0,1,2,3, \dots\}$$

El conjunto C_* de los números naturales tiene como elementos exactamente aquellos números que se pueden obtener a partir de 0 aplicando repetidamente la operación sucesora.

2. Los enteros
Sean U el conjunto de todos los números reales y $B = \{0\}$. Esta vez considera las operaciones sucesoras S y la operación predecesor P:

$$S(x) = x + 1, \qquad P(x) = x - 1.$$

Ahora C_* es el conjunto de todos los enteros:

$$C_* = \{\dots, -2, -1,0,1,2, \dots\}$$

En este caso, observa que es posible obtener el 2 como elemento de C_*, ya que $S\big(S(0)\big) = 2 = S\left(P\left(S\big(S(0)\big)\right)\right)$.

3. Las fórmulas
Sea U el conjunto de todas las expresiones y B el conjunto de los símbolos de enunciado. Sea $\mathcal{F}$ el conjunto de las operaciones $\mathcal{E}_{\neg}, \mathcal{E}_{\wedge}, \mathcal{E}_{\vee}, \mathcal{E}_{\rightarrow}$ y $\mathcal{E}_{\leftrightarrow}$. Entonces C_* es el conjunto de todas las fórmulas.

Ahora se verificará que las dos definiciones son equivalentes, es decir que $C^* = C_*$.

Para empezar se va a demostrar que $C^* \subseteq C_*$ y para ello sólo se necesita verificar que C_* es inductivo, es decir, que $B \subseteq C_*$ y que C_* es cerrado bajo las funciones. Claramente, $B = C_1 \subseteq C_*$. Si x, y están en C_*, entonces es posible concatenar sus sucesiones de construcción y agregar al final $f(x, y)$ para obtener una sucesión de construcción que coloca a $f(x, y)$ en C_* y de la misma forma es cerrado bajo g.

Finalmente, para probar que $C_* \subseteq C^*$ considera un elemento de C_* y una sucesión de construcción $\langle x_o, \dots, x_n\rangle$ para él. Por inducción, como la que comúnmente has usado, sobre i, se puede ver que $x_i \in C^*$, si $i \leq n$. En primer lugar, $x_0 \in B \subseteq C^*$. Para el paso inductivo se usa el hecho de que C^* es cerrado bajo las funciones. De esta manera se tiene que:

$$\bigcup_n C_n = C_* = C^* = \bigcap \{S : S \text{ } es \text{ } inductivo\}$$

Como $C^* = C_*$, se denomina a este conjunto C simplemente, y se refiere a él como el conjunto **generado a partir de B por las funciones que pertenecen a $\mathcal{F}$**. A continuación se presenta un resultado muy utilizado para demostraciones de muchos teoremas:

1.3.1.1. Teorema [principio de inducción]

Sea C el conjunto generado a partir de B por las funciones que pertenecen a $\mathcal{F}$. Si S es un subconjunto de C que contiene a B y es cerrado bajo las funciones en $\mathcal{F}$, entonces $S = C$.

Demostración

S es inductivo, por lo que $C = C^* \subseteq S$. La otra contención se tiene en las hipótesis. ■

El caso espacial que es de interés por esta ocasión, es el que se presenta en el punto 4 del ejemplo, donde C es la clase de las fórmulas generadas a partir del conjunto de los símbolos de enunciado por las operaciones de construcción de fórmulas.

Recursión

Se presenta ahora de nuevo el caso abstracto, considera un conjunto U (como el conjunto de todas las expresiones), un subconjunto B de U (como el conjunto de los símbolos de enunciado) y dos funciones f y g, donde:

$$f: U \times U \to U, \qquad g: U \to U$$

C es el conjunto generado a partir de B por f y g.

Ahora se quiere definir recursivamente una función en C. Esto es, suponiendo que han sido dadas:

1. Reglas para calcular $\bar{h}(x)$ para $x \in B$.
2. Reglas para calcular:
    a. $\bar{h}\big(f(x,y)\big)$ usando $\bar{h}(x)$ y $\bar{h}(y)$.
    b. $\bar{h}\big(g(x)\big)$ usando $\bar{h}(x)$.

No es difícil ver que, a lo más, puede haber una función $\bar{h}$ en C que satisfaga todas las condiciones dadas.

Pero es posible que no exista tal función, por ejemplo, al usar reglas contradictorias. Para ver esto considera:

$$\begin{aligned}
&U = el\ conjunto\ de\ los\ \text{ú}meros\ reales,\\
&B = \{0\},\\
&f(x,y) = x \cdot y,\\
&g(x) = x + 1.
\end{aligned}$$

En este caso C es el conjunto de los números naturales. Supóngase que se imponen las siguientes condiciones a $\bar{h}$:

1. $\bar{h}(0) = 0$.

2.

a. $\bar{h}(f(x,y)) = f\left(\bar{h}(x), \bar{h}(y)\right)$,

b. $\bar{h}(g(x)) = \bar{h}(x) + 2$

Entonces no puede existir una tal función $\bar{h}$.

Se dice que C está **libremente generado** a partir de B por f y g sii además de las condiciones para que esté generado se tiene (considerando f_C, g_C las restricciones de f y g a C):

1. f_C y g_C son uno a uno
2. El rango de f_C, el rango de g_C y el conjunto B son ajenos dos a dos.

1.3.1.2. Teorema [de la reclusión]

Supóngase que el subconjunto C de U está libremente generado a partir de B por f y g, donde:

$$f: U \times U \to U, \qquad g: U \to U$$

Supóngase que V es un conjunto y F, G y h son funciones tales que:

$$h: B \to V$$
$$F: V \times V \to V$$
$$G: V \to V$$

Entonces existe una única función $\bar{h}: C \to V$ tal que:

i. $\forall x \in B, \bar{h}(x) = h(x)$,

ii. Para $x, y \in C$,

a. $\bar{h}(f(x,y)) = F\left(\bar{h}(x), \bar{h}(y)\right)$,

b. $\bar{h}(g(x)) = G\left(\bar{h}(x)\right)$.

Analizando esto desde el álgebra moderna, la conclusión del teorema dice que cualquier función h de B en V se puede extender a un homomorfismo $\bar{h}$ de C (con las operaciones f y g) en V (con las operaciones F y G).

Para comprender un poco más cómo funciona este teorema, considera nuevamente los ejemplos que se presentaron anteriormente para la inducción:

Ejemplo (condiciones en el teorema de recursión)

1. $B = \{0\}$ con la operación de sucesor S. Entonces C es el conjunto $\mathbb{N}$ de los números naturales. Como la operación sucesora es uno a uno, esto se presenta en los axiomas

de Peano, C es libremente generado a partir de $\{0\}$ por S. Por lo tanto, por el teorema de recursión, dado cualquier conjunto V, cualquier $a \in V$, y cualquier $F: V \to V$ existe una única $\bar{h}: \mathbb{N} \to V$ tal que $\bar{h}(0) = a$ y $\bar{h}\big(S(x)\big) = F\left(\bar{h}(x)\right)$ para cada $x \in \mathbb{N}$. Por ejemplo, existe una única $\bar{h}: \mathbb{N} \to \mathbb{N}$ tal que $\bar{h}(0) = 0$ y $\bar{h}\big(S(x)\big) = 1 - \bar{h}(x)$. Esta función toma el valor 0 en los números pares y el valor 1 en los números impares.

2. Los números enteros están generados a partir de $\{0\}$ por las operaciones de sucesor y predecesor, pero no libremente.

3. Las fórmulas están libremente generadas, a partir de los símbolos de enunciado, por las cinco operaciones de construcción de fórmulas; en el subtema se verá a fondo esta afirmación. Se sigue de esto, por ejemplo, que hay una única función $\bar{h}$ definida en el conjunto de las fórmulas tal que:
   $\bar{h}(A) = 1,$ si es un símbolo de enunciado,
   $\bar{h}\big((\neg\alpha)\big) = 3 + \bar{h}(\alpha),$
   $\bar{h}\big((\alpha \wedge \beta)\big) = 3 + \bar{h}(\alpha) + \bar{h}(\beta),$
   y similarmente para $\vee, \to$, y $\leftrightarrow$. Esta función da la longitud de cada fórmula.

Demostración del teorema de recursión. La idea es construir $\bar{h}$ como la unión de muchas funciones que aproximan cada vez más lo que se quiere lograr. Por el momento considera una función v que va de un subconjunto C en V, que sea **aceptable** si satisface las condiciones **i**. y **ii**. que se impusieron a $\bar{h}$. Más precisamente, v es aceptable sii el dominio de v es un subconjunto de C, su rango un subconjunto de V y

- i'. Si $x \in B \cap Dom\ v$, entonces $v(x) = h(x)$.
- ii'. Si $f(x,y) \in Dom\ v$, entonces x, y también pertenecen a $Dom\ v$, y $v\big(f(x,y)\big) = F\big(v(x), v(y)\big)$. Si $g(x) \in Dom\ v$, entonces también $x \in Dom\ v$, y $v\big(g(x)\big) = G\big(v(x)\big)$.

Sea K la colección de todas las funciones aceptables y $\bar{h}$ la unión de K. Por lo tanto:

$$\langle x, y\rangle \in \bar{h}\ sii\ v(x) = y\ para\ alguna\ v \in K.$$

A continuación se presentan algunas ideas de la demostración; debes resolver los detalles considerando ideas conjuntistas y de álgebra previas a este curso.

1. Se tiene que $\bar{h}$ es una función. Sea:

$$S = \left\{x \in C: \text{para a lo más una } y, \langle x, y\rangle \in \bar{h}\right\}$$
$$= \{x \in C: \text{todas las funciones aceptables definidas en } x \text{ coinciden}\}$$

Usando las condiciones **i’**. y **ii’**. puedes ver que el conjunto S es inductivo, por lo tanto $S = C$ y $\bar{h}$ es una función.

2. Se tiene también que $\bar{h} \in K$; es decir, que $\bar{h}$ es una función aceptable. Esto se sigue de la definición de $\bar{h}$ y el hecho de que es función.

3. Se tiene que $\bar{h}$ está definida en todo C. Basta probar que el dominio de $\bar{h}$ es inductivo. Esto se sigue de la hipótesis de que C es libremente generado. Por ejemplo, un caso es el siguiente: Supón que x está en el dominio de $\bar{h}$. Entonces $\langle g(x), G\left(\bar{h}(x)\right)\rangle$ es aceptable. Se requiere la libertad en la generación para probar que es una función. En consecuencia, $g(x)$ pertenece al dominio de $\bar{h}$.

4. Se afirma que $\bar{h}$ es única. Dadas dos funciones que cumplen las condiciones, sea S el conjunto en el que coinciden. Entonces S es inductivo, por lo tanto igual a C.

■

La demostración, si bien aquí se presenta a grandes rasgos, deja algunas ideas que permiten comprender un poco más las condiciones y conclusión del teorema. A manera de ampliar la visión del teorema, se presentará una manera de describir la demostración del teorema de recursión, presentando la función $\bar{h}$ como el conjunto de parejas generado a partir de un conjunto por ciertas operaciones. Sean:

$$\hat{U} = U \times V,$$
$$\hat{B} = h, \text{ un subconjunto de } \hat{U},$$
$$\hat{F}(\langle x,u\rangle, \langle y,v\rangle) = \langle f(x,y), F(u,v)\rangle,$$
$$\hat{G}(\langle x,u\rangle) = \langle g(x), G(u)\rangle.$$

De modo que $\hat{F}$ es la operación binaria en $\hat{U}$ que se obtiene como el producto cartesiano de las operaciones f y F. Sea $\bar{h}$ el subconjunto de $\hat{U}$ generado a partir de $\hat{B}$ por $\hat{F}$ y $\hat{G}$. Entonces se puede verificar que $\bar{h}$ tiene las propiedades que se requieren y es necesaria la libertad para probar que $\bar{h}$ es función.

1.3.2. Unicidad de lectura

En el subtema anterior se mostró, en particular en la demostración del teorema de recursión, que el conjunto de las fórmulas está libremente generado por las cinco operaciones a partir del conjunto de símbolos de enunciado; a continuación se probarán algunos resultados al respecto.

Para empezar, se va a demostrar que se han usado suficientes paréntesis para eliminar cualquier ambigüedad al analizar las fórmulas, esto considerando que la extención $\bar{v}$ de v depende de esta falta de ambigüedad. Considera qué puede pasar cuando no se introducen paréntesis, la ambigüedad que resulta se ilustra en la siguiente fórmula:

$$A_1 \vee A_2 \wedge A_3,$$

que se puede formular de dos maneras, correspondientes a $\big((A_1 \vee A_2) \wedge A_3\big)$ y a $\big(A_1 \vee (A_2 \wedge A_3)\big)$. Si $v(A_1) = V$ y $v(A_3) = F$, entonces hay un conflicto irresoluble que surge al tratar de calcular $\bar{v}(A_1 \vee A_2 \wedge A_3)$.

Se debe probar que con los paréntesis este tipo de ambigüedad no surge, al contrario, toda fórmula se forma de manera única. Hay un sentido en el que este hecho no tiene importancia: si fallara, simplemente se cambia de notacion hasta que fuera cierto. Aunque existen métodos más limpios en la construcción de las fórmulas que evitan la ambigüedad, se verá que no son necesarios pues basta con lo que se ha hecho. Para esto se presentan los siguientes lemas.

1.3.2.1. Lema
Toda fórmula tiene el mismo número de paréntesis izquierdos que de derechos.

La demostración de este lema se basa en lo visto en el subtema 1.1.

1.3.2.2. Lema
Cualquier segmento inicial propio de una fórmula tiene un exceso de paréntesis izquierdos; así ningún segmento inicial propio de una fórmula puede ser, a su vez, una fórmula.

Demostración
Se probará que el conjutno S de fórmulas, con la propiedad de que sus segmentos iniciales propios tienen más paréntesis izquierdos, es inductivo. Una fórmula que consista en un solo símbolo de enunciado no tiene segmentos iniciales propios, por lo tanto esta en S por vacuidad. Para verificar que S es cerrado bajo $\mathcal{E}_\wedge$, considera dos elementos α y β de S. Los segmentos iniciales propios son:

1. $($,
2. $(\alpha_0$, donde α_0 es un segmento inicial propio de α,
3. $(\alpha$,
4. $(\alpha \wedge$,
5. $(\alpha \wedge \beta_0$, donde β_0 es un segmento inicial propio de β,
6. $(\alpha \wedge \beta$.

Al aplicar la hipótesis inductiva de que α y β están en S, se tiene la conclusión.

■

1.3.2.3. Teorema [de unicidad de la lectura de fórmulas]
Las cinco operaciones de construcción de fórmulas, restringidas al conjunto de fórmulas:

a) tienen rangos disjuntos entre sí y disjuntos del conjunto de los símbolos de enunciado
b) son inyectivas.

(Es decir, como se mencionó, el conjunto de las fórmulas es libremente generado.)

Demostración

Para probar que la restricción de $\mathcal{E}_\wedge$ es inyectiva, supón que:

$$(\alpha \wedge \beta) = (\gamma \wedge \delta)$$

Borrando el primer símbolo se tiene que:

$$\alpha \wedge \beta) = \gamma \wedge \delta)$$

Entonces se debe tener que $\alpha = \gamma$, a menos que uno de ellos sea un segmento inicial propio del otro, lo que contradice al lema 1.3.2.2. Y entonces se sigue que $\beta = \delta$. De manera similar, este argumento se aplica a $\mathcal{E}_\vee, \mathcal{E}_\rightarrow$, $\mathcal{E}_\leftrightarrow$ y $\mathcal{E}_\neg$.

Para demostrar que las operaciones tienen rangos disjuntos considera, por ejemplo, si:

$$(\alpha \wedge \beta) = (\gamma \rightarrow \delta),$$

donde α, β, γ y δ son fórmulas, entonces, de forma similar a lo anterior, se tiene $\alpha = \gamma$. Pero esto implica que $\wedge = \rightarrow$, lo cual contradice lo que se planteó sobre los símbolos desde un inicio, siendo que éstos son distintos. Por lo tanto, $\mathcal{E}_\wedge$ y $\mathcal{E}_\rightarrow$ restringidas a las fórmulas tienen rangos disjuntos. De manera similar sucede para las otras parejas binarias.

Para los casos restantes considera lo siguiente: Si, por ejemplo, $(\neg\alpha) = (\beta \wedge \gamma)$, entonces β comienza con $\neg$, lo cual contradice que β es una fórmula. Además los rangos de las operaciones son disjuntos del conjunto de símbolos de enunciado, ya que ningún símbolo de enunciado es una sucesión de símbolos que comienza por $($.

■

Al regresar al problema de definir una extensión de una asignación de verdad v a $\bar{v}$, considera primero el caso en el que v es una asignación de verdad para el conjunto de todos los símbolos de enunciado. Entonces, aplicando los teoremas de unicidad de lectura y el de recursión presentados en el subtema 1.3.1, se concluye que existe una única extensión $\bar{v}$ al conjunto de todas las fórmulas con las propiedades que se requieren.

De manera general, considera una asignación de verdad v para un conjunto $\mathcal{F}$ de símbolos de enunciado. El conjunto $\bar{\mathcal{F}}$ generado a partir de $\mathcal{F}$ por las cinco operaciones de construcción de fórmulas está libremente generado, esto por el teorema de unicidad de lectura. Entonces, por el teorema de recursión, existe una única extensión $\bar{v}$ de v a ese conjunto con las propiedades que se piden.

Algoritmo

La demostración del teorema de unicidad de la lectura se puede convertir de una prueba en un algoritmo que, dada una fórmula, producirá su único árbol genealógico; pero si le asigna una expresión que no es fórmula detecta este hecho.

Si se da una expresión, inicialmente ella es el único vértice del árbol y por lo tanto el mínimo, pero mientras continúa el procedimiento, el árbol bajará a partir de la expresión dada.

1. Si todos los vértices minimales tienen símbolos de enunciado, entonces el procedimiento ha terminado. De otra forma, se escoge un vértice minimal que tenga alguna expresión que no sea un símbolo de enunciado.

2. El primer símbolo debe ser (, de no ser así no es una fórmula. Si el segundo símbolo es el símbolo de negación, se va al paso 4.

3. Se recorre la expresión desde la izquierda hasta llegar a $(\alpha$, donde α es una expresión con igual número de paréntesis izquierdos que de derechos, en el caso en que se llegue al final de la expresión antes de encontrar este α entonces la expresión original no era una fórmula. Entonces α es el primer componente. El siguiente símbolo debe ser $\wedge$,$\vee$, $\rightarrow$ o $\leftrightarrow$ y es la conectiva principal. El resto de la expresión, $\beta)$, debe consistir en una expresión β y un paréntesis derecho o de manera similar a como sucede en 2, no sería una fórmula. El segundo componente es β. Esto completa la descomposición de la expresión elegida; entonces se regresa al paso 1.

4. Si el segundo símbolo es el símbolo de negación, entonces ésa es la conectiva principal. El resto de la expresión, $\beta)$, debe consistir en una expresión β y un paréntesis derecho. β es el componente. Esto completa la descomposición de la expresión elegida; se regresa al paso 1.

Existen algunas observaciones que se deben hacer respecto del algoritmo:

a) Dada cualquier expresión, el procedimiento termina después de un número finito de pasos. Esto es porque cualquier vértice contiene una expresión más corta que la que está arriba de él, así la profundidad del árbol está acotada por la longitud de la expresión dada.

b) El procedimiento es único. Por ejemplo, en el paso 3 se llega a una expresión α, no es posible usar menos de α como componente, ya que no habría un balance entre paréntesis derechos e izquierdos; no se puede usar más de α porque esto contendría el segmento inicial propio α que estaba balanceado. Por lo tanto sólo es posible usar a α. Así, la elección de la conectiva principal es inevitable.

c) Si la expresión dada es tal que el procedimiento no la rechaza, entonces, recorriendo el árbol de abajo hacia arriba, se tiene inductivamente que todo vértice tiene una fórmula, incluyendo al vértice superior (que tiene la expresión dada).

d) También se puede usar el árbol para ver cómo se obtiene $\bar{v}(\alpha)$. Para cualquier fórmula α existe un único árbol que la construye. Recorriendo este árbol de abajo hacia arriba se llega sin ambigüedades al valor de $\bar{v}(\alpha)$.

En lo sucesivo, por notación, al nombrar fórmulas, no se mencionarán necesariamente todos los paréntesis. Para establecer una notación más compacta, se adoptan las siguientes convenciones:

i) El primer y último paréntesis no necesitan ser mencionados. Por ejemplo, cuando escribas “$A \wedge B$” te refieres a $(A \wedge B)$.

ii) El símbolo de negación se aplica a la menor cantidad de símbolos como sea posible. Por ejemplo, $\neg A \wedge B$ es $(\neg A) \wedge B$, de acuerdo con lo anterior esto es $\big((\neg A) \wedge B\big)$; que no es lo mismo que $\big(\neg(A \wedge B)\big)$.

iii) La conjunción y la disyunción se aplican a tan poco como sea posible, siempre y cuando se observe la condición 2. Por ejemplo:

$$A \wedge B \to \neg C \vee D \text{ es } \Big((A \wedge B) \to \big((\neg C) \vee D\big)\Big)$$

iv) Cuando se usa repetidamente un símbolo de conectiva, la agrupación es hacia la derecha:

$$\alpha \wedge \beta \wedge \gamma \; es \; \alpha \wedge (\beta \wedge \gamma)$$
$$\alpha \to \beta \to \gamma \; es \; \alpha \to (\beta \to \gamma).$$

Estas convenciones facilitan mucho la escritura de las fórmulas, porque ahora, en especial, no se necesita nombrar expresiones que no sean fórmulas.

1.4. Conectivos lógicos

Hasta ahora sólo se han usado cinco símbolos de conectivas para la construcción de las fórmulas. Aun en ausencia de una definición general de “conectiva”, está claro que las cinco conectivas familiares no son las únicas posibles. En este punto podrías preguntarte si se obtendría “algo” más si se aumenta el número de conectivas al lenguaje o, por el contrario, si podría perderse “algo” al omitir alguno de ellos. Ahora se hacen precisas estas preguntas y se dan algunas respuestas.

1.4.1. Funciones booleanas

Primero considera un ejemplo informal, para empezar agranda el lenguaje mediante un símbolo de conectiva de tres lugares, #, llamado el símbolo de mayoría. Ahora se permitirá que la expresión sea una fórmula cuando α, β y γ son fórmulas. En otras palabras, se agrega una sexta operación de construcción de fórmulas:

$$\mathcal{E}_{\#}(\alpha, \beta, \gamma) = (\#\alpha\beta\gamma).$$

Ahora se debe dar la interpretación de este símbolo. Esto es, se tiene que decir cómo calcular $\bar{v}\big((\#\alpha\beta\gamma)\big)$, dados los valores $\bar{v}(\alpha), \bar{v}(\beta)$ y $\bar{v}(\gamma)$. Se elige la definición siguiente:

$\bar{v}\big((\#\alpha\beta\gamma)\big),$ coincide con la mayoría de $\bar{v}(\alpha), \bar{v}(\beta), \bar{v}(\gamma)$

Lo que aquí se afirma es que esta extensión no ha aportado algo, en el siguiente sentido preciso: para toda fórmula del lenguaje extendido existe en el lenguaje original, una fórmula tautológicamente equivalente. (Por otro lado, la fórmula del lenguaje original puede ser mucho más larga que la del lenguaje extendido.) Se probará esto (en una situación mucho más general) más adelante; aquí sólo se hará notar que es tautológicamente equivalente a…

$$(\alpha \wedge \beta) \vee (\alpha \wedge \gamma) \vee (\beta \wedge \gamma).$$

A manera de argumento, recuerda que el valor de $\bar{v}\big((\#\alpha\beta\gamma)\big)$ debe ser calculable a partir de $\langle \bar{v}(\alpha), \bar{v}(\beta), \bar{v}(\gamma) \rangle$, ya que esto juega un papel definitivo. En el lenguaje coloquial hay operadores unarios como "es posible que" o "creo que". Se puede aplicar uno de estos operadores a un enunciado, produciendo otro enunciado cuya verdad o falsedad no se puede determinar solamente con base en la verdad o falsedad del enunciado original.

Al generalizar el ejemplo anterior, el lenguaje formal será más un obstáculo que una ayuda. Es posible reformular el problema al usar sólo funciones.

1.4.1.1. Definición

Una **función booleana de k variables** es una función de $\{V, F\}^k$ en $\{V, F\}$.

Una **función booleana** es, entonces, cualquier función booleana de k variables para alguna k. Esto se amplía un poco permitiendo que V y F sean funciones booleanas de cero variables. Como ejemplos de funciones booleanas se tienen las definidas por las ecuaciones siguientes (donde $X \in \{V, F\}$):

$$I_i^n(X_1, \dots, X_n) = X_i$$
$$N(V) = F, \qquad N(F) = V,$$
$$K(V, V) = V, \qquad K(F, X) = K(X, F) = F,$$

$$A(F,F)=F, \qquad A(V,X)=A(X,V)=V,$$
$$C(V,F)=F, \qquad C(F,X)=C(X,V)=V,$$
$$E(X,X)=V, \qquad E(V,F)=E(F,V)=F.$$

A partir de una fórmula α se puede obtener una función booleana. Por ejemplo, si α es la fórmula $A_1 \wedge A_2$, entonces puedes hacer una tabla (siguiente). Los 2^2 renglones de la tabla corresponden a las 2^2 asignaciones de verdad para $\{A_1, A_2\}$. Para cada una de las 2^2 parejas $\bar{X}$, se hace $B_\alpha(\bar{X})$ igual al valor que α recibe cuando se les dan a sus símbolos de enunciado los valores indicados por $\bar{X}$.

Tabla

A_1	A_2	$A_1 \wedge A_2$	
F	F	F	$B_\alpha(F,F)=F$
F	V	F	$B_\alpha(F,V)=F$
V	F	F	$B_\alpha(V,F)=F$
V	V	V	$B_\alpha(V,V)=V$

En general, supondrás que α es una fórmula cuyos símbolos de enunciado están entre $A_1, \ldots, A_n$.

1.4.1.2. Definición

Se define una función booleana de n variables B_α^n, la función booleana realizada por α, como

$$B_\alpha^n(X_1, \ldots, X_n) = el\ valor\ que\ recibe\ \alpha\ cuando\ a\ A_1, \ldots, A_n\ les\ son\ asignados\ los\ valores X_1, \ldots, X_n.$$

En otras palabras: $B_\alpha^n(X_1, \ldots, X_n) = \bar{v}(\alpha)$, donde v es la asignación de verdad para $\{A_1, \ldots, A_n\}$, para la cual $v(A_i) = X_i$. Así B_α^n surge de considerar $\bar{v}(\alpha)$ como una función de v, con α fija.

Por ejemplo, las funciones booleanas mencionadas se pueden obtener de esta manera:

$$I_i^n = B_{A_i}^n,$$
$$N = B_{\neg A_1}^1,$$
$$K = B_{A_1 \wedge A_2}^2,$$
$$A = B_{A_1 \vee A_2}^2,$$
$$C = B_{A_1 \rightarrow A_2}^2,$$
$$E = B_{A_1 \leftrightarrow A_2}^2.$$

A partir de estas funciones se pueden componer otras. Por ejemplo:

$$B_{\neg A_1 \vee \neg A_2}^2(X_1, X_2) = A\left(N\left(I_1^2(X_1, X_2)\right), N\left(I_2^2(X_1, X_2)\right)\right)$$

En resultados siguientes se llegará a la cuestión de si toda función booleana se puede obtener de esta manera.

Como afirma el teorema siguiente, al cambiar la atención de las fórmulas a las funciones booleanas que realizan, quedan identificadas entre sí las fórmulas tautológicamente equivalentes. Ordenando el conjunto $\{V, F\}$ definiendo $F < V$. (Usualmente se considera $V = 1$ y $F = 0$, entonces éste es el orden natural).

1.4.1.1. Teorema

Sean α, β fórmulas cuyos símbolos de enunciado están entre $A_1, \dots, A_n$. Entonces:

a) $\alpha \vDash \beta$ ssi para toda $\vec{X} \in \{T, F\}^n, B_\alpha(\vec{X}) < B_\beta(\vec{X})$.
b) $\alpha \vDash\!\!\!\dashv \beta$ ssi $B_\alpha = B_\beta$.
c) $\vDash \alpha$ ssi B_α es la función constante con valor V.

Demostración

De (a)

$\alpha \vDash \beta$ ssi para todas las asignaciones de verdad v para $A_1, . . , A_n$, tales que $\bar{v}(\alpha) = V$, también $\bar{v}(\beta) = V$. (Esto es cierto aún si los símbolos de enunciado de α y β no incluyen a todos los símbolos $A_1, \dots, A_n$, considera hacer esto como ejercicio.) Por lo tanto:

$\alpha \vDash \beta$ ssi para todas las asignaciones v, $\bar{v}(\alpha) = V \Rightarrow \bar{v}(\beta) = V$,
ssi para todas las n-adas $\vec{X}$, $B_\alpha^n(\vec{X}) = V \Rightarrow B_\beta^n(\vec{X}) = V$,
ssi para todas las n-adas $\vec{X}$, $B_\alpha^n(\vec{X}) \leq B_\beta^n(\vec{X})$
donde $F < V$.

∎

Además de identificar fórmulas tautológicamente equivalentes, te has librado del lenguaje formal. Ahora existe cierta libertad de considerar cualquier función booleana, sea o no realizada por alguna fórmula. Pero esta libertad es sólo aparente:

1.4.1.2. Teorema

Sea G una función booleana de n variables, con $n \geq 1$. Es posible encontrar una fórmula α tal que $G = B_\alpha^n$, es decir, dicha α realiza la función G.

Demostración

Caso I: si G es la función constante con valor F, considera $\alpha = A_1 \wedge \neg A_1$.
Caso II: de otra forma existen k puntos en los cuales G tiene el valor V, con $k > 0$. Sea:

$$\begin{aligned}
\vec{X}_1 &= \langle X_{11}, X_{12}, \dots, X_{1n} \rangle, \\
\vec{X}_2 &= \langle X_{21}, X_{22}, \dots, X_{2n} \rangle, \\
&\dots \\
\vec{X}_k &= \langle X_{k1}, X_{k2}, \dots, X_{kn} \rangle.
\end{aligned}$$

una lista de dichos puntos.

Sean:

$$\beta_{ij} = \begin{cases} A_j, & si\ X_{ij} = V, \\ (\neg A_j), & si\ X_{ij} = F, \end{cases}$$
$$\gamma_i = \beta_{i1} \wedge \cdots \wedge \beta_{in},$$
$$\alpha = \gamma_1 \vee \gamma_2 \vee \cdots \vee \gamma_k$$

Se afirma que $G = B_\alpha^n$.

Para comprender un poco mejor la razón de estas construcciones, considera el siguiente ejemplo. Sea G la siguiente función booleana de tres variables.

$$G(F,F,F) = F,$$
$$G(F,F,V) = V,$$
$$G(F,V,F) = V,$$
$$G(F,V,V) = F,$$
$$G(V,F,F) = F,$$
$$G(V,F,V) = F,$$
$$G(V,V,F) = F,$$
$$G(V,V,V) = V.$$

Entonces, la lista de ternas en las cuales G toma el valor F tiene cuatro elementos:

$$FFV \quad \neg A_1 \wedge \neg A_2 \wedge A_3,$$
$$FVF \quad \neg A_1 \wedge A_2 \wedge \neg A_3,$$
$$VFF \quad A_1 \wedge \neg A_2 \wedge \neg A_3,$$
$$VVV \quad A_1 \wedge A_2 \wedge A_3.$$

A la derecha de cada terna se han escrito la conjunción correspondiente γ_i. Entonces α es la fórmula:

$$(\neg A_1 \wedge \neg A_2 \wedge A_3) \vee (\neg A_1 \wedge A_2 \wedge \neg A_3) \vee (A_1 \wedge \neg A_2 \wedge \neg A_3) \vee (A_1 \wedge A_2 \wedge A_3).$$

Se tiene que α da una lista explícita de las ternas en las cuales G toma el valor V.

Al regresar a la demostración, observa que $B_\alpha^n(\vec{X}_i) = V$ para $1 < i < k$; esto porque la asignación de verdad correspondiente a $\vec{X}_i$ satisface γ_i, por lo tanto satisface a α. Por otra parte, sólo una asignación de verdad para $\{A_1, .., A_n\}$ satisface γ_i, de donde sólo k asignaciones pueden satisfacer a α. Por lo tanto $B_\alpha^n(\vec{Y}) = F$ para las demás $2^n - k$ n-adas Y. Así para todos los casos se tiene $B_\alpha^n(\vec{Y}) = G(Y)$.

■

En el teorema anterior se ha visto que cualquier función booleana G es realizable, pero esta realización no es necesariamente única, pues cualquier fórmula tautológicamente equivalente también realiza a G, para el ejemplo dentro de la demostración: $A_1 \leftrightarrow A_2 \leftrightarrow A_3$ realiza a G.

Como corolario del teorema anterior, se puede concluir que se tienen suficientes conectivas. En efecto, si se extiende el lenguaje agregándole algunas conectivas nuevas, cualquier fórmula φ de este lenguaje extendido realiza alguna función B_φ^n. Por el teorema anterior, se tiene una fórmula α del lenguaje original tal que $B_\varphi^n = B_\alpha^n$. Por tanto φ y α son tautológicamente equivalentes por el teorema 1.4.1.1.

La demostración del teorema prueba que la forma de α incluye sólo las conectivas ∧,∨ y ¬; ésta es la llamada **forma normal disyuntiva.** Esto es:

$$\alpha = \gamma_1 \vee \gamma_2 \vee \cdots \vee \gamma_k$$

donde:

$$\gamma_i = \beta_{i1} \wedge \cdots \wedge \beta_{in}$$

Cada β_{ij} es un símbolo de enunciado o la negación de un símbolo de enunciado. Así se tiene el siguiente corolario.

1.4.1.3. Corolario

Para cualquier fórmula φ, es posible encontrar una fórmula tautológicamente equivalente a α en forma normal disyuntiva.

Para cada n existen 2^{2^n} funciones booleanas de n variables. Por lo tanto, si se identifica cada conectiva con su función booleana, se tiene que hay 2^{2^n} conectivas n-arias. Ahora se presenta una clasificacion con $n \leq 2$.

Conectivos 0-arias

Existen dos funciones de 0 variables, V y F. Como símbolos de conectiva correspondientes se han tomado ⊤ y ⊥. Ahora una conectiva n-aria se combina con n fórmulas para producir una nueva fórmula. Cuando $n = 0$, se tiene que ⊥ es por sí mismo una fórmula. Difiere de los símbolos de enunciado en que $\bar{v}(\bot) = F$ para toda v, es decir, ⊥ es un símbolo lógico al que siempre se le asigna el valor de F. Similarmente ⊤ es una fórmula y $\bar{v}(\top) = V$ para toda v. Por ejemplo, $A \rightarrow \bot$ e suma fórmula, tautologicamente equivalente a $\neg A$.

Conectivas unarias

Hay cuatro conectivas unarias: la negación, la identidad y las dos funciones constantes.

Conectivas binarias

Hay 16 conectivas binarias, pero sólo las últimas 10 que aparecen en la tabla siguiente son “binarias”.

Símbolo	Equivalentes	Observaciones
	$\top$	Constante de dos variables, escencialmente 0-aria
	$\bot$	Constante de dos variables, escencialmente 0-aria
	A	Proyección, esencialmente unaria
	B	Proyección, esencialmente unaria
	$\neg A$	Negación, esencialmente unaria
	$\neg B$	Negación, esencialmente unaria
$\wedge$	$A \wedge B$	Si $V = 1$ y $F = 0$, entonces ésta es la multiplicación en el campo $\{0,1\}$
$\vee$	$A \vee B$	O
$\rightarrow$	$A \rightarrow B$	Condicional
$\leftrightarrow$	$A \leftrightarrow B$	Bicondacional
$\leftarrow$	$B \leftarrow A$	Condicional invertido
$+$	$(A \vee B) \wedge \neg(A \wedge B)$	O exclusivo
$\downarrow$	$\neg(A \vee B)$	Nor, no A ni B
$\vert$	$\neg(A \wedge B)$	Nand, el símbolo se llama la raya de Sheffer
$<$	$(\neg A) \wedge B$	El orden usual
$>$	$A \wedge (\neg B)$	El orden usual

Conectivas ternarias

Hay 256 conectivas ternarias: 2 esencialmente 0-arias, 6 son esencialmente unarias, 30 son esencialmente binarias y 218 son realmente ternarias. Hasta ahora se ha trabajado con la conectiva mayoría #.

1.4.2. Conjuntos de conectivos completos

De acuerdo con los teoremas del subtema anterior, dado que toda funcion $G: \{\wedge, \vee, \neg\}^n \rightarrow \{V, F\}$ para cada $n \geq 1$ puede ser realizada por una fórmula que usa sólo los símbolos de conectiva del conjutno $\{\wedge, \vee, \neg\}$, se dice que el conjunto $\{\wedge, \vee, \neg\}$ es **completo**. Ser completo es en realidad una propiedad de las funciones booleanas asociadas a esos símbolos.

Los siguientes teoremas exponen conjuntos de conectivos que son completos, es decir, para cualquier fórmula φ es posible construir una fórmula que use sólo las conectivas del conjunto y que realice B_φ, entonces $\alpha \vDash\dashv \varphi$. La completud de $\{\wedge, \vee, \neg\}$ se puede mejorar:

1.4.2.1. Teorema

$\{\wedge, \neg\}$ y $\{\vee, \neg\}$ son ambos completos.

Demostración

Se debe demostrar que cualquier función booleana G puede ser realizada por una fórmula que use únicamente, en el primer caso, $\{\wedge, \neg\}$. Sea α que use $\{\wedge, \vee, \neg\}$ y realice G, basta encontrar una fórmula tautólogicamente equivalente α' que use solamente $\{\wedge, \neg\}$. De acuerdo con la ley de De Morgan que está en la lista de tautologias anteriormente:

$$\beta \vee \gamma \vDash\Dashv \neg(\neg\beta \wedge \neg\gamma).$$

Aplicando esta ley repetidamente, es posible eliminar por completo $\vee$ de α.

De manera formal, es posible demostrar por induccion sobre α que existe una α' tautológicamente equivalente en la que sólo ocurren las conectivas $\wedge, \neg$. Los dos casos no triviales en el paso inductivo son:

Caso $\neg$: Si $\alpha = (\neg\beta)$, considera $\alpha' = (\neg\beta')$.
Caso $\vee$: Si $\alpha = (\beta \vee \gamma)$, se hace $\alpha' = \neg(\neg\beta' \wedge \neg\gamma')$.

Como β' y γ' son tautológicamente equivalentes a β y γ respectivamente, entonces:

$$\begin{aligned} \alpha' &= \neg(\neg\beta' \wedge \neg\gamma') \\ &\vDash\Dashv \neg(\neg\beta \wedge \neg\gamma) \\ &\vDash\Dashv \beta \vee \gamma \\ &= \alpha. \end{aligned}$$

■

1.4.2.2. Teorema
Los conjuntos siguientes son completos:

a) $\{|\}$
b) $\{\downarrow\}$
c) $\{\neg, \rightarrow\}$
d) $\{\bot, \rightarrow\}$

Demostración
Se presenta la demostración de {| }; de manera similar, se procede para el resto de los conjuntos y puedes hacerlo como ejercicio:

$$\neg\alpha \vDash\Dashv \alpha|\alpha$$
$$\alpha \vee \beta \vDash\Dashv (\neg\alpha)|(\neg\beta)$$

Como {¬,∨} es completo y ¬, ∨ se pueden simular usando solamente |, entonces {| } es completo.

■

Con respecto al inciso c), se puede notar que de las 10 conectivas realmente binarias que se mostraron en la tabla que aparece en el tema anterior, 8 de ellas forman un conjunto completo al agregarle el símbolo ¬, las que no cumplen esto son $+$ y $\leftrightarrow$.

1.4.3. Conjuntos de conectivos incompletos

Demostrar que un conjunto dado de conectivos no es completo es, generalmente, más díficil que uno que sí es completo. El método consiste primero en demostrar, generalmente por inducción, que para toda fórmula α que use sólo esas conectivas la función B_α^n tiene una "peculiaridad", y después probar que alguna función booleana carece de dicha "peculiaridad".

Considera el siguiente ejemplo para entender cómo demostrar ese tipo de "peculiaridad".

1.4.3.1. Proposición
$\{\wedge, \rightarrow\}$ no es completo.

Demostración
Con estas conectivas, si se asigna V a los símbolos de enunciado, entonces toda fórmula recibe el valor V. En particular, no se tiene nada tautológicamente equivalente a $\neg A$.

De manera detallada, es posible probar por induccion que para cualquier fórmula α que use sólo estas conectivas y que tenga A como único símbolo de enunciado, se tiene que $A \vDash \alpha$, en términos de funciones se tiene que $X \leq B_\alpha^1(X)$.

Para reforzar los temas anteriores, resuelve la siguiente actividad.

1.5. Compacidad y efectividad.

Para empezar, se dará la prueba del teorema 1.2.2.1., que se presentó anteriormente, pues ya se tienen las herramientas necesarias para esto y además se plantea un aspecto muy interesante respecto al método que se dio para evaluar una fórmula, el de las tablas de verdad; ya que éste existe, vale la pena preguntar si se podría tener un método más eficaz, es decir, que se lleve a cabo con el uso de menos "recursos".

1.5.1. Teorema de compacidad

Se dirá satisfactible a un conjunto Σ de fórmulas ssi existe una asignación de verdad que satisface a todos los elementos de Σ. Así es posible replantear el teorema 1.2.2.1 de la siguiente forma.

1.5.1.1. Teorema [teorema de compacidad]
Un conjunto de fórmulas es satisfactible ssi todos sus subconjuntos finitos son satisfactibles.

Por el momento, considera que Σ es **finitamente satisfactible** ssi todo subconjunto finito de Σ es satisfactible. El teorema de compacidad dice que esta noción coincide con la de satisfactibilidad. Las primeras observaciones que se hacen con respecto al teorema son:

a) Si Σ es satisfactible, entonces automáticamente es finitamente satisfactible.
b) Si Σ es finito, la conversa es trivial.
c) La parte no trivial consiste en probar que si un conjunto infinito es finitamente satisfactible, entonces es satisfactible.

Demostración del teorema de compacidad

La demostración se hará en dos partes. En la primera, se toma el conjunto finitamente satisfactible Σ y se extenderá a un conjunto finitamente satisfactible maximal Δ. Para la segunda, se usa Δ para construir una asignación de verdad que satisfaga a Σ.

Parte 1

Sea $\alpha_1, \alpha_2, \ldots$ una enumeración fija de las fórmulas, esto es posible porque el conjunto de los símbolos de enunciado, y por lo tanto las expresiones, es numerable. Por recursión sobre los números naturales se define:

$$\Delta_0 = \Sigma$$
$$\Delta_{n+1} = \begin{cases} \Delta_n; \alpha_{n+1}, & \text{Si esto es finitamente satisfactible} \\ \Delta_n; \neg\alpha_{n+1}, & \text{En el otro caso} \end{cases}$$

Donde se tiene que $\Delta_n; \alpha_{n+1} = \Delta_n \cup \{\alpha_{n+1}\}$. Por tanto, cada Δ_n es finitamente satisfactible, esto puedes verificarlo. Sea $\Delta = \bigcup_n \Delta_n$, el límite de las Δ_n.

Se cumplen, así, las siguientes propiedades:

1) $\Sigma \subseteq \Delta$
2) Para toda fórmula α, se tiene que $\alpha \in \Delta$ o $\neg\alpha \in \Delta$.
3) Δ es finitamente satisfactible, pues todo subconjunto finito de Δ es subconjunto finito de alguna Δ_n y es, por tanto, satisfactible.

Esto completa la primera parte que se dijo en la demostración. El conjunto que se creó en general no es único, pero existe al menos uno.

Parte 2

Para esta parte se va a definir una asignación de verdad v para el conjunto de todos los símbolos de enunciado:

$$v(A) = V, \qquad ssi\ A \in \Delta,$$

para cualquier símbolo de enunciado A. Entonces se tiene que para toda fórmula

$$v \text{ satisface a } \varphi, \qquad ssi\ \varphi \in \Delta.$$

Esto se demuestra por inducción sobre φ. Como $\Sigma \subseteq \Delta$, v debe entonces satisfacer a todos los elementos de Σ.

■

1.5.1.2. Corolario

Si $\Sigma \vDash \tau$, entonces existe un subconjunto finito $\Sigma_0 \subseteq \Sigma$ tal que $\Sigma_0 \vDash \tau$.

Demostración

Se hace uso del hecho de que $\Sigma \vDash \tau$ ssi $\Sigma; \neg\tau$ es insatisfactible.

$$\begin{aligned}
&\Sigma_0 \nvDash \tau \text{ para cualquier conjunto finito } \Sigma_0 \subseteq \Sigma\\
&\Rightarrow \Sigma_0; \neg\tau \text{ es satisfactible para cualquier } \Sigma_0 \subseteq \Sigma \text{ finito}\\
&\Rightarrow \Sigma; \neg\tau \text{ es finitamente satisfactible}\\
&\Rightarrow \Sigma; \neg\tau \text{ es satisfactible}\\
&\Rightarrow \Sigma \nvDash \tau
\end{aligned}$$

■

De hecho, se tiene que el corolario anterior es equivalente al teorema de compacidad, esto se deja como ejercicio en la actividad 3.

1.5.2. Efectividad y calculabilidad

Ya se dijo que la existencia del método de las tablas de verdad tiene conclusiones teóricas interesantes. Para empezar, considera que se tiene un conjunto Σ de fórmulas y el problema es saber si existe un procedimiento **efectivo** que, dada una fórmula τ, decida sí o no $\Sigma \vDash \tau$. Por supuesto, hay que decir de manera sólo un poco más estructurada qué significa que algo sea efectivo, así un procedimiento es **efectivo** cuando satisface las siguientes condiciones:

1. Se deben tener instrucciones exactas (es decir, un programa), de longitud finita, que expliquen cómo ejecutar el procedimiento. Éstas no deben exigir ningún ingenio de parte de la persona (o máquina) que haya de seguirlas. La idea es que se pueda ejecutar el procedimiento siguiendo mecánicamente las instrucciones.

2. Deben ser instrucciones que puedan ser mecánicamente implementadas. El procedimiento debe evitar medios aleatorios, o cualquier medio tal que, en la práctica sólo se puede aproximar.

3. En el caso de un procedimiento de decisión, como el mencionado en el problema que se está considerando, dada una fórmula τ, el procedimiento debe producir una respuesta de sí o no después de un número finito de pasos (esto es, el procedimiento debe ser un algoritmo para determinar la respuesta).

Por otro lado, no hay cotas sobre el tiempo o el número de pasos que se requiere para que se llegue a la respuesta, mientras éstos sean finitos. Éstas dependerán, entre otras cosas, de la entrada τ. No es permitido hacer un número infinito de pasos y después dar la respuesta.

Para las personas enfocadas en la computación, o en general trabajos con máquinas, la efectividad se alcanza cuando es posible tener la respuesta en un periodo “razonable” de tiempo. Donde la idea de razonable depende de las situaciones.

La definición que se presentó difícilmente puede ser precisa, en lo siguiente se usará la palabra efectiva en un sentido intuitivo e informal. En lo siguiente se trata de trabajar con la idea de efectivo, como ya se dijo, desde un punto de vista empírico, de que cuando un procedimiento parece efectivo para “algunos” en general lo es. Las ideas intuitivas funcionan en buena manera porque se considera que el proceso es efectivo. Debido a que se está tomando el concepto de manera informal, las definiciones y teoremas que la usen se señalan con el siguiente teorema*.

***1.5.2.1. Teorema**
Existe un procedimietno efectivo que, dada cualqueir expresión α, decide si es una fórmula.

Demostración
Está basada en el algoritmo presentado cuando se muestra la unicidad de lectura y las observaciones en éste.

■

1.5.3. Decidibilidad y ejemplos de enunciados

Del último teorema del subtema anterior y su demostración puedes notar que se requería que una expresión dada fuera sometida al algoritmo; sin embargo, dado que se tiene un lenguaje con una cantidad infinita de símbolos, podrías preguntar la forma en que se tiene “dada” la fórmula ε. Para empezar, podrías imaginar que esto es porque alguien pudo escribir los símbolos de ε, uno después del otro. Es imposible que una persona escriba una cantidad infinita de símbolos. Para evitarlo, se considera el siguiente “formato de entrada y salida”. En lugar de escribir por ejemplo A_5, se usa A'''', una cadena de cinco símbolos. Así, el número total de símbolos en el alfabeto es sólo 9:

$$(,), \neg, \wedge, \vee, \rightarrow, \leftrightarrow, A, '$$

(Si se identifican estos nueve símbolos con los dígitos 1-9, se obtienen expresiones que parecen particularmente familiarizados en los entornos computacionales. Puedes tomar el dígito 0 para las expresiones de separación.)

El siguiente teorema establece que el conjunto de las fórmulas es **decidible** en el sentido de la definición a continuación.

***1.5.3.1. Definición [decidible]**
Un conjunto Σ de expresiones es **decidible** si y sólo si existe un procedimiento eficaz que, dada una expresión α, decidirá si $\alpha \in \Sigma$ o $\alpha \notin \Sigma$.

Por ejemplo, cualquier conjunto finito es decidible. (Las instrucciones pueden simplemente listar los elementos del conjunto. Entonces el algoritmo puede comprobar la entrada contra cada miembro de la lista). Algunos conjuntos infinitos son decidibles, pero no todos. Por un lado, hay conjuntos de expresiones no mumerables ($2^{\aleph_0}$); por otro, sólo puede haber una cantidad numerable de procedimientos eficaces. Esto es porque el procedimiento está completamente determinado por sus intrucciones, las cuales son sólo un número finito. Sólo hay $\aleph_0$ sucesiones finitas de letras y las instrucciónes, cuando se escriben hacia fuera, forman una sucesión finita de letras.

***1.5.3.1. Teorema**
Hay un procedimiento efectivo que, dado un conjunto $\Sigma;\tau$ de fórmulas, decidirá si $\Sigma \vDash \tau$ o no.

Demostración
El procedimiento de tablas de verdad expuesto cumple con estos requerimientos. ■

En este teorema se especifica la finitud de $\Sigma;\tau,$ ya que no se puede tener "dado" de manera directa o efectiva la totalidad de un objeto infinito.

***1.5.3.2. Corolario**
Dado un conjunto finito Σ, el conjunto de las consecuencias tautológicas de Σ es decidible. En particular, el conjunto de las tautologías es decidible.

Si Σ es un conjunto infinito decidible, entonces, por lo general, el conjunto de sus consecuencias tautológicas no es decidible, esto se verá en temas más adelante. Pero es posible obtener un resultado más débil, que en cierto sentido da la mitad de la decidibilidad.

1.5.3.2. Definición [efectivamente enumerable]
Se llamará a un conjunto A de expresiones **efectivamente enumerable** ssi existe algún procedimiento efectivo que produce una lista, en algún orden, de los elementos de A. (Si A es infinito, entonces el procedimiento nunca termina. Pero cualquier elemento particular de A debe, después de un tiempo finito, aparecer en la lista).

A continuación se presentan un par de resultados con respecto a esta definición para tenerla un poco más clara.

***1.5.3.3. Teorema**
Un conjunto A de expresiones es efectivamente enumerable ssi existe un procedimiento efectivo que, dada cualquier expresión ε, produce la respuesta "sí" en el caso de que $\varepsilon \in A$.

(Si $\varepsilon \notin A$ procedimiento puede producir la respuesta “no”, o puede continuar indefinidamente sin producir ninguna, pero no debe originar un “sí”.)

Demostración

Si A es efectivamente enumerable, entonces dada cualquier ε es posible examinar la lista de los elementos de A mientras el procedimiento la produce. Y cuando aparezca, se da la respuesta “sí”. (Así si $\varepsilon \notin A$, no se da ninguna respuesta. Esto es lo que hace que A no sea decidible. Si ε no ha ocurrido entre los primeros 10^{10} elementos enumerados de A, en general no hay manera de saber si $\varepsilon \notin A$, en cuyo caso se debería dejar de buscarlo, o si aparecerá justamente en el siguiente paso).

Conversamente, supón que se tiene el procedimiento descrito en el teorema. Se quiere crear una lista de los elementos de A. La idea es enumerar todas las expresiones y aplicar el procedimiento dado a cada una. Pero se debe administrar nuestro tiempo de manera razonable. Es bastante fácil enumerar efectivamente todas las expresiones:

$$\varepsilon_1, \varepsilon_2, \varepsilon_3, \ldots$$

Se procede así:

1. Dedicar un minuto a verificar si ε_1 es elemento de A (usando el procedimiento dado).
2. Dedicar dos minutos a verificar ε_1, luego dos minutos a ε_2.
3. Dedicar tres minutos a ε_1, luego tres minutos a ε_2 y tres minutos a ε_3.

Etcétera. Por supuesto que siempre que el procedimiento produzca un “sí” se coloca la expresión aceptada en la lista de salida. De esta manera para todo elemento de A llegará un momento en que aparecerá en la lista. (Aparecerá una infinidad de veces a menos que se modifiquen las instrucciones anteriores para evitar la duplicación.) ■

***1.5.3.4. Teorema**

Un conjunto de expresiones es decidible ssi tanto él como su complemento.(con respecto al conjunto de todas las expresiones) son efectivamente enumerables.

La demostración de este teorema se deja como ejercicio en la evidencia de aprendizaje.

Nota que si los conjuntos A y B son efectivamente enumerables, entonces, asimismo, lo son $A \cup B$ y $A \cap B$. La clase de los conjuntos decidibles también es cerrada bajo unión e intersección y además es cerrada bajo complementación.

Ahora un resultado más substancial:

***1.5.3.5. Teorema**

Si Σ es un conjunto decidible de fórmulas, entonces el conjunto de las consecuencias tautológicas de Σ es efectivamente enumerable.

Demostración

De hecho basta con que Σ esté efectivamente enumerado; considera una enumeración:

$$\sigma_1, \sigma_2, \sigma_3, \ldots$$

Dada cualquier fórmula τ, es posible verificar (por medio de tablas de verdad) si es que:

$$\begin{aligned} \emptyset &\vDash \tau \\ \{\sigma_1\} &\vDash \tau \\ \{\sigma_1, \sigma_2\} &\vDash \tau \\ \{\sigma_1, \sigma_2, \sigma_3\} &\vDash \tau \end{aligned}$$

etcétera. Si alguna de estas condiciones es satisfecha, entonces se contesta “sí”. De otra forma, continúa probando. Esto produce una respuesta afirmativa siempre y cuando $\Sigma \vDash \tau$, por el corolario al teorema de compacidad.

■

Más tarde, se querrá utilizar procedimientos eficaces para calcular funciones. Se tiene que una función f es efectivamente computable (o simplemente computable) si y sólo si existe un procedimiento eficaz que, dada una entrada x, a su vez, producirá la salida correcta $f(x)$.

Hasta aquí se tienen algunos de los temas más importantes en la lógica de enunciados, resuelve las actividades a continuación y considera la sección *Para saber más*.

Cierre de la unidad

Como observaste, esta unidad te permite tener un pensamiento matemático, lógico; las herramientas para estructurar este pensamiento y el lenguaje para poder expresar dicho pensamiento lógico con lógica matemática.

Para saber más

El enfoque que se presentó desde el inicio en este curso de lógica es, si bien apropiado, algo distinto a la forma en que usualmente se expone en un curso de lógica donde desde el principio se dan una serie de formalizaciones resultados. En este curso se dieron sólo ideas intuitivas y bosquejos de muchas demostraciones, para ver una forma alternativa de la presentacion de la lógica matematica, revisa los siguientes textos:

Supper, P. y Hill, S. (2005). *Introducción a la lógica matemática*.Méxic, Editorial Reverté.
Rautenberg, W. (2006). *A Concise Introduction to Mathematical Logic*,EUA, 2a ed. Springer.

En ambos considera revisar los temas relacionados con la lógica de enunciados, es decir, el capítulo 1.

Fuentes de consulta

- Amor, J. A. y Rojas, R. (1991) *Sistemas formales*. México: Vínculos Matemáticos, núm. 149, Facultad de Ciencias-UNAM.
- Amor, J. A. (1996). *Lógica proposicional dentro de la lógica de primer orden*. México: Vínculos Matemáticos, núm. 113, Facultad de Ciencias-UNAM.
- Enderton, H., A. (2001). *Mathematical Introduction to Logic*. EUA: Universidad de California, Academic Press.
- Kleene, S. C. (2002). *Mathematical Logic.* Nueva York: Dover Publications.
- Mendelson, E. (1997). *Introduction to Mathematical Logic*. EUA: Chapman & Hall.