Lógica Matemática para Informáticos

Sergio Augusto Cardona Torres Jorge Mario García Usuga
Alejandra María Pulgarín Galvis

2014

Lógica Matemática para Informáticos

Sergio Augusto Cardona Torres, Jorge Mario García Usuga, Alejandra María Pulgarín Galvis

2014

ISBN: 978-958-8801-22-3

Lógica Matemática para Informáticos

2014

Sergio Augusto Cardona Torres.
Programa de Ingeniería de Sistemas y Computación
Facultad de Ingeniería
Universidad del Quindío

Alejandra María Pulgarín Galvis.
Programa de Licenciatura en Matemáticas
Facultad de Educación
Universidad del Quindío

Jorge Mario García Usuga.
Programa de Licenciatura en Matemáticas
Facultad de Educación
Universidad del Quindío

Universidad del Quindío
2014

Presentación

La lógica matemática contribuye significativamente en la formación de los ingenieros de sistemas, informáticos y profesionales. En los currículos de Ciencias de la Computación (ACM & IEEE, 2008), Ingeniería de Software (ACM & IEEE, 2004b) e Ingeniería Informática (ACM & IEEE, 2004a), la contemplan como un concepto dentro su cuerpo de conocimiento. La lógica matemática posee una estrecha relación con áreas como las matemáticas discretas, la programación de computadores, las estructuras de datos, la teoría de grafos, las bases de datos (Castel de Haro, 2011). También está relacionada con la Inteligencia Artificial al ser la lógica la base de métodos para representar el conocimiento. En la Ingeniería de Software, la lógica se usa en la especificación de requisitos utilizando lenguajes formales.

Existe una importante cantidad de productividad académica relacionada con los temas de lógica matemática a nivel de pregrado universitario. Como metodología para la enseñanza de la lógica, es común que se presenten las técnicas básicas de resolución de problemas, acompañadas de una serie de ejercicios sobre cada una de las temáticas.

Los lineamientos metodológicos y pedagógicos del libro buscan aportar al propósito general de la formación de los ingenieros de sistemas y de profesiones afines. Así mismo, se pretende que el libro contribuya al desarrollo de capacidades en los cuales, el estudiante pueda explicar, demostrar, solucionar problemas y hacer representaciones, haciendo uso de la lógica matemática. El libro presenta por capítulos los elementos conceptuales fundamentales de la lógica. En cada uno de los capítulos se proponen una serie de actividades de aprendizaje, las cuales están orientados al desarrollo de las competencias definidas al inicio de las unidades temáticas.

La estructura del libro, está orientada a que el estudiante asuma en cada actividad un rol activo en su proceso de aprendizaje, pues se considera fundamental que ellos identifiquen la importancia de esta disciplina dentro de su proceso de formación. Así mismo se espera que el profesor cuente con una guía que soporte el aprendizaje en cada uno de los temas. La experiencia en la enseñanza de la lógica matemática motiva a proponer un texto universitario a partir del cual se resuelvan problemas lógico – matemáticos de una forma estructurada y formal.

Índice general

Introducción

El libro Lógica Matemática para Informáticos tiene como propósito fundamental, contribuir a la formación de los futuros profesionales de la Ingeniería de Sistemas y afines. El libro presenta los conceptos fundamentales de la lógica matemática con aplicaciones a las ciencias computacionales.

El lector de este libro identificará los elementos fundamentales de la lógica matemática. Así mismo, comprenderá los problemas que puede abordar y resolver desde esta área del conocimiento. En cada uno de los capítulos el lector podrá establecer la relación conceptual entre la teoría y su aplicación en el área de la Ingeniería de Sistemas. El libro presenta una serie de actividades a resolver por el estudiante, las cuales se presentan de forma incremental y buscando que se desarrolle un pensamiento formal para la solución de problemas.

El libro está estructurado en 5 capítulos, los cuales se presentan a continuación.

El capítulo 1 denominado fundamentos de la lógica matemática. Se presentan los aspectos básicos para determinar las propiedades de las expresiones proposicionales. Así mismo, se exponen los principales operadores booleanos. Finalmente se muestran los aspectos fundamentales para transformar expresiones en lenguaje natural a su equivalente en lenguaje proposicional, mediante el uso de los operadores lógicos y de los signos de agrupación.

En el capítulo 2 llamado cálculo proposicional, se presenta un lenguaje proposicional a partir de un alfabeto lógico, sintaxis y semántica. Se expone el concepto de interpretación booleana y su aplicación en la valoración semántica de expresiones proposicionales. También se exponen dos métodos de deducción automática: método de resolución y método de los tableros semánticos, a partir de los cuales es posible determinar propiedades de satisfacibilidad, validez, no validez e insatisfacibilidad de una fórmula proposicional.

El capítulo 3 denominado argumentos, reglas de inferencia y límites de la lógica, presenta los principales elementos de los métodos de razonamiento, a partir de los cuales se pueden determinar propiedades de validez y no validez. Se expone las reglas de inferencia más comúnmente aplicadas en el cálculo proposicional. Se explicarán reglas de inferencia como Modus Ponens, doble negación, Tollendo Tollens, Tollendo Ponens, De Morgan, Simplificación, Conmutativa, Bicondicional, Contrarrecíproca, Disyuntiva, entre otras.

El capítulo 4 llamado lógica de predicados, presenta los aspectos fundamentales de la aplicación de los predicados. Se presentan las equivalencias lógicas entre los diferentes cuantificadores (universales

y existenciales). Se explican los conceptos de fórmulas atómicas, términos y argumentos. También se realiza la formalización de los predicados y las propiedades de las fórmulas.

El capítulo 5 del libro denominado programación lógica en Prolog, presenta los elementos sintácticos fundamentales de la programación lógica. Se introduce el concepto de sección, hecho y conjuntos de reglas aplicables en Prolog. Se muestra el funcionamiento y aplicación de la recursividad en Prolog.

Como elemento complementario al proceso de enseñanza al estudiante, se propone el desarrollo de dos proyectos de curso. En ambos proyectos se describe el desarrollo de dos herramientas informáticas que soporten el método de resolución y el método de los tableros semánticos. Estos proyectos se establecen como estrategia de formación complementaria en el curso.

Para el desarrollo del proyecto, se propone la metodología de proyectos formativos; los cuales, son una metodología para el desarrollo y evaluación de competencias. La metodología está fundamentada en el método de proyectos, el cual fue conceptualizado por Kilpatrick (Kilpatrick, 1918), como un proceso dinámico para organizar la enseñanza mediante fases (propuesta, planificación, ejecución y evaluación) y actividades formativas articuladas, en las cuales se involucra de forma activa a los estudiantes en contextos de desempeño específicos y en donde se espera, que ellos desplieguen sus diferentes competencias encaminadas a la solución de un problema. Un proyecto formativo se estructura por medio de fases en las cuales, participan el profesor y los estudiantes, estas fases son los escenarios en donde se establecen las actividades de aprendizaje necesarias para que los estudiantes alcancen las competencias definidas. Las fases de un proyecto formativo son: direccionamiento, planeación, ejecución y socialización (Tobón, 2013).

La evaluación en los proyectos formativos está basada en evidencias, las cuales, son valoradas mediante rúbricas con los niveles de dominio. Los niveles de dominio son los grados que describen cómo se forman y desarrollan las competencias desde lo más sencillo a lo más complejo, y a partir de los cuales se unifican los criterios que orientan al docente y a los estudiantes en torno a la valoración de las competencias que se van desarrollando (Tobón, 2013)

CAPÍTULO 1

Fundamentos de la lógica matemática

> En cualquier formalización consistente de las matemáticas que sea lo bastante fuerte para definir el concepto de números naturales, se puede construir una afirmación que ni se puede demostrar ni se puede refutar dentro de ese sistema.
>
> Kurt Gödel.
> 1906 – 1978

Los aprendizajes esperados para el capítulo fundamentos de lógica matemática son:

- Capacidad para identificar y comprender los conceptos fundamentales de la lógica matemática y su aplicación en la resolución de diversos tipos de problemas.
- Representa mediante un lenguaje de lógica proposicional diversos tipos de expresiones considerando las propiedades de los operadores lógicos.
- Determina las propiedades de una expresión proposicional aplicando adecuadamente un algoritmo de deducción.
- Comprende el valor de una expresión teniendo en cuenta el significado de los operadores lógicos y los signos de agrupación.
- Transformar una expresión proposicional a su equivalente en notación prefija.
- Construir el árbol de formación de una fórmula proposicional de acuerdo a la precedencia de los operadores lógicos.
- Transformar una expresión en lenguaje natural a su equivalente en lenguaje proposicional, haciendo uso de los operadores lógicos y de los signos de agrupación.

1.1 Introducción

La lógica en el área de las matemáticas, es la ciencia que estudia los métodos de razonamiento, permite demostrar teoremas, indica reglas y técnicas para determinar la validez o no de un argumento y proporciona métodos adecuados para obtener conclusiones. La lógica por medio de la formalización del lenguaje y de sus reglas básicas proporciona las herramientas necesarias para resolver diferentes tipos de problemas que tienen sus orígenes y aplicaciones en diferentes áreas de las ciencias. En diversos trabajos (Zarate, 2003), (Castel de Haro, 2011), (Ostra, 2008), se presentan las concepciones filosóficas y matemáticas que dieron origen al concepto de lógica.

El ser humano puede comunicarse por medio de diversos sistemas convencionales de símbolos. Una de estas formas de comunicación es posible mediante el lenguaje natural, a través del cual es posible estructurar oraciones interrogativas (preguntas), oraciones exclamativas (de sorpresa), oraciones desiderativas (de deseo), oraciones imperativas (de orden) y oraciones declarativas (afirmativas o negativas). La lógica provee los elementos para describir y representar el conocimiento a través de métodos de formalización de las frases declarativas.

En este capítulo se expondrán los conceptos básico de la lógica proposicional desde las frases declarativas simples (enunciados o proposiciones), debido a que estas se constituyen en los elementos básicos de transmisión del conocimiento humano. Las frases interrogativas, exclamativas, desiderativas e imperativas, no denotan hechos y por lo tanto no es posible verificar su veracidad.

1.2 Proposiciones

Una proposición matemática es una expresión declarativa expresada en lenguaje natural que posee un significado de acuerdo un criterio definido y a partir del cual puede ser clasificada inequívocamente como verdadera o falsa, pero no de ambas formas a la vez. En la lógica matemática se consideran las oraciones declarativas como oraciones a partir de las cuales se pueden constituir proposiciones. Todas las proposiciones se consideran oraciones, pero no todas las oraciones se consideran proposiciones.

Las proposiciones pueden denotar acciones, establecer relaciones entre objetos y sujetos, y determinar propiedades a objetos, personas y animales.

Las siguientes oraciones o enunciados son proposiciones, debido a que para cada de ellas es posible establecer si es verdadera o falsa.

- Los estudiantes participan en la jornada cultural.
- La política en Colombia se está trasformando.
- Alberto se graduó de la Maestría en Educación.
- Java es un lenguaje de programación.
- 31 no es un número primo.
- Las personas utilizan las redes sociales para establecer relaciones de amistad.
- Se firmó el tratado de libre comercio con Corea del Sur.

En el lenguaje natural existen oración que son comúnmente utilizadas, estos son las oraciones interrogativas, exclamativas, imperativas. En lenguajes naturales tales como el español, alemán, inglés, entre otros, las proposiciones no se pueden constituir a partir de oraciones imperativas, exclamativas o interrogativas. Este tipo de oraciones no especifican hechos y por lo tanto no es posible determinar en ellas propiedades de verdad o falsedad. Tampoco las oraciones exclamativas o admirativas, las

dubitativas y los juicios de valor, son consideradas como proposiciones (Zarate, 2003).

Las siguientes oraciones no se consideran proposiciones.(ver tabla 1.1)

Tipo	Oración
Interrogativas	¿Quién soy? ¿Qué hora es? ¿Cuál es tu fecha de nacimiento?
Imperativas	Debemos alimentarnos sanamente. Tienes que cuidar a tu hermana.
Exclamativas	¡Hola amigo! ¡Por favor estudia! ¡Por Dios!
Dubitativas	Posiblemente es mejor de esa manera. Quizás no estudió lo suficiente.

Tabla 1.1: Tipos de Proposiciones

El significado de verdad o falsedad de una proposición está determinado a partir de su estructura semántica. Por ejemplo si se tienen las oraciones:

- Está descompuesto el automóvil.
- El automóvil está descompuesto.

Ambas proposiciones desde la interpretación en lenguaje natural son iguales, a pesar de tener una estructura sintáctica diferente, por lo tanto, la verdad o falsedad de una proposición se determina desde el punto de vista de su estructura, sin tener en cuenta el significado semántico. Por ejemplo, si se tiene la expresión:

- Violeta brilla en América.

En esta expresión, no se sabe si Violeta es una persona, una planta, un animal o cualquier otro concepto. Es posible otorgar el valor de verdad o falsedad a la expresión, pero su semántica resulta compleja para ubicarla en un determinado contexto.

Las siguientes oraciones presentan posible ambigüedad en su significado semántico:

- Francisco le dio la vuelta a Santiago.
- Lima está de fiesta.
- Florencio vive en la cordillera central.
- Kennedy pertenece a Bogotá.
- Karina invadió a Nicaragua.

.:Actividad 1.1. *Identificación de proposiciones*

1. Indique cuáles de las siguientes oraciones corresponden o no a proposiciones.

Oración	Si	No
Que tengas un feliz día		
Los números primos son impares		
Alberto es disciplinado		
El número 8 es un número perfecto		
¿Por qué no participaste en el evento?		
X es un numero par		
Nunca engañes a las personas		
¡No hagas parte de ese juego!		
¿Podrías asumir esa deuda?		
No puedes usar las redes sociales		
El País está en un periodo de cambio		
Java no es un lenguaje de programación		
No cambies de gasolina a Diésel		

2. Dadas las siguientes expresiones, señale a cual tipo pertenece: imperativa, exclamativa o dubitativa.

Oración	Imperativa	Exclamativa	Dubitativa
Tal vez es posible la solución al caso			
¿Cuántos años tienes?			
Por favor entregue las llaves de la moto			
Quizás las cosas mejoren para ella			
Seguramente será elegido como alcalde			
¿Es posible una rebaja de precio?			
¡Me imagino la alegría de tu papa!			
Escuche lo que dice el Presidente			
¡Qué alegría saber que llegaste bien!			
A lo mejor viaje a Córdoba el próximo mes			
¡Felicidades por tu nuevo título!			
Sin duda alguna es la mejor opción			

1.3 Evaluación de las proposiciones

Una proposición puede ser clasificada como verdadera o falsa, para lo cual es necesario establecer una notación específica. Tradicionalmente las proposiciones se representan con las letras finales del alfabeto. Para este libro se utilizarán las letras minúsculas: p, q, r, s, t,... . Cada una de estas letras recibe el nombre de átomo.

1.3.1 Representación de una proposición

La estructura para la representación de una proposición será un átomo acompañado de : (dos puntos), y a continuación la expresión que denota la proposición.

p: proposición 1
q: proposición 2
r: proposición 3

Basados en la estructura definida, las siguientes expresiones denotan una proposición:

- p: Bogotá tiene alcaldías locales.
- q: Valdivia está ubicado geográficamente al sur de Chile.
- r: $54 > 38\,\%\ 2$
- t: $x + y = y + x$
- u: los tiquetes a Miami están en promoción.
- s: los estudiantes realizan autoevaluación de sus trabajos.

Cuando se clasifica o establece la verdad o falsedad a una proposición, se está asignando una interpretación. Para representar una interpretación a la proposición, se usará la letra minúscula v, seguida de un átomo entre paréntesis y finalmente la asignación del valor de verdad o de falsedad.

v(átomo) = valor de verdad
v(átomo) = valor de falsedad

Una proposición puede tomar sólo uno de dos posibles valores: Verdadero (V) o Falso (F).

$$v(\text{átomo}) = V \text{ ó } v(\text{átomo}) = \text{F}$$

Los siguientes ejemplos muestran la interpretación dada a la proposición:

Proposición	**Interpretación**
p: Manuel es experto en mercadeo	$v(p) = F$ (p se asignó falso)
q: El número 9 es un numero perfecto	$v(q) = V$ (q se asignó verdadero)
r: $2 + 8 \neq 10$	$v(r) = V$ (r se asignó verdadero)
t: 10 no es divisible por 4	$v(t) = F$ (p se asignó falso)

1.3.2 Negación de una proposición

Cuando se niega una proposición, ésta se convierte en falsa si es verdadera, o se convierte en verdadera si es falsa. La negación se puede expresar en lenguaje matemático a través de diferentes símbolos, entre los que se encuentran:

- Sobreponiendo a una proposición el símbolo -
- Anteponiendo a una proposición el símbolo $\neg$ o el símbolo $\sim$ (notación simbólica de Scholz)
- Anteponiendo a una proposición el símbolo !

Para este libro, la negación se representará mediante el símbolo $\neg$. En lenguaje natural algunas de las expresiones usadas para expresar la negación son:

- No p

- Es falso p
- No es cierto p
- No es el caso p
- No se da el caso que p

Si p es verdadera, entonces $\neg p$ es falsa, y viceversa.

p	$\neg p$
V	F
F	V

Para mostrar la aplicación de la negación en lenguaje natural, a partir de la oración: "*Todas redes sociales fomentan la difusión de información*", se presentan las diversas formas en que ésta puede ser negada:

- No todas las redes sociales fomentan la difusión de información.
- No es cierto que todas las redes sociales fomentan la difusión de información.
- No es el caso que todas las redes sociales fomentan la difusión de información.
- Es falso que todas las redes sociales fomentan la difusión de información.
- Algunas redes sociales no fomentan la difusión de información.

En algunos casos traducir directamente la interpretación de la negación al lenguaje natural puede resultar complejo dadas las implicaciones semánticas a las que se pueden llegar.

Por ejemplo si se tiene la proposición: "*La variable se encuentra inicializada*", una negación directa sería: "*No la variable se encuentra inicializada*". La forma en la cual está expresada la negación de la proposición no es común y su expresión semántica no es la más adecuada. Para este caso es adecuado utilizar otras formas para la lectura de la negación de la oración. Por ejemplo, una mejor forma de expresar la negación de la proposición negación sería: "*La variable no se encuentra inicializada*".

El símbolo de la negación ($\neg$) es un operador unario que se puede aplicar a los átomos y a expresiones más complejas. A continuación se presenta una proposición y al frente su correspondiente en negación.

Proposición	**Negación**
37 es un número primo	37 no es número primo
5 es mayor que 7	5 no es mayor que 7
Las redes sociales existen	Es falso que las redes sociales existen
x es mayor que z	x no es mayor que z
El empleado gana bonificación	El empleado no gana bonificación
$x > y$	$x \leq y$

La negación también se aplica a los átomos.

Valor de la proposición	Valor de la negación de la proposición
$v(q) = F$	$v(\neg q) = V$
$v(\neg p) = V$	$v(p) = F$
$v(r) = V$	$v(\neg r) = F$
$v(\neg s) = F$	$v(s) = V$
$v(\neg\neg t) = F$	$v(\neg t) = V$

En la lógica matemática se utiliza la doble negación, es decir, la negación de la negación de una determinada proposición. Cuando a una proposición se aplica la doble negación, esta es por si misma la misma proposición. La doble negación se puede expresar en lenguaje matemático mediante diferentes símbolos:

- Sobreponiendo a una proposición dos veces el símbolo -
- Anteponiendo a una proposición dos veces el símbolo $\neg$ o dos veces el símbolo $\sim$
- Anteponiendo a una proposición dos veces el símbolo !

Los siguientes son ejemplos en lenguaje natural donde se usa la doble negación:

- *No es cierto que no tenga motivación para estudiar.*
- *No es verdad que el programa académico no está acreditado*
- *Nadie sabe nada*
- *No voy nunca a viajar en avión*

En la actividad 1.2 se proponen ejercicios para trabajar con negación de las proposiciones.

.:Actividad 1.2. *Negación de proposiciones*

1. Dadas las siguientes expresiones, escriba su negación correspondiente.

Proposición	Negación de la proposición
28 es un número perfecto	
32 es el factorial de 6	
21 en serie Fibonacci corresponde al valor 8	
2 es el máximo común divisor entre 26 y 6	
Alas pertenece a las aerolíneas nacionales	

2. Dadas los siguientes valores de las proposiciones, escriba su correspondiente negación.

Proposición	Negación de la proposición
$v(\neg q) = V$	
$v(p) = V$	
$v(\neg r) = F$	
$v(\neg\neg s) = F$	
$v(\neg\neg\neg t) = V$	

3. Dadas las siguientes expresiones, niéguelas usando mínimo las frases más utilizadas en lenguaje natural:
   - El plan de estudios de Ingeniería de Sistemas y Computación tiene 178 créditos y tiene registro calificado.
   - El presidente de la república está buscando la paz con los actores de la violencia.
   - Colombia tiene representación en la ONU.

1.4 Tipos de proposiciones

Las proposiciones se clasifican en dos tipos de acuerdo a su estructura: simples y compuestas. A continuación se explica cada una de ellas.

1.4.1 Proposición simple

Una proposición en su forma más elemental se llama atómica o simple. Se considera simple si solo expresa una idea sobre algo y no incorpora conjunciones de índole gramatical. En las proposiciones simples no se encuentran conectores lógicos y no es posible encontrar en ellas otras proposiciones.

Ejemplos de proposiciones atómicas o simples.

- o: Moodle es un sistema de gestión de aprendizaje.
- p: Los sistemas adaptativos son utilizados en la educación virtual.
- q: Felipe fue ascendido en su puesto.
- r: 23 es un número impar.
- s: La temperatura llegó a los 30^oC
- t: 17 no es un número compuesto.
- $:u$ $15\,\%\ 4 * 5 < 200$

Las proposiciones simples de acuerdo a su estructura pueden ser de carácter relacional o basadas en predicados. Las de carácter relacional se constituyen de dos o más objetos o sujetos vinculados entre sí, por ejemplo:

- Armenia es la capital del Quindío.
- Tatiana es la madrina de Alberto.
- Juliana es mayor que Andrea.

Las proposiciones simples basadas en predicados, son las que constan de un objeto o sujeto del cual se afirma algo, por ejemplo:

- Juan es investigador.
- 11 no es un número perfecto.
- La mascota está enferma.

La característica principal de las proposiciones simples, es que estas no se pueden dividir y por esta razón, es posible identificar el valor de la formula atómica, bien sea verdadero o falso, lo que implica que el número posible de interpretaciones de una fórmula simple es 2.

1.4.2 Proposición compuesta

Las proposiciones compuestas, son aquellas que están conformadas por proposiciones simples unidas a través de conectores lógicos. Los conectores lógicos más frecuentemente usados son: '*si... entonces...*', '*si y sólo si...*', '*y*', '*o*'. Una proposición compuesta también se puede identificar cuando ésta tiene varios verbos, sujetos u objetos. A continuación se muestran algunas proposiciones compuestas:

- m: Gabriel tuvo nacionalidad colombiana y mexicana
- n: 15 es un número divisible por 4 o por 13
- o: $x^2 - 16 = 0$ si y sólo si $x = 4$ o $x = -4$
- p: Java es un lenguaje de programación imperativo o declarativo
- q: Camilo y Orlando están haciendo su tesis de maestría
- r: Felipe deposita su tesis si y sólo aprueba los créditos
- s: 18 es múltiplo de 9 y divisor de 54, o 18 es divisible por 3

Los conectivos lógicos permiten la relación de proposiciones simples. Por ejemplo, si se tienen las proposiciones simples:

- o: La recursividad en la programación es un tema complejo.
- p: El estudiante no tiene bases de programación.
- q: Los algoritmos recursivos tienen pocas líneas de código.
- r: El estudiante utiliza programación iterativa.

Es posible construir enunciados compuestos que denotan proposiciones más complejas para su análisis. En este caso se utilizarán los conectores lógicos: y, o, si... entonces..., si y sólo si...

- La recursividad en la programación es un tema complejo si y solo si el estudiante no tiene bases de programación.
- El estudiante no tiene bases de programación o el estudiante utiliza programación recursiva.
- Si el estudiante no tienen bases de programación, entonces utiliza programación iterativa.

La verdad o falsedad de una proposición compuesta se determina a partir de la verdad o falsedad de cada proposición simple y de los operadores que se usan como conectores.

.:Actividad 1.3. *Proposiciones simples y compuestas*

1. Dadas las siguientes proposiciones, construya enunciados utilizando los conectores lógicos: *y*, *o*, *si... entonces...*, *si y sólo si...*
   - o: Los tiquetes aéreos son costosos.
   - p: El repuesto es construido en Francia.
   - q: El impuesto a la gasolina es anual.
   - r: Los viajeros han disminuido el viaje.

Nombre	Conectivo lógico	Símbolo
Conjunción	Y	$\wedge$
Disyunción Inclusiva	O	$\vee$
Disyunción Exclusiva	O	$\veebar$
Condicional	Si ... entonces	$\rightarrow$
Bicondicional	si y sólo si	$\leftrightarrow$

Tabla 1.2: Principales operadores lógicos

1.5 Operadores lógicos

Las proposiciones se relacionan a través de operadores que permiten formar otras proposiciones, estos operadores son los responsables de las relaciones lógicas entre las expresiones. Los operadores que permiten la unión proposiciones se llaman operadores binarios. Los siguientes son los principales operadores lógicos: (tabla 1.2)
A continuación se hará un análisis de cada uno de los conectivos lógicos.

1.5.1 Conjunción

El conectivo lógico para la conjunción se representa mediante el símbolo $\wedge$. Sean p y q dos proposiciones, entonces $p \wedge q$ es llamada la conjunción entre las proposiciones p y q. Algunas frases en lenguaje natural en las que aparece la conjunción pueden ser:

- p y q
- p pero q
- p aunque q
- p sin embargo q
- p no obstante q
- p a pesar de q
- p a menos q
- p igualmente q

La proposición $p \wedge q$ se considera verdadera cuando p es verdadera y q es verdadera, es decir, cuando ambas proposiciones son verdaderas a la vez. Algunos ejemplos de representación en lenguaje natural en los cuales se utiliza la conjunción son los siguientes:

- La cobertura en educación y la calidad de los profesores es inadecuada.
- El rendimiento académico del estudiante es bueno, sin embargo, su actitud no es la adecuada.
- El producto es bueno, no obstante, su precio lo hace inalcanzable.
- El rector tiene un plan de gobierno, no obstante, tiene un equipo de trabajo desintegrado.
- Aunque esté lloviendo es posible realizar el vuelo.
- Está nevando pero es posible navegar.

En el lenguaje natural es frecuente usar formas abreviadas para expresar proposiciones, por ejemplo:

- Jorge y Mauricio van a estudiar un Doctorado en Ingeniería.

Equivale a expresar por separado:

- Jorge va a estudiar un Doctorado en Ingeniería.
- Mauricio va a estudiar un Doctorado en Ingeniería.

Es común la representación de proposiciones utilizando átomos unidos a través de conectivos lógicos. Por ejemplo si se tiene la expresión $p \wedge q$, se puede dar una interpretación arbitraria a cada uno de los átomos que componen la expresión:

- $v(p) = V$
- $v(q) = F$

Entonces $v(p \wedge q) = F$, dado que al menos una de las interpretaciones es falsa.

Las siguientes proposiciones tienen asignadas interpretaciones arbitrarias a los átomos, los cuales determinan la evaluación de la proposición.

Proposición	**Interpretación**	**Evaluación Proposición**
$(p \wedge \neg q)$	$v(p) = F, \quad v(\neg q) = V$	$v(p \wedge \neg q) = F$
$(p \wedge s)$	$v(p) = V, \quad v(s) = V$	$v(p \wedge s) = V$
$(p \wedge q)$	$v(p) = F, \quad v(q) = F$	$v(p \wedge q) = F$
$(\neg q \wedge q)$	$v(\neg p) = V, \quad v(q) = F$	$v(\neg q \wedge q) = F$
$\neg(p \wedge \neg q \wedge r)$	$v(p) = V, \quad v(\neg q) = V, \quad v(r) = F$	$v\neg(p \wedge \neg q \wedge r) = V$

Tabla 1.3

Para el caso de la expresión: $\neg(p \wedge \neg q \wedge r)$, de acuerdo con los valores asignados a cada uno de los átomos, se tiene que $(p \wedge \neg q \wedge r)$ evalúa como falsa la expresión, pero teniendo en cuenta que a la misma le antepone el operador $\neg$, la expresión queda evaluada como verdadera.

El uso del operador de conjunción, es equivalente en la lógica de conjuntos a la operación de intersección.

.:Actividad 1.4. *Conjunción*

1. Dadas las siguientes proposiciones, defina una interpretación para cada uno de los átomos y evalúe cada proposición.

Proposición	**Interpretación**	**Evaluación Proposición**
$\neg(p \wedge \neg q)$		
$\neg(p \wedge \neg q \wedge r)$		
$(p \wedge q \wedge r)$		
$((\neg p \wedge q) \wedge p)$		

2. A continuación se muestran pares de proposiciones, a los cuales es necesario determinar su valor de verdad.

Proposición	**Proposición**	**Expresión**	**Evaluación**
t: 2 es número par (V)	s: 2 es número primo (V)	$t \wedge s$	
w: $5 < 7$ (V)	r: $4 \geq 0$ (V)	$r \wedge w$	
p: $0 = x$ (F)	q: $x + 1 = x$ (F)	$p \wedge q$	

1.5.2 Disyunción inclusiva

La disyunción inclusiva se representa mediante el conectivo lógico $\vee$. La proposición $p \vee q$ es llamada la disyunción inclusiva entre las proposiciones p y q. Se considera $p \vee q$ falsa, cuando la proposición p y la proposición q son falsas a la vez. Este operador se evalúa como verdadero, en el caso que al menos una proposición sea verdadera, incluyendo el caso en que ambas variables también lo sean.

Algunas frases en las que aparece la disyunción son:

- p o q
- p o q o ambos
- al menos p o q
- mínimo p o q

Algunas frases de representación en lenguaje natural en los cuales se utiliza la disyunción son las siguientes:

- El experimento estuvo mal diseñado o las personas mintieron.
- Para cobrar el subsidio al menos debe ser menor de edad o estar estudiando.
- Para pagar el impuesto debe tener cuenta corriente o de ahorros.
- El parcial estaba difícil o mal redactado, o ambas cosas.

Las siguientes proposiciones tienen asignadas interpretaciones arbitrarias a los átomos, los cuales determinan la evaluación de la proposición.

Proposición	**Interpretación**	**Evaluación Proposición**
$(\neg p \vee \neg q)$	$v(\neg p) = F,\ v(\neg q) = V$	$v(\neg p \vee \neg q) = V$
$(p \vee q \vee r)$	$v(p) = F,\ v(q) = F,\ v(r) = F$	$v(p \vee q \vee r) = F$
$(p \vee q \vee r)$	$v(p) = F,\ v(q) = F,\ v(r) = F$	$\neg v(p \vee q \vee r) = V$

Tabla 1.4

El uso del operador de conjunción, es equivalente en la lógica de conjuntos a la operación de la unión.

.:Actividad 1.5. *Disyunción Inclusiva*

1. Dadas las siguientes proposiciones, defina una interpretación para cada uno de los átomos y evalúe.

Proposición	**Interpretación**	**Evaluación Proposición**
$\neg(\neg p \vee \neg q)$		
$\neg(p \vee q \vee \neg r)$		
$(p \vee q \vee r)$		
$\neg\neg(\neg p \vee \neg r)$		
$\neg\neg\neg(\neg p \vee \neg\neg q)$		

2. A continuación se muestran pares de proposiciones, a los cuales es necesario determinar su valor de verdad.

Proposición	**Proposición**	**Expresión**	**Evaluación**
r: 2 es número primo (V)	s: 2 es número positivo (V)	$r \vee s$	
t: $2 + 8 \neq 10$ (F)	q: $(5+3) \leq 2$ (F)	$t \vee q$	
u: El triángulo tiene tres lados (V)	w: El rectángulo es un pentágono (F)	$t \vee w$	

1.5.3 Disyunción exclusiva

La disyunción exclusiva se representa mediante el conectivo lógico $\veebar$. La proposición $p \veebar q$ se denomina la disyunción exclusiva, $p \veebar q$ es verdadera, únicamente cuando una de las dos proposiciones es verdadera, pero no ambas a la vez:

Algunos ejemplos de representación en lenguaje natural en los cuales se utiliza la disyunción exclusiva son los siguientes:

- Mañana iré a cine o iré a estudiar.
- La tesis de maestría es laureada o meritoria.
- La fecha límite para el pago del seguro es abril o mayo.
- La elección del procurador es por vía directa o mediante consulta.

La disyunción se usa en un contexto de exclusividad, por ejemplo la oración:

- La pasantía estudiantil dará inicio el mes de diciembre o el mes de enero.
- Mario compra el tiquete para viajar a Aruba o Lima.
- Me comprometo a trabajar con Tatiana o con Cristina.

Se puede interpretar que Mario compra el tiquete para viajar a Aruba o para viajar a Lima, pero no en ambas situaciones. Así mismo que la pasantía estudiantil dará inicio para un mes en especial y no para ambos.

Las siguientes proposiciones tienen asignadas interpretaciones arbitrarias a los átomos, los cuales determinan la evaluación de la proposición.

Proposición	**Interpretación**	**Evaluación Proposición**
$(\neg p \veebar \neg q)$	$v(\neg p) = F,\ v(\neg q) = V$	$v(\neg p \veebar \neg q) = V$
$(p \veebar q)$	$v(p) = F,\ v(q) = F$	$v(p \veebar q) = F$
$(p \veebar q)$	$v(p) = V,\ v(q) = V$	$v(p \veebar q) = V$
$\neg(p \veebar q)$	$v(p) = V,\ v(q) = V$	$v\neg(p \veebar q) = F$

Tabla 1.5

No es sencillo definir si una oración es inclusiva o exclusiva. Existen casos en los cuales es sencillo identificar cuando una oración se interpreta de forma exclusiva, como por ejemplo en "*el niño está despierto o está dormido*". Existen casos en los cuales una oración se interpreta de forma inclusiva, por ejemplo "*Quindío es más pequeño en extensión que Amazonas o Guaviare*".

.:Actividad 1.6. *Disyunción Exclusiva*

1. Dadas las siguientes proposiciones, defina una interpretación para cada uno de los átomos y evalúe las expresiones.

Proposición	**Interpretación**	**Evaluación Proposición**
$(\neg p \veebar q \veebar s)$		
$(\neg p \veebar q)$		
$\neg(p \veebar q \veebar r)$		
$\neg(\neg p \veebar \neg q \veebar r)$		

2. A continuación se muestran pares de proposiciones, a los cuales es necesario determinar su valor de verdad.

Proposición	**Proposición**	**Expresión**	**Evaluación**
p: 2 es un número par (V)	q: 2 es un número impar(F)	$p \veebar q$	
t: 15 es un número primo (F)	w: 15 es un número compuesto (V)	$t \veebar w$	

1.5.4 Condicional

El conectivo lógico que representa el condicional es el símbolo $\rightarrow$. Sean p y q dos proposiciones: entonces si p entonces q, se representa mediante $p \rightarrow q$. La proposición $p \rightarrow q$ es falsa si la primera proposición (antecedente) es verdadera y la segunda proposición (consecuente) es falsa. Algunos casos de representación en lenguaje natural en los cuales se utiliza el condicional son:

- Si p entonces q
- p implica q
- p sólo si q
- q si p
- p es suficiente para q
- Para q es suficiente p
- No p a menos que q
- q cuando p
- q es necesario para p
- Para p es necesario q
- p en consecuencia q
- p se deduce q
- p por ende q

Algunos ejemplos en lenguaje natural son los siguientes:

- Si el sol está brillando entonces se puede hacer deporte.
- Si Pedro es matemático entonces calcula.
- Si un número es par entonces es divisible por 2.
- Si un número tiene dos divisores es suficiente para que sea primo.
- Un número es divisible por 5 cuando termina en 0 o en 5.
- Hoy es lunes y por ende hay pico y placa para mi carro.
- José perdió la materia, en consecuencia perdió el semestre.

En el condicional si el antecedente es verdadero, su valor de verdad es igual al valor de verdad de su consecuente. Si el antecedente es falso, entonces el condicional es verdadero. Por ejemplo la oración:

- Si el Barcelona gana, el número de socios se incrementa

En este caso si es verdad que el Barcelona gana, entonces la oración "*Si el Barcelona gana, el número de socios se incrementa*", es verdadera si y solo si se incrementa el número de socios. Si el caso fuera que en Barcelona no gana, entonces la oración "*Si el Barcelona gana, el número de socios se incrementa*" es verdadero ya que su antecedente es falso.

A continuación se darán unas fórmulas proposicionales y se asignarán interpretaciones arbitrarias a los átomos. El operador que se utilizará es el condicional.

Proposición	**Interpretación**	**Evaluación Proposición**
$(\neg q \to p)$	$v(\neg q) = F,\ \ v(p) = V$	$v(\neg q \to p) = V$
$(\neg q \to p)$	$v(p) = V,\ \ v(s) = F$	$v(p \to p) = F$
$\neg(p \to q)$	$v(p) = V,\ \ v(q) = F$	$v\neg(p \to q) = V$
$(p \to s) \to q$	$v(p) = V,\ \ v(s) = F,\ \ v(q) = F$	$v(p \to s) \to q = F$

Tabla 1.6

.:Actividad 1.7. *Condicional*

1. Dadas las siguientes proposiciones, defina una interpretación para cada uno de los átomos y evalúe mediante su valor cada proposición.

Proposición	**Interpretación**	**Evaluación Proposición**
$\neg\neg(\neg q \to p)$		
$\neg(\neg p \to \neg s)$		
$\neg(p \to q) \to \neg r$		
$(p \to s) \to (q \to r)$		
$\neg(p \to s) \to \neg(q \to r)$		
$\neg\neg(p \to s) \to \neg\neg(q)$		

2. A continuación se muestran pares de proposiciones, a los cuales es necesario determinar su valor de verdad.

Proposición	**Proposición**	**Expresión**	**Evaluación**
p: $5 + 5 = 10$ (V)	q: $5 \times 2 = 10$ (V)	$p \to q$	
w: 8 es un número par (V)	m: 8 no es divisible por 2 (F)	$w \to z$	

1.5.5 Bicondicional

El conectivo lógico que representa la disyunción inclusiva es el símbolo $\leftrightarrow$. Sean p y q dos proposiciones, la proposición p si y sólo si q, se representa $p \leftrightarrow q$. El bicondicional es verdadero únicamente cuando tanto p como q tienen los mismos valores de verdad.

Algunos ejemplos de representación en lenguaje natural en los cuales se utiliza el bicondicional son los siguientes:

- p si y sólo si q
- p es necesario y suficiente para q
- p es equivalente a q
- p cuando y sólo cuando q
- p entonces y sólo entonces q

La proposición $p \leftrightarrow q$ es verdadera sólo cuando las dos proposiciones son ambas verdaderas o falsas.

Algunos ejemplos en el lenguaje natural donde podemos encontrar el bicondicional:

- Juan ve si y sólo si no es ciego
- $5 + 5 = 10$ si y sólo si $5 \times 2 = 10$
- 28 es par si y sólo si es divisible por 2
- Un número es compuesto si y sólo si tiene más de dos divisores
- Un número es divisible por 3 si y sólo si al sumar sus cifras el resultado es múltiplo de 3

A continuación se darán unas fórmulas proposicionales y se asignarán interpretaciones arbitrarias a los átomos. El operador que se utilizará es el condicional.

Proposición	**Interpretación**	**Evaluación Proposición**
$(\neg q \leftrightarrow p)$	$v(\neg q) = F, \quad v(p) = F$	$v(\neg q \leftrightarrow p) = V$
$(p \leftrightarrow s)$	$v(p) = V, \quad v(s) = F$	$v(p \leftrightarrow s) = F$
$\neg(p \leftrightarrow s)$	$v(p) = F, \quad v(s) = V$	$v(p \leftrightarrow s) = F$
$\neg\neg(p \leftrightarrow \neg q)$	$v(p) = V, \quad v(\neg q) = F$	$v(p \leftrightarrow s) = F$

Tabla 1.7

.:Actividad 1.8. *Bicondicional*

1. Dadas las siguientes proposiciones, defina una interpretación para cada uno de los átomos y evalúe mediante su valor cada proposición.

Proposición	**Interpretación**	**Evaluación Proposición**
$\neg$ ($\neg$ q $\leftrightarrow$ s)		
$\neg$ (p $\leftrightarrow$ $\neg$ s)		
($\neg$ p $\leftrightarrow$ $\neg$ $\neg$ s)		
$\neg$ $\neg$ ($\neg$ p $\leftrightarrow$ $\neg$ q)		
($\neg$ q $\leftrightarrow$ s) $\leftrightarrow$ p		

2. A continuación se muestran pares de proposiciones, a los cuales es necesario determinar su valor de verdad.

Proposición	**Proposición**	**Expresión**	**Evaluación**
p: 10 no es número par (F)	q: $10 \geq 8$ (V)	$p \leftrightarrow q$	
q: 28 es un numero perfecto (V)	r: 28 no es un número impar (V)	$q \leftrightarrow r$	

3. Dadas las siguientes proposiciones, escribir la expresión en lenguaje natural:

   p: Aprobó la asignatura

   q: Le gusta la matemática

   s: Ganó la beca

Proposición	**Expresión**
$\neg p$	
$\neg q$	
$\neg(p \wedge q)$	
$\neg(p \vee q)$	
$(\neg p \vee \neg q)$	
$(\neg q \rightarrow p)$	
$(\neg q \leftrightarrow p)$	
$\neg(q \rightarrow p)$	
$\neg\neg p$	
$\neg\neg q$	
$\neg(\neg q \leftrightarrow \neg p)$	
$(p \wedge q \wedge s)$	
$s \vee (p \wedge q)$	

1.6 Precedencia de los conectivos lógicos

El uso de los operadores lógicos requiere establecer un orden de precedencia para su aplicación en una fórmula proposicional. Con este orden se evitan ambigüedades en la evaluación de las fórmulas. Si no se establece esta prioridad de los operadores, es muy posible generar más de una interpretación para alguna fórmula. Es común que el orden de precedencia esté determinado de la siguiente manera:

1. Operador de negación $\neg$
2. Operador de conjunción $\wedge$
3. Operador de disyunción $\vee$
4. Operador condicional $\rightarrow$
5. Operador bicondicional $\leftrightarrow$

El operador unario $\neg$ tiene la prioridad más alta. Para el caso de los operadores binarios $\wedge$ tiene mayor prioridad seguido de $\vee$, $\rightarrow$ y posteriormente $\leftrightarrow$. Cuando se tengan expresiones con operadores que tengan el mismo orden de precedencia, se analizará la expresión de izquierda a derecha. Por ejemplo

la expresión $\neg p \vee q \wedge r$, se debe entender como $\neg p \vee (q \wedge r)$.

Las fórmulas pueden ser escritas con paréntesis que contribuyen a eliminar la ambigüedad en la interpretación de la fórmula proposicional, sin embargo, en ocasiones las fórmulas pueden ser escritas con paréntesis innecesarios. Por ejemplo la expresión $(\neg p \vee \neg q)$ contiene paréntesis y podría ser escrita $\neg p \vee \neg q$.

La representación de fórmulas proposicionales mediante árboles de formación, es otra alternativa para evitar la ambigüedad en la interpretación de la fórmula. Por ejemplo si se tiene la expresión: $p \vee (q \wedge s)$, el árbol de representación se muestra en la figura 1.1.

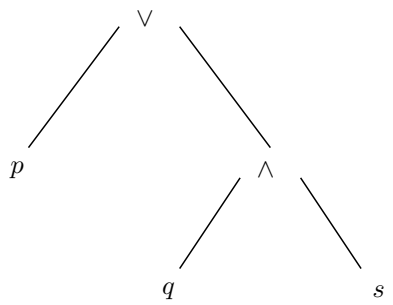


Figura 1.1: Árbol de representación de la fórmula $p \vee (q \wedge s)$

La subexpresión que está en paréntesis se representa inicialmente.

Por ejemplo, si se tiene la expresión: $(p \vee \neg q) \rightarrow (\neg q \wedge r)$, el árbol de representación será el que se muestra en la figura 1.2:

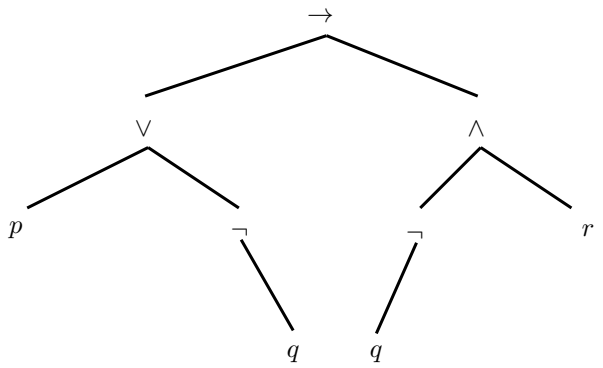


Figura 1.2: Árbol de representación de la fórmula $(p \vee \neg q) \rightarrow (\neg q \wedge r)$

En este caso, el operador principal de la expresión es la implicación y por ello se ubica como el nodo principal del árbol de formación, el cual a su vez tiene dos subexpresiones, denominadas subárbol izquierdo y subárbol derecho, cada uno de los cuales a su vez se compone de una expresión.

La expresión: $((p \vee q) \wedge (s \rightarrow t)) \leftrightarrow (s \vee \neg t)$, tiene el siguiente árbol de representación que se ve en la figura 1.3:

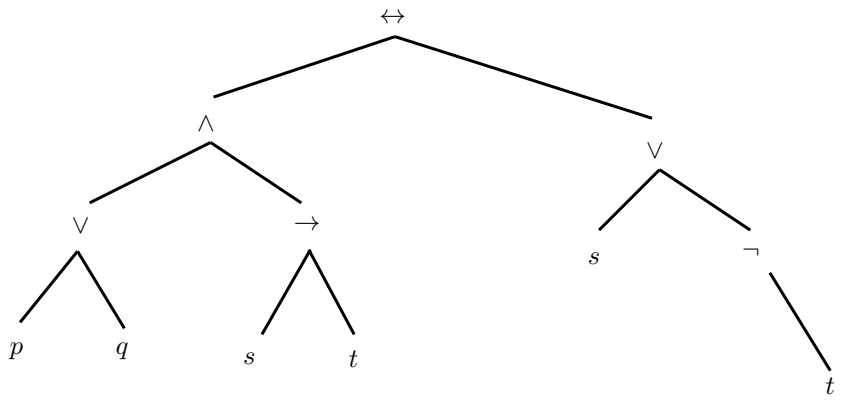


Figura 1.3: Árbol de representación de la fórmula $((p \vee q) \wedge (s \rightarrow t)) \leftrightarrow (s \vee \neg t)$

En este caso, el operador principal de la expresión es el bicondicional y por ello se ubica como el padre del árbol, este árbol tiene dos subárboles (derecho e izquierdo), cada uno de los cuales se compone de una expresión. El izquierdo tiene a su vez una expresión compuesta de subárbol izquierdo y subárbol derecho.

.:Actividad 1.9.

1. Escriba la expresión asociada al siguiente árbol de formación. (figura 1.4)

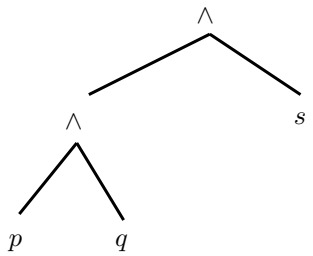


Figura 1.4

2. Dadas las siguientes fórmulas proposicionales, dibuje su árbol de formación:

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

3. Dados los siguientes árboles de formación, escriba sus correspondientes fórmulas proposicionales. (figuras 1.5 y 1.6)

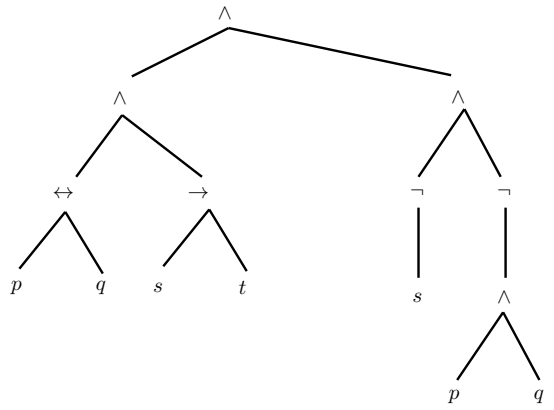


Figura 1.5

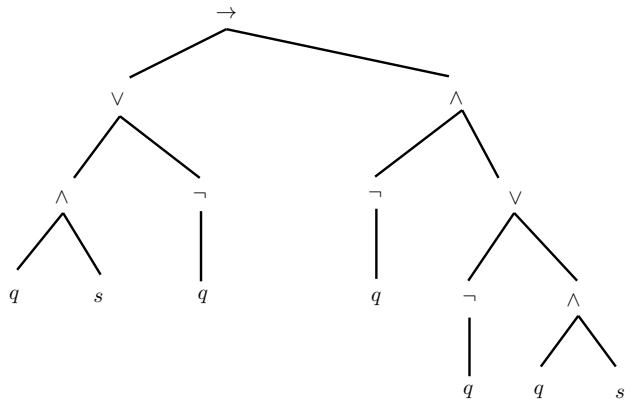


Figura 1.6

1.7 Notación prefija

Una fórmula proposicional puede ser presentada e interpretada sin ambigüedades siempre y cuando se tenga establecida una prioridad para los operadores. La notación prefija permite representar fórmulas proposicionales de forma tal que los operadores que la componen, anteceden a los átomos de la fórmula. Estos operadores deben ser binarios.

Esta forma de escribir fórmulas es útil cuando se quiere programar pilas de instrucciones, además, permiten eliminar los paréntesis, es decir, la nueva fórmula sólo contiene operadores lógicos y letras proposicionales. La notación prefija fue desarrollada por el lógico polaco Jan Łukasiewicz ([Korfhage, 1970] y [Caicedo, 1990]) en los años 50, razón por la cual se le conoce como notación polaca.

Inicialmente, la notación polaca no usaba los mismos símbolos para expresar las fórmulas. Łukasiewicz utilizó otras letras para representar los operadores lógicos: (tabla 1.8)

1.7.1 Expresiones en prefijo

Para transformar una fórmula proposicional a su equivalente en prefijo, es posible establecer una serie de pasos. Por ejemplo, para la fórmula: $a \wedge b$, para expresarla en en prefijo, se deben realizar los

Nombre	Notación sufija	Notación polaca
Negación	$\neg p$	Np
Disyunción	$p \vee q$	Apq
Conjunción	$p \wedge q$	Kpq
Condicional	$p \rightarrow q$	Cpq
Bicondicional	$p \leftrightarrow q$	Epq

Tabla 1.8: Símbolos de la notación polaca

siguientes pasos:

1. Se debe agrupar los elementos de la fórmula, teniendo en cuenta la prioridad de los operadores $(a \wedge b)$
2. Se debe escribir en prefijo la fórmula proposicional: $\wedge ab$

Por ejemplo, se desea transformar la fórmula: $a \vee (b \wedge c)$, a su equivalente en prefijo:
Se convierte inicialmente a prefijo la fórmula que está entre paréntesis:

$$a \vee (\wedge bc)$$

Considerando la expresión $(\wedge bc)$ como un operando, la expresión en prefijo queda

$$\vee a \wedge bc \qquad \text{(Fórmula en prefijo)}$$

Dada la siguiente fórmula: $(p \rightarrow q \vee s) \rightarrow (q \leftrightarrow r \wedge t)$, se debe transformar a su equivalente en prefijo.

En este caso es necesario agrupar los átomos de la fórmula teniendo en cuenta la precedencia de los operadores. La fórmula queda expresada de la siguiente manera:

$$(p \rightarrow (q \vee s)) \rightarrow (q \leftrightarrow (r \wedge t))$$

Primero se aplica el procedimiento de transformar a prefijo las expresiones que están con paréntesis más internos:

$$(p \rightarrow (\vee qs)) \rightarrow (q \leftrightarrow (\wedge rt))$$

Posteriormente es posible eliminar de la fórmula los paréntesis más internos:

$$(p \rightarrow \vee qs) \rightarrow (q \leftrightarrow \wedge rt)$$

Luego se trasforma a prefijo tanto la fórmula $(p \rightarrow \vee qs)$ como $(q \leftrightarrow \wedge rt)$, quedando de la siguiente manera:

$$(\rightarrow p \vee qs) \rightarrow (\leftrightarrow q \wedge rt)$$

Finalmente se transforma la fórmula en prefijo y la misma queda expresada de la siguiente manera:

$$\rightarrow\rightarrow p \vee qs \leftrightarrow q \wedge rt \qquad \text{(Fórmula en prefijo)}$$

Dada la siguiente fórmula:

$$(a \vee b) \wedge (b \rightarrow a)$$

La prioridad está establecida mediante la agrupación de los operandos por paréntesis:

$$(\vee ab) \wedge (\rightarrow ba)$$

Las expresiones $(\vee ab)$ y $(\rightarrow ba)$ son consideradas operandos, entonces se procede a transformar la fórmula a prefijo. Se eliminan los paréntesis existentes en la fórmula.

$$\wedge \rightarrow ab \rightarrow ba \qquad (\text{ Fórmula en prefijo})$$

En este nuevo ejemplo, se transformará la expresión a su equivalente en prefijo:

$$(p \leftrightarrow q) \vee ((q \rightarrow r) \wedge (r \rightarrow p))$$

$$(\leftrightarrow pq) \vee ((\rightarrow qr) \wedge (\rightarrow rp))$$

$$(\leftrightarrow pq) \vee (\wedge \rightarrow qr \rightarrow rp)$$

$$\vee \leftrightarrow pq \wedge \rightarrow qr \rightarrow rp \quad (\text{Fórmula en prefijo})$$

Ahora bien, dada la nueva fórmula: $(p \vee q) \wedge r \wedge s$. Se inicia agrupando por paréntesis teniendo en cuenta la prioridad de operadores y posteriormente se resuelve la fórmula a prefijo.

$$(((p \vee q) \wedge r) \wedge s)$$

$$(((\vee pq) \wedge r) \wedge s)$$

$$((\wedge \vee pqr) \wedge s)$$

$$\wedge \wedge \vee pqrs \qquad (\text{Fórmula en prefijo})$$

Transformar a prefijo la siguiente fórmula proposicional: $p \wedge (q \vee s) \wedge t \wedge (p \leftrightarrow q)$
La fórmula se agrupa por medio de paréntesis:

$$(((p \wedge (q \vee s)) \wedge t) \wedge (p \leftrightarrow q))$$

$$((p \wedge (\vee qs)) \wedge t) \wedge (\leftrightarrow pq)$$

$$((\wedge p \vee qs) \wedge t) \wedge (\leftrightarrow pq)$$

$$(\wedge \wedge p \vee qst) \wedge (\leftrightarrow pq)$$

$$\wedge \wedge \wedge p \vee qst \leftrightarrow pq \qquad (\text{Fórmula en prefijo})$$

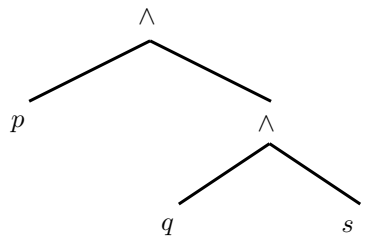


Figura 1.7: Árbol de formación de $p \wedge (q \wedge s)$

1.7.2 Árbol de formación

Una fórmula proposicional bien formada puede representarse mediante un árbol de formación. Por ejemplo la fórmula: $p \wedge (q \wedge s)$, se expresa mediante un árbol de formación de la siguiente forma: (figura 1.7).

Un árbol de formación se puede construir a partir de la transformación a su equivalente en prefijo. Por ejemplo si se tiene la fórmula $p \rightarrow q$, primero se transforma a notación en prefijo: $\rightarrow pq$. El operador $\leftrightarrow$ (primer elemento) se inserta como la raíz del árbol de formación. Posteriormente se insertan por la izquierda elementos a la fórmula hasta que se encuentre un operador, para este caso se inserta el átomo p. Posteriormente se regresa en el árbol hasta el padre del nodo insertado y se inserta el siguiente elemento en la derecha del nodo. Si el elemento insertado fue un operador, este se inserta a la izquierda. Si el elemento insertado fue un átomo, se regresa en la altura del árbol y se inserta el siguiente elemento a la derecha. Este proceso continúa hasta que se agote la expresión escrita en prefijo. El árbol de formación de la expresión queda como se muestra a continuación: (figura 1.8)

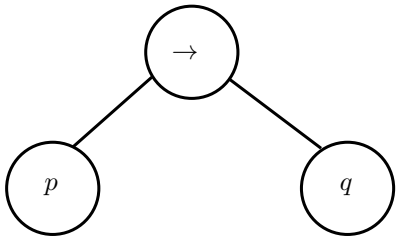


Figura 1.8: Árbol de formación de $p \rightarrow q$

El árbol de formación para la fórmula: $p \wedge q \vee r$, se construye a partir de su equivalente en prefijo. Inicialmente se agrupa la expresión de acuerdo a la precedencia de los operadores:

$$((p \wedge q) \vee r)$$

$$((\wedge pq) \vee r)$$

$$\vee \wedge pqr \qquad \text{(Fórmula en prefijo)}$$

La siguiente figura muestra el árbol de formación para la expresión: $p \wedge q \vee r$ (figura 1.9)

La fórmula: $r \wedge p \rightarrow (s \leftrightarrow r) \wedge p$, en su equivalente en prefija es:

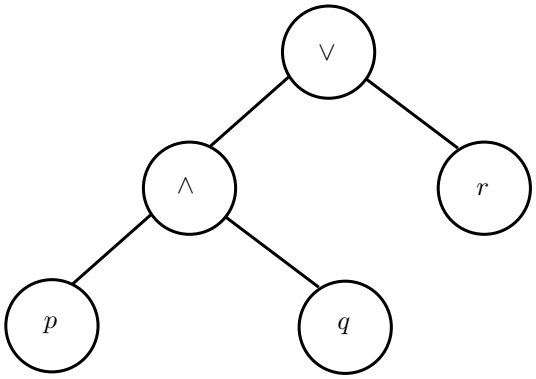


Figura 1.9: Árbol de formación para la expresión: $p \wedge q \vee r$

$$(r \wedge p) \rightarrow ((s \leftrightarrow r) \wedge p)$$
$$(\wedge rp) \rightarrow ((\leftrightarrow sr) \wedge p)$$
$$(\wedge rp) \rightarrow (\wedge \leftrightarrow srp)$$
$$\rightarrow \wedge rp \wedge \leftrightarrow srp$$

El árbol de formación de la fórmula es: (figura 1.10)

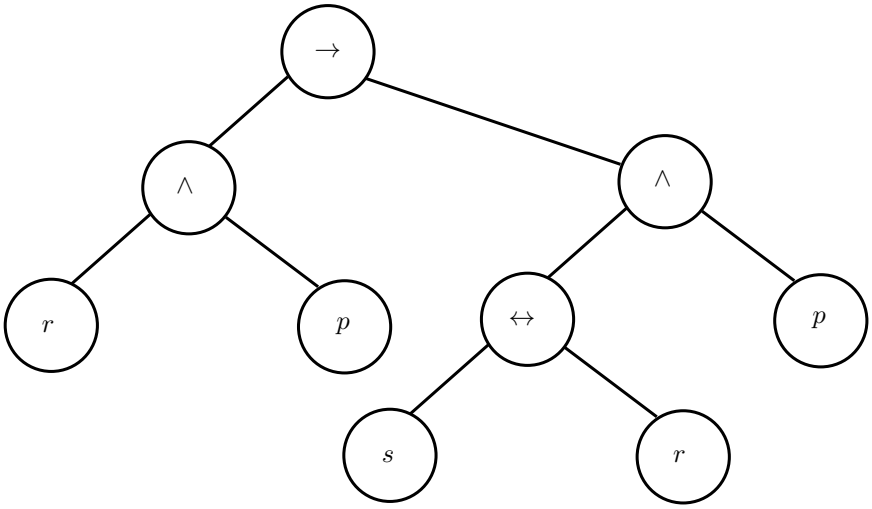


Figura 1.10: Árbol de formación para la expresión: $r \wedge p \rightarrow (s \leftrightarrow r) \wedge p$

.:Actividad 1.10.

1. Eliminar todos los paréntesis posibles de las siguientes fórmulas:
   - $(((p \rightarrow q) \vee r) \rightarrow (p \wedge \neg p))$
   - $(\neg(p \wedge q) \rightarrow (q \wedge r))$

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

2. Escribir con paréntesis las siguientes fórmulas:

- $p \to q \leftrightarrow r \vee s$
- $q \to \neg p \vee r \vee s$
- $p \vee q \leftrightarrow \neg r \vee s$
- $q \wedge \neg q \vee p \to r$

3. Transformar a prefija las siguientes expresiones infijas:

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

1.8 Tablas de verdad

Las tablas de verdad son un procedimiento utilizados para determinar el valor de verdad o falsedad de una formula proposicional simple o compuesta. El procedimiento consiste en ubicar todas las posibilidades de certeza o falsedad en forma de una tabla, posibilitando el análisis de cada una de las opciones que aparecen en ellas.

Para construir una tabla de verdad, se debe establecer las posibles combinaciones de la fórmula proposicional, es decir, en cuántas formas diferentes pueden combinarse los valores de verdad asignados a las fórmulas atómicas que las componen. Si p es una fórmula atómica, p sólo tiene dos posibles combinaciones (V) o (F). Si p tiene dos fórmulas atómicas, existen cuatro combinaciones posibles. Si p tiene tres fórmulas atómicas, sus valores de verdad se pueden combinar de ocho formas diferentes, y así sucesivamente. Así, si p tiene n fórmulas atómicas, habrá 2^n combinaciones posibles.

Las tablas de verdad se construyen a partir de la interpretación que se realiza a los operadores lógicos y, o, si... entonces..., si y sólo si... A continuación se presentan las tablas de verdad para los conectores lógicos presentados en la sección anterior.

1.8.1 Tabla de verdad de la conjunción

La conjunción es verdadera si y solo si las dos proposiciones simples que la conforman son verdaderas. La conjunción es equivalente a la intersección de dos conjuntos, además debe cumplir con la propiedad conmutativa y asociativa.

La tabla de verdad de la conjunción se representa de la siguiente forma:

A continuación se muestra un ejemplo del uso de la tablas de verdad, extraído de (Cardona, Hernández & Jaramillo, 2010), en este caso la tabla de verdad de la conjunción, el cual parte de una expresión compuesta:

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Tabla 1.9: Tabla de verdad de la conjunción

- Marina cocina arroz con coco.

Pueden suceder entonces las siguientes combinaciones:

1. Que Marina tenga el arroz y tenga el coco.
2. Que Marina tenga el arroz y no tenga el coco.
3. Que Marina no tenga el arroz y tenga el coco.
4. Que Marina no tenga el arroz y no tenga el coco.

Solo en el primer caso Marina puede preparar arroz con coco, debido a que para la preparación se tiene tanto el arroz como el coco. En los otros casos no podrá preparar arroz con coco. Estas combinaciones se pueden aplicar en la anterior tabla de verdad.

.:Actividad 1.11.

1. Construya la tabla de verdad para la formula proposicional: $\neg(\neg p \wedge \neg\neg s)$.
2. Complete la tabla de verdad para la siguiente fórmula proposicional: $(\neg p \wedge \neg q) \wedge r$

p	q	r	$\neg p$	$\neg q$	$(\neg p \wedge \neg q)$	$(\neg p \wedge \neg q) \wedge r$
V	V	V				
V	V	F				
V	F	V				
V	F	F				
F	V	V				
F	V	F				
F	F	V				
F	F	F				

1.8.2 Tabla de verdad de la disyunción inclusiva

La disyunción inclusiva es falsa si las dos proposiciones simples que la conforman son falsas. La disyunción ofrece la alternativa tanto que sea una proposición o la otra, como que sean ambas proposiciones. Es equivalente a la operación de unión de dos conjuntos.

La tabla de verdad de la disyunción inclusiva se representa de la siguiente forma:

A continuación se muestra el uso de la tabla de verdad de la disyunción inclusiva, que parte de una expresión compuesta:

- En una empresa de software requiere un ingeniero de desarrollo que tenga certificación en Cubic o en experiencia en Flic.

Pueden darse cuatro posibilidades:

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

Tabla 1.10: Tabla de verdad de la disyunción

1. Tiene certificación en Cubic o en experiencia en Flic
2. Tiene certificación en Cubic o no tiene experiencia en Flic.
3. No tiene certificación en Cubic o tiene experiencia en Flic.
4. No tiene certificación en Cubic o no tiene experiencia en Flic.

El ingeniero es aceptado en el primer caso porque tiene certificación y posee experiencia. En los casos 2 y 3 también es aceptado porque el ingeniero cumple con al menos una de las dos. Sólo en el caso 4 el ingeniero no es aceptado porque no dispone de ninguno de los dos requerimientos solicitados por la empresa de software.

.:Actividad 1.12.

1. Construya la tabla de verdad para la formula proposicional: $\neg(\neg p \vee \neg q)$
2. Complete la tabla de verdad para la siguiente fórmula proposicional: $(p \vee \neg q) \wedge \neg r$

p	q	r	$\neg p$	$\neg q$	$\neg r$	$(p \vee \neg q)$	$(p \vee \neg q) \wedge \neg r$
V	V	V					
V	V	F					
V	F	V					
V	F	F					
F	V	V					
F	V	F					
F	F	V					
F	F	F					

1.8.3 Tabla de verdad de la disyunción exclusiva

La disyunción exclusiva es verdadera sólo si una de las dos proposiciones simples que la conforman es verdadera. No pueden ser verdaderas o falsas ambas proposiciones al mismo tiempo.

La tabla de verdad de la disyunción exclusiva se representa de la siguiente forma:

p	q	$p \veebar q$
V	V	F
V	F	V
F	V	V
F	F	F

Tabla 1.11: Tabla de verdad de la disyunción exclusiva

A continuación se muestra el uso de la tabla de verdad de la disyunción exclusiva:

- Un cliente solicita una tarjeta de ahorros, el banco le asignará un número de tarjeta que termine en número par o número impar.

Pueden darse cuatro posibilidades:

1. Que el número de tarjeta termine en par o termine impar.
2. Que el número de tarjeta termine en par o no termine en impar.
3. Que el número de tarjeta no termine en par o termine en impar.
4. Que el número de tarjeta no termine en par o no termine en impar.

El primer caso no es posible porque no pueden ocurrir a la vez ambas situaciones. Los casos 2 y 3 pueden ocurrir, ya que sólo se puede asignar al final un número par o impar. En el caso 4 no es posible que ocurran ambas situaciones a la vez.

.:Actividad 1.13.

Complete la tabla de verdad para la siguiente fórmula proposicional: $\neg(\neg p \veebar q) \vee \neg r$.

p	q	r	$\neg q$	$\neg r$	$\neg(\neg p \veebar q)$	$\neg(\neg p \veebar q) \vee \neg r$
V	V	V				
V	V	F				
V	F	V				
V	F	F				
F	V	V				
F	V	F				
F	F	V				
F	F	F				

1.8.4 Tabla de verdad del condicional

El condicional es falso cuando la proposición antecedente es verdadera y la proposición consecuente es falsa. En cualquier otro caso la proposición es verdadera.

La tabla de verdad del condicional se representa de la siguiente forma:

p	q	$p \rightarrow q$
V	V	V
V	F	F
F	V	V
F	F	V

Tabla 1.12: Tabla de verdad del condicional

El uso de la tabla de verdad del condicional, a partir de la siguiente expresión:

- Si Andrea es Ingeniera de Sistemas entonces construye software.

Pueden darse cuatro posibilidades:

1. Andrea es Ingeniera de Sistemas, entonces construye software.
2. Andrea es Ingeniera de Sistemas, entonces no construye software.

3. Andrea es no es Ingeniera de Sistemas, entonces construye software.

4. Andrea es no es Ingeniera de Sistemas, entonces no construye software.

De acuerdo a las anteriores posibilidades se puede afirmar:

- En el caso 1, si Andrea es Ingeniera de Sistemas, entonces construye software. Esta afirmación es verdadera.
- En el caso 2, si Andrea es Ingeniera de Sistemas, entonces no construye software. Esta afirmación es falsa porque para ser Ingeniera de sistemas, es necesario que Andrea construya software.
- En el caso 3, si Andrea es no es Ingeniera de Sistemas, entonces construye software. Esta afirmación es verdadera porque alguien puede construir software sin ser Ingeniero de sistemas.
- En el caso 4, si Andrea no es Ingeniera de Sistemas, entonces no construye software. Esta afirmación es verdadera por deducción directa.

Existen diferentes formas de representar la conectiva condicional: forma directa y la forma recíproca. La forma directa tiene la estructura $p \to q$, en este caso se tiene primero el antecedente y luego el consecuente. Por ejemplo las expresiones tienen la forma directa:

- El Barcelona gana todos los partidos, por lo tanto es el líder.
- Estamos en invierno, en consecuencia hace frio.
- Ganó el parcial, se deduce que estudió.
- El estado embargo las cuentas, se infiere que el ciudadano no pago impuestos.

La forma recíproca tiene la estructura $q \to p$, se invierte el antecedente y el consecuente. Por ejemplo las expresiones tienen la forma recíproca.

- Se terminó el partido puesto que no paró la lluvia.
- Ganaré el concurso si cumplo con todos los requisitos
- Haré inversión en la empresa si gano la convocatoria pública.

.:Actividad 1.14.

Complete la tabla de verdad para la siguiente fórmula proposicional: $(\neg p \to r) \veebar \neg q$

p	q	r	$\neg p$	$\neg q$	$\neg r$	$(\neg p \to r)$	$(\neg p \to r) \veebar \neg q$
V	V	V					
V	V	F					
V	F	V					
V	F	F					
F	V	V					
F	V	F					
F	F	V					
F	F	F					

p	q	$p \leftrightarrow q$
V	V	V
V	F	F
F	V	F
F	F	V

Tabla 1.13: Tabla de verdad del bicondicional

1.8.5 Tabla de verdad del bicondicional

El bicondicional es verdadero cuando las proposiciones que lo conforman poseen el mismo valor de verdad. La tabla de verdad del condicional se representa de la siguiente forma: (ver Tabla 1.13)

Por ejemplo:

- Un número es compuesto si y sólo si tiene más de dos divisores.

Puede suceder que para que un número sea compuesto es necesario que tenga más de dos divisores, y es suficiente que tenga más de dos divisores para que sea compuesto, por lo tanto el bicondicional es verdadero cuando las dos proposiciones que lo conforman son verdaderas. Por otro lado, si un número no es compuesto es porque no tiene más de dos divisores, y es suficiente que no tenga más de dos divisores para que no sea compuesto. El bicondicional también es evaluado como verdadero cuando las dos proposiciones que lo conforman son falsas.

1.9 Otras tablas de verdad

Aparte de los tablas de verdad, definidas anteriormente, se pueden construir nuevas tablas de verdad, definidas con base a las operaciones binarias básicas, y son las siguientes.

1.9.1 Tabla de verdad barra de Sheffer

Esta operación indica que " p es incompatible con q" y es equivalente a $\neg(p \wedge q)$

La tabla de verdad de Shefer se representa de la siguiente forma:

p	q	$p \mid q$
V	V	F
V	F	V
F	V	V
F	F	V

Tabla 1.14: Tabla de verdad de la barra de Sheffer

1.9.2 Tabla de verdad de la flecha de Peirce

La cual indica que "ni p, ni q" está operación equivale a $\neg(p \vee q)$

1.9.3 Implicación conversa

Esta operación actua igual que el condicional tradicional, sólo que el antecedente es q y el consecuente p.

p	q	$p \downarrow q$
V	V	F
V	F	F
F	V	F
F	F	V

Tabla 1.15: Tabla de verdad de La barra de Peirce

p	q	$p \leftarrow q$
V	V	V
V	F	V
F	V	F
F	F	V

Tabla 1.16: Tabla de verdad de la implicación conversa

.:Actividad 1.15.

- Complete la tabla de verdad para la siguiente fórmula proposicional: $(\neg p \leftrightarrow r) \rightarrow \neg q$

p	q	r	$\neg p$	$\neg q$	$\neg r$	$(\neg p \leftrightarrow r)$	$(\neg p \leftrightarrow r) \rightarrow \neg q$
V	V	V					
V	V	F					
V	F	V					
V	F	F					
F	V	V					
F	V	F					
F	F	V					
F	F	F					

- Construya la tabla de verdad de los siguientes enunciados:
  - $(p|\neg q) \downarrow (q \downarrow \neg r)$
  - $(p \leftarrow q) \rightarrow (q|\neg(p \leftarrow q))$
  - $(q \leftarrow \neg r) \leftarrow (p \wedge (\neg r|p))$
  - $p \downarrow ((r \leftarrow \neg r) \downarrow (\neg p \downarrow q))$
  - $(p \wedge q) \downarrow ((q \vee r)|((p \wedge r) \leftarrow (p \leftrightarrow r)))$
  - $(p \leftarrow (q|s)) \downarrow r$
- Construya la tabla de verdad de los siguientes enunciados y trate de deducir una conclusión:
  - $p \leftarrow \neg p$
  - $p|\neg p$
  - $p \downarrow \neg p$

1.10 Formalización de proposiciones

La formalización de las proposiciones consiste en una representación formal de una expresión en lenguaje natural. Esta formalización se realiza a partir del uso de átomos unidos por operadores

lógicos. La formalización permite aplicar reglas de deducción que permiten determinar la validez de una o varias proposiciones.

Por ejemplo si se tiene la expresión: Si Deportivo gana, entonces Independiente clasifica.

La expresión se puede representar mediante las proposiciones atómicas:

- p: Deportivo gana
- q: Independiente clasifica

Donde la expresión se formaliza mediante: $p \rightarrow q$

En la tabla 1.17, se presenta un conjunto de proposiciones y su equivalente en un enunciado en lenguaje natural.

Proposiciones	**Enunciado**
$\neg q$	Julián no práctica deporte
$\neg p$	Martha no se levanta a tiempo
$p \vee q$	Mario prepara comida o escucha música
$p \wedge q$	Sandra prepara comida y escucha música
$q \vee \neg p$	Lorena prepara comida o no escucha música
$\neg p \wedge \neg q$	José no prepara comida y no escucha música
$\neg\neg q$	No es cierto que Victoria no escucha música
$p \leftrightarrow q$	Álvaro prepara comida, si y sólo si, escucha música
$p \rightarrow q$	Si Martin prepara comida, entonces escucha música
$p \rightarrow \neg q$	Si Felipe prepara comida, entonces no escucha música
$q \rightarrow p$	Si Fernando escucha música, entonces prepara comida

Tabla 1.17: Proposiciones y su equivalente en un enunciado en lenguaje natural.

Es posible a partir de una expresión, determinar el valor de verdad o falsedad de una fórmula proposicional, por ejemplo si se tiene:

- Si Andrea no realiza la ponencia, entonces no obtiene la bonificación.

Deseamos saber cuándo esta expresión es verdadera y cuando es falsa. Para ello definimos:

- p: Andrea realiza la ponencia.
- q: Andrea obtiene la bonificación

La expresión entonces se puede expresar en lenguaje proposicional mediante la fórmula: $\neg p \rightarrow \neg q$.

La tabla de verdad asociada a la anterior expresión, contiene dos variables proposicionales y teniendo en cuenta que una tabla de verdad con n variables proposicionales tiene 2^n proposiciones, para este caso $2^2 = 4$ asignaciones. Estas combinaciones se pueden representar mediante la tabla de verdad: (tabla 1.18)

Para comprender las combinaciones de la tabla de verdad se analiza la expresión:

p	q	$\neg p$	$\neg q$	$\neg p \rightarrow \neg q$
V	V	F	F	V
V	F	F	V	V
F	V	V	F	F
F	V	V	V	V

Tabla 1.18: Tabla de verdad de $\neg p \rightarrow \neg q$

- Si Andrea no realiza la ponencia, entonces no obtiene la bonificación.

Puede suceder que:

- Andrea no realiza la ponencia, entonces no obtiene la bonificación.
- Andrea no realiza la ponencia, entonces obtiene la bonificación.
- Andrea realiza la ponencia, entonces no obtiene la bonificación.
- Andrea realiza la ponencia, entonces obtiene la bonificación.

Analicemos cada una de las siguientes posibilidades:

- En el caso 1, si Andrea no realiza la ponencia, entonces no obtiene la bonificación. Esta afirmación es verdadera.
- En el caso 2, si Andrea no realiza la ponencia, entonces obtiene la bonificación. Esta afirmación es verdadera, porque Andrea puede ganar obtener la bonificación si realizar la ponencia
- En el caso 3, si Andrea realiza la ponencia, entonces no obtiene la bonificación. Esta afirmación es falsa, porque si Andrea realiza la ponencia entonces debe recibir la bonificación.
- En el caso 4, Andrea realiza la ponencia, entonces obtiene la bonificación. Esta afirmación es verdadera.

Para el siguiente ejemplo se desea averiguar cuándo es verdadera la siguiente expresión:

- El departamento de planeación progresa si y sólo si se tiene crecimiento económico y no existe la corrupción.

En este caso definimos las expresiones atómicas de la siguiente manera:

- p: El departamento de planeación progresa
- q: El departamento de planeación tiene crecimiento económico
- r: En el departamento de planeación existe la corrupción

La representación de la expresión en lógica proposicional es: $p \leftrightarrow (q \wedge \neg r)$

La tabla de verdad de esta expresión contiene tres variables p,q,r, tenemos que $2^3 = 8$ asignaciones. Representemos las anteriores combinaciones mediante una tabla de verdad: (tabla 1.19)

- Las posibilidades en las cuales la expresión “El departamento de planeación progresa si y sólo si se tiene crecimiento económico y no existe la corrupción”, se hace verdadera son las siguientes:

p	q	r	$\neg r$	$(q \wedge \neg r)$	$p \leftrightarrow (q \wedge \neg r)$
V	V	V	F	F	F
V	V	F	V	V	V
V	F	V	F	F	F
V	F	F	V	F	F
F	V	V	F	F	V
F	V	F	V	V	F
F	F	V	F	F	V
F	F	F	V	F	V

Tabla 1.19: Tabla de verdad de $p \leftrightarrow (q \wedge \neg r)$

- El departamento planeación progresa si y sólo si se tiene crecimiento económico y no existe la corrupción.
- El departamento planeación no progresa si y sólo si se tiene crecimiento económico y no existe la corrupción.
- El departamento de planeación no progresa si y sólo si no se tiene crecimiento económico y no existe la corrupción.
- El departamento de planeación no progresa si y sólo si no se tiene crecimiento económico y existe la corrupción.

A continuación se formalizan una serie de expresiones mediante lógica proposicional.

- Las plantas, como los animales, son seres vivos.

$$p \wedge q$$

- Las personas sólo están desesperadas si no actúan normalmente o se dejan llevar por las preocupaciones.

  Para traducir esta afirmación en lógica de proposiciones es necesario identificar las proposiciones atómicas, p: "Las personas solo están desesperadas", q: "las personas actúan normalmente", r "las personas se dejan llevar por las preocupaciones". Finalmente se tiene la representación lógica:

$$p \rightarrow (\neg q \vee r)$$

  En este caso la expresión también puede representarse: $p \rightarrow \neg q \vee r$, por la prioridad definida para los operadores.

- Si el tiempo de ejecución del algoritmo es exponencial, será porque el estudiante no realizó el proceso de análisis de forma correcta o por errores en el cálculo final.

  Las proposiciones atómicas de la expresión son:

  p : El tiempo de ejecución del algoritmo es exponencial

  q : El estudiante no realizó el proceso de análisis de forma correcta

  r : El estudiante tuvo errores en el cálculo final

La representación lógica es: $(\neg q \vee r) \rightarrow p$

- Andrea cancela análisis de algoritmos o estructura de datos, pero no ambas; no obstante si Andrea cancela análisis de algoritmos, tampoco cancela estructuras de datos.

  Las proposiciones atómicas de la expresión son:

  p : Andrea cancela análisis de algoritmos

  q : Andrea cancela estructura de datos

  La representación lógica es: $(p \vee q) \wedge (\neg p \vee \neg q) \wedge (\neg p \rightarrow \neg q)$

  En este ejemplo se tiene que $(p \vee q)$ corresponde a "Andrea cancela análisis de algoritmos", la palabra pero denota el operador $\wedge$, "no ambos" se representa mediante la expresión $(\neg p \vee \neg q)$, la palabra "no obstante" denota el operador $\wedge$, finalmente "si Andrea cancela análisis de algoritmos, tampoco cancela estructuras de datos".

- Estar o no estar cuando ser o no ser

  Las proposiciones atómicas de la expresión son:

  q : estar

  p : ser

  La representación lógica es: $(q \vee \neg q) \rightarrow (p \vee \neg p)$

- La causa del bajo rendimiento académico de los estudiantes en la Universidad es que se está implementando un modelo educativo erróneo en la enseñanza media. A pesar de eso, el efecto de la existencia de dicho bajo rendimiento académico de los estudiantes es que la deserción estudiantil se ha incrementado.

  Las proposiciones atómicas de la expresión son:

  p : La causa del bajo rendimiento académico de los estudiantes en la Universidad

  q : se está implementando un modelo educativo erróneo en la enseñanza media

  r : la deserción estudiantil se ha incrementado

  La representación lógica es: $(q \rightarrow p) \wedge (p \rightarrow r)$

- El software funciona correctamente solo cuando se introducen buenas prácticas, además, el software no es seguro a menos que no se introduzcan técnicas de encriptación.

  Las proposiciones atómicas de la expresión son:

  p : el software funciona correctamente

  q : se introducen buenas prácticas

  r : el software es seguro

  s : se introduzcan técnicas de encriptación

  La representación lógica es: $(p \rightarrow q) \wedge (\neg s \rightarrow \neg r)$

- El software se desarrolla en el laboratorio de ingeniería o en el de ciencias (no en ambos), sin embargo, el software se desarrolla en el laboratorio de ciencias solo si el de ingeniería está ocupado.

  Las proposiciones atómicas de la expresión son:

  p : el software se desarrolla en el laboratorio de ingeniería

  q : el software se desarrolla en el laboratorio de ciencias

  r : el laboratorio de ingeniería está ocupado

  La representación lógica es: $(p \vee q) \wedge (\neg p \vee \neg q) \wedge (q \rightarrow r)$

  De la anterior proposición se puede concluir:

  Que es suficiente que el desarrollo de software no se realice en el laboratorio de ingeniería porque el laboratorio de ciencias está ocupado.

- Los profesores están conformes cuando el estudiante gana el parcial a pesar de que no enseñan lo adecuado.

  Las proposiciones atómicas de la expresión son:

  p : Los profesores están conformes

  q : el estudiante gana el parcial

  r : Los profesores enseñan lo adecuado

  La representación lógica es: $(q \rightarrow p) \wedge \neg r$

- Para ganar análisis de algoritmos es necesario que los estudiantes ganen los parciales, además deben asistir a asesoría con los profesores, sin embargo, los alumnos no ganan la asignatura.

  Las proposiciones atómicas de la expresión son:

  p : ganar análisis de algoritmos

  q : los estudiantes deben ganar los parciales

  r : los estudiantes deben asistir a asesoría con los profesores

  s : los alumnos no ganan la asignatura

  La representación lógica es: $(q \rightarrow p) \wedge r \wedge \neg s$

- En Zambia hay inflación y no hay crecimiento económico, por tanto, Zambia no va bien.

$$(p \wedge \neg q) \rightarrow \neg r$$

- Para ganar el concurso Master Chef es suficiente cocinar bien si se cuenta con los ingredientes adecuados.

$$r \rightarrow (q \rightarrow p)$$

- Para que Julián esté alegre es necesario que tenga salud, a no ser que a Mario le dé un infarto o se enferme.

$$\neg(p \to \neg q) \to (r \vee s)$$

.:Actividad 1.16.

1. Dadas las proposiciones p y q y las siguientes expresiones, formalícelas en lógica proposicional

Expresiones	**Formalización**
como mínimo p	
p suficiente para q	
q a no ser que q	
q es suficiente para p	
a veces q, siempre p	
q siempre que p	
q a pesar de p	

2. Sean las proposiciones p: Es tenista profesional, q: Vive en Armenia, formalizar los siguientes enunciados:

Expresiones	**Formalización**
Es tenista profesional y vive en Armenia	
Es tenista profesional pero no vive en Armenia	
Es falso que sea tenista profesional o viva en Armenia	
No es tenista profesional ni vive en Armenia	
No cierto que no sea tenista profesional	
No cierto que no sea tenista profesional o no sea tenista profesional y no viva en Armenia	
Es tenista profesional si y solo si vive en Armenia o no vive en Armenia.	
q a pesar de p	

3. Formalizar a lógica proposicional los siguientes razonamientos

   - Si la complejidad computacional del algoritmo es logarítmica, será porque se implementó un algoritmo muy eficiente y porque se aplicaron técnicas de optimización de código.
   - El bajo rendimiento académico de los estudiantes no logrará mejorarse a no ser que se logre identificar su causa y se consiga encontrar alternativas adecuadas o bien para prevenirlo o para mejorarlo.
   - Si el América gana la copa o llega a las finales, será debido a que tiene muy buenos jugadores y a que tendrá a la afición a su favor.
   - A no ser que haya derechos de petición, el resultado de la convocatoria saldrá púbicamente el jueves.

4. Dada la expresión: Para que hoy gane el Barcelona basta que sea domingo, a no ser que haya mal clima y hoy hay mal clima. Y las siguientes son las proposiciones:

   - p: hoy es domingo
   - q: hoy gane el Barcelona
   - r: hay mal clima

5. Seleccionar el formalismo correcto:

   - $(((p \wedge r) \rightarrow \neg q) \wedge r$
   - $(((p \wedge \neg r) \rightarrow q) \wedge r$
   - $(\neg(p \rightarrow q) \rightarrow r) \wedge r$
   - $(p \rightarrow (r \rightarrow q)) \wedge r$

6. Dada la expresión: No es cierto que me enfermo siempre que trasnocho; solo me enfermo si tengo muchas preocupaciones. Seleccione su formalización.

   - $\neg(q \rightarrow p) \wedge (p \rightarrow q)$
   - $(\neg q \rightarrow p) \wedge (p \rightarrow r)$
   - $(\neg q \rightarrow p) \wedge (r \rightarrow p)$
   - $\neg(q \rightarrow p) \wedge (r \rightarrow p)$

1.11 Clasificación de las fórmulas de acuerdo a las tablas de verdad

Cuando se realiza la tabla de verdad de una fórmula, se obtienen tres posibilidades, la primera de ellas es que todas las posibles valuaciones, es decir, todas las filas de la tabla de verdad, sean verdaderas V, o que todas ellas fueran falsedades F o que en las posibles valuaciones se tienen una mezcla entre V y F.

Es esta sección, se hará una forma alternativa de crear las tablas de verdad, la cual consiste en escribir la fórmula y bajo cada letra proposicional y cada conectivo lógico se ubican sus correspondientes valuaciones. De esta forma, el resultado final de la tabla de verdad no está en la última columna, sino en la columna que está bajo el operador principal.

1.11.1 Tautología

Es una proposición compuesta, cuya tabla de verdad siempre es verdadera V, independientemente de los valores de verdad de las proposiciones simples que la componen. Por ejemplo la fórmula $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$ (tabla 1.20)

$\neg$	$(p$	$\wedge$	$q)$	$\leftrightarrow$	$(\neg p$	$\vee$	$\neg q)$
F	V	V	V	V	F	F	F
V	V	F	F	V	F	V	V
V	F	F	V	V	V	V	F
V	F	F	F	V	V	V	V

Tabla 1.20: Tabla de verdad de $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$

Otro ejemplo de Tautología puede ser la fórmula $(p \wedge q) \rightarrow p$, veamos la tabla 1.21:

$(p$	$\wedge$	$q)$	$\rightarrow$	p
V	V	V	V	V
V	F	F	V	V
F	F	V	V	F
F	F	F	V	F

Tabla 1.21: Tabla de verdad de $(p \wedge q) \rightarrow p$

1.11.2 Contradicción

Es una proposición compuesta cuya tabla de verdad tiene todas las posibles valuaciones falsas, independientemente de los valores de verdad de las proposiciones que la componen. Por ejemplo, la tabla de verdad de la fórmula $\neg(p \vee q) \leftrightarrow (p \vee q)$: (tabla 1.22)

$\neg$	$(p$	$\vee$	$q)$	$\leftrightarrow$	$(p$	$\vee$	$q)$
F	V	V	V	F	V	V	V
F	V	V	F	F	V	V	F
F	F	V	V	F	F	V	V
V	F	F	F	F	F	F	F

Tabla 1.22: Tabla de verdad de $\neg(p \vee q) \leftrightarrow (p \vee q)$

Otro ejemplo de contradicción lo encontramos en la fórmula $p \wedge \neg p$, su tabla de verdad la vemos en la tabla 1.23:

p	$\wedge$	$\neg$	p
V	F	F	V
F	F	V	F

Tabla 1.23: Tabla de verdad de $p \wedge \neg p$

1.11.3 Contingencia o indeterminación

Es una proposición compuesta, cuya tabla de verdad siempre posee falsedad o verdad en todas su posibles valuaciones. Tomemos por ejemplo la fórmula $(p \vee q) \rightarrow p$: (tabla 1.24)

$(p$	$\vee$	$q)$	$\rightarrow$	p
V	V	V	V	V
V	V	F	V	V
F	V	V	F	F
F	F	F	V	F

Tabla 1.24: Tabla de verdad de $(p \vee q) \rightarrow p$

1.11.4 Clasificación de la tautologías

Las tautologías pueden clasificarse de acuerdo con el operador principal de cada fórmula.

1.11.4.1 Tautologías tipo implicación

Es una tautología, cuya fórmula que tiene como conector principal un condicional. Un ejemplo de esta puede ser la fórmula $(p \wedge q) \rightarrow \neg(\neg p \vee \neg q)$: (tabla 1.25)

$(p$	$\wedge$	$q)$	$\rightarrow$	$\neg$	$(\neg p$	$\vee$	$\neg q)$
V	V	V	V	V	F	F	F
V	F	F	V	F	F	V	V
F	F	V	V	F	V	V	F
F	F	F	V	V	V	F	V

Tabla 1.25: Tabla de verdad de $(p \wedge q) \rightarrow \neg(\neg p \vee \neg q)$

1.11.4.2 Tautología tipo equivalencia

Es una tautología cuya fórmula tiene como operador principal un bicondicional. Por ejemplo la fórmula $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$, es un tautología tipo equivalencia, pues su fórmula tiene como operador principal un bicondicional. (tabla 1.26)

$\neg$	$(p$	$\wedge$	$q)$	$\leftrightarrow$	$(\neg p$	$\vee$	$\neg q)$
F	V	V	V	V	F	F	F
V	V	F	F	V	F	V	V
V	F	F	V	V	V	V	F
V	F	F	F	V	V	V	V

Tabla 1.26: Tabla de verdad de $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$

En la figura 1.11 muestra la clasificación de las fórmulas:

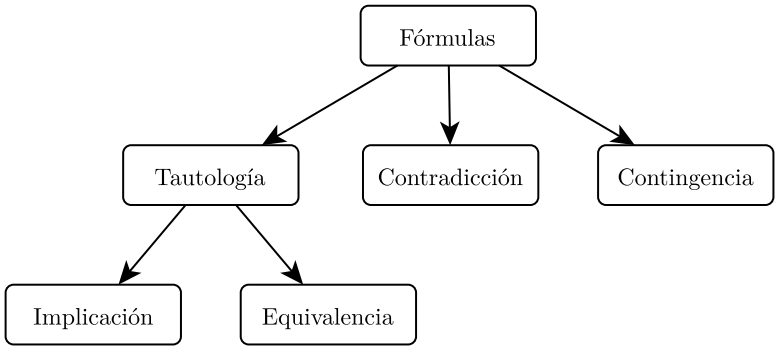


Figura 1.11: Clasificación de las fórmulas

De acuerdo a las definiciones anteriores, es posible presentar las siguientes proposiciones:

.:Proposición 1.1. *La fórmula α es lógicamente equivalente a la fórmula β si y sólo si $\alpha \leftrightarrow \beta$ es una tautología.*

Por ejemplo, en la tabla 1.26, la tabla de verdad de la fórmula $\neg(p \wedge q) \leftrightarrow (\neg p \vee \neg q)$, pero en realidad, esta expresión dice que las sub fórmulas $\underbrace{\neg(p \wedge q)}_{\alpha}$ y $\underbrace{(\neg p \vee \neg q)}_{\beta}$ son equivalentes. El tema de las equivalencia será abordado en la sección 2.6.

.:Proposición 1.2. *Si las fórmulas α y $\alpha \rightarrow \beta$ son tautologías, entonces β es una tautología.*

La proposición 1.2, dice que si una fórmula que es una tautología y además, otra fórmula que es una tautología tipo implicación, donde la sub fórmula que hace las veces de antecedente es la primera implicación, entonces la fórmula que hace las veces de consecuente debe ser otra tautología. Esto se debe a la naturaleza del condicional, que en la tabla de verdad de este operador lógico (ver sección 1.8.4), cuando el antecedente es una verdad, la única forma de obtener otra verdad es que el consecuente también sea una verdad.

.:Actividad 1.17.

- Determine si las siguientes fórmulas moleculares son tautologías, contradicciones o contingencia. En caso de ser tautología, determine si es de tipo implicación o de tipo equivalencia.
  - $(\neg p \vee q) \rightarrow (\neg q \rightarrow \neg p)$
  - $(p \wedge \neg q) \leftrightarrow \neg(\neg q \vee p)$
  - $(p \rightarrow q) \leftrightarrow (\neg q \rightarrow \neg p)$
  - $((p \rightarrow q) \wedge (q \rightarrow r)) \rightarrow (p \rightarrow r)$
  - $((p \rightarrow q) \wedge (q \rightarrow r)) \wedge \neg(p \rightarrow r)$
- Encuentre dos ejemplos para cada una de las proposiciones 1.1 y 1.2.

CAPÍTULO 2

Cálculo proposicional

> !Cuanto más chistes uno sabe, mas chistes uno se olvida.... Cuanto más chistes uno se olvida, menos chistes uno sabe. Por Logica: Cuanto menos chistes uno sabe, menos chistes uno se olvida, y cuantos menos chistes uno se olvida, mas chistes uno sabe.!
>
> Juan Francisco Verdaguer
> Humorista y actor uruguayo
> 1915 – 2001

Al finalizar el estudio de éste capítulo el estudiante estará en condiciones de:

- Entender el lenguaje proposicional a partir de un alfabeto, sintaxis y semántica.
- Aplicar el concepto de interpretación booleana para realizar la evaluación semántica de una fórmula proposicional.
- Comprender las propiedades de las fórmulas proposicionales y las principales equivalencias lógicas del cálculo proposicional.
- Emplear el método de resolución y de los tableros semánticos como mecanismo de decisión en el cálculo proposicional.
- Aplicar métodos de decisión para determinar la satisfacibilidad, validez, no validez e insatisfacibilidad de una fórmula proposicional.
- Comprender y aplicar el concepto de forma normal conjuntiva y forma normal disyuntiva.

2.1 Introducción

En el capítulo anterior se presentaron los elementos fundamentales de la lógica matemática, las formas de representación de las expresiones y las propiedades de los operadores lógicos. Así mismo, se explicó el valor de una proposición teniendo en cuenta el significado de los operadores lógicos y los signos de

agrupación. La interpretación del valor de una fórmula proposicional se trabajó por medio del método de las tablas de verdad, el cual es un método mecánico que no implica complejidad en cuando al análisis y manipulación de las fórmulas proposicionales.

En este capítulo se presentan la definición de un lenguaje proposicional basado en un alfabeto, sintaxis y semántica, a partir del cual se estructuran fórmulas bien formadas. También se trabajarán métodos de decisión para determinar propiedades de las fórmulas proposicionales, tales como la la satisfacibilidad, validez, no validez e insatisfacibilidad. El propósito es que el estudiante aplique de forma sistemática estos métodos como mecanismos eficientes de deducción en el cálculo proposicional. Los métodos de deducción que se presentarán serán el método de los tableros semánticos y de resolución proposicional. Al final del capítulo se presentaran algunas herramientas informáticas que han sido implementadas para dar soporte a los métodos de deducción anteriormente mencionados.

2.1.1 Lenguaje de la lógica proposicional

El lenguaje en la lógica proposicional (LP) permite formalizar el alfabeto, la sintaxis y la semántica, elementos a partir de los cuales se puede formalizar inequívocamente una fórmula proposicional bien formada. Inicialmente se definirá el alfabeto de la fórmula proposicional, luego se establecerá una sintaxis y finalmente una semántica para la interpretación booleana.

2.1.2 Alfabeto de la lógica proposicional

El alfabeto es el conjunto principal de donde obtenemos los componenetes básicos de construcción para una determinada frase. Por ejemplo, si queremos escribir una carta, nuestra primera idea de alfabeto seria:

$$\{a, b, c, d, \ldots, z, A, B, C, D, \ldots, Z\}$$

Si usamos sólo este alfabeto, tendríamos problemas para entender algunas partes de nuestra carta, debemos agregar los signos de puntuación:

$$\{a, b, c, d, \ldots, z, A, B, C, D, \ldots, Z, \{,\}, \{.\}, \{;\}\}$$

También otros signos como los de interrogación y admiración:

$$\{a, b, c, d, \ldots, z, A, B, C, D, \ldots, Z, \{,\}, \{.\}, \{;\}, ?, ¿, !\}$$

Si lo que deseamos es escribir un artículo en matemáticas, entonces el alfabeto debe ser más amplio, pues además de los números, debe contener otros símbolos:

$$\{a, b, c, d, \ldots, z, A, B, C, D, \ldots, Z, \{,\}, \{.\}, \{;\}, ?, ¿, !, \ldots\}$$

Junto con:

$$\left\{1, 2, 3, \ldots, 9, \alpha, \beta, \gamma, \ldots, \int, \sum, \partial, \ldots\right\}$$

Veamos la definición formal de alfabeto:

.:Definición 2.1. *Un alfabeto es un conjunto de símbolos que permite a partir de expresiones simples, la construcción de expresiones más complejas.*

Retomando el ejemplo de la cata, podemos pensar en ¿qué lenguaje vamos a usar?, en nuestro caso es el castellano, pero puede ser otro idioma como el ingles o el italiano. Así que es importante definir el lenguaje, en LP un lenguaje se puede definir como la unión entre el alfabeto y el conjunto de palabras correctamente construidas. Si escribimos nuestra carta con mala ortografía o sin usar signos de puntuación, pues es muy probable que no se entienda el mensaje que se desea transmitir.

.:Definición 2.2. *Un lenguaje formal $\Im$ es una pareja ordenada (A, ξ), donde A es el alfabeto y $\xi \in A^*$ es el conjunto de todas las expresiones bien formadas.*

En LP, el alfabeto para dar soporte a la lógica proposicional se estructura a partir de los siguientes elementos:

1. **Letras proposicionales**: $p, q, r, s, t, u, v, w, x, y, z, \ldots,$ de igual forma se usan las letras proposicionales indexadas: $p_1, p_2, \ldots, p_n, q_1, q_2, \ldots, q_n, \ldots$
2. **Operadores lógicos**:

   Unarios: $\neg$

   Binarios: $\vee$, $\wedge$, $\rightarrow$, $\leftrightarrow$

   donde
   $$\bigstar = \{\wedge, \vee, \rightarrow, \leftrightarrow\}$$
3. **Símbolos de asociación**: paréntesis izquierdo '(' y derecho ')', la coma $(,)$
4. **Valores de verdad**: Verdadero (V), Falso (F).

Asi, el alfabeto para LP toma la forma:

$$A = \{p, q, \ldots, z, \neg, \wedge, \vee, \rightarrow, \leftrightarrow, (,), V, F\}$$

.:Nota 2.1. *Se utilizan letras griegas $\alpha, \beta, \gamma, \ldots$, para denotar cadenas arbitrarias de simbolos en LP.*

Una vez definido el alfabeto, es necesario definir una estructura formal para la representación de fórmulas proposicionales. Esta estructura se realiza a partir de una especificación sintáctica.

2.1.3 Sintaxis de la lógica proposicional

Una vez definido el alfabeto básico, el siguiente paso es definir reglas de formación que garanticen que una fórmula está bien formada. La sintaxis determina las reglas que permiten la estructuración de fórmulas proposicionales bien formadas y está constituida por reglas de simbolización, las cuales se presentan a continuación ([Caicedo, 1990] y [Korfhage, 1970]):

2.1.3.1 Reglas de Formación en LP

la siguiente definición nos muestra la regla básica de construcción de fórmulas correctamente construidas en LP:

.:Definición 2.3. *Una cadena de símbolos en LP, es una fórmula bien formada (fbf) si sólo si puede obtenerse por medio de las siguientes reglas de formación:*

F_1 *Las letras proposicionales son fbf_s.*

F_2 *Si α es fbf, entonces $\neg(\alpha)$ es fbf.*

F_3 *Si α y β son fbf_s, entonces $(\alpha)\bigstar(\beta)$ es fbf. donde $\bigstar = \{\wedge, \vee, \rightarrow, \leftrightarrow\}$*

La siguiente regla, tiene implícita las tres reglas anteriores y dice que toda fbf se puede analizar por medio de fbf_s más simples.

F_4 *Toda* fbf *de LP o es una letra proposicional o tiene una de las formas* $\neg(\alpha)$, $(\alpha)\bigstar(\beta)$, *donde* α *y* β *son* fbf_s.

La primera regla nos dice que las letras proposicionales son fbf, es decir que p o q ya es una fbf. Ahora bien, autores como [Caicedo, 1990] nos dice que la forma correcta es ponerla entre paréntesis, es decir (p), sin embargo, es poco usual representar las letras proposiciones de esta forma.

De igual manera, la segunda regla nos dice que: $\neg(\alpha)$ es fbf, pero se puede simplificar como: $\neg\alpha$ si α es una letra proposicional, pues $\neg(p)$ se puede escribir sin problema como $\neg p$, pero cuando α es una fórmula compuesta si debemos conservar los paréntesis, para delimitar el alcance de la negación, recordemos que:

$$\neg(p \wedge q) \neq \neg p \wedge q$$

En la primera fórmula, el alcance de la negación es para toda la fórmula, mientras que en la segunda, sólo abarca la primera letra proposicional.

La tercera regla nos dice que: si α y β son fbf_s, entonces $(\alpha)\bigstar(\beta)$ es fbf.. Aquí las dos letras proposicionales están entre paréntesis, si α y β son letras proposiciones no hay problema, se pueden suprimir los paréntesis, por ejemplo $p \wedge q$, pero si α y β son compuestas, entonces debemos tener cuidado con el alcance de los operadores binarios. Por ejemplo: $p \wedge q \rightarrow r$ es diferente de $p \wedge (q \rightarrow r)$ y de $(p \wedge q) \rightarrow r$.

Con base en la sintaxis, las siguientes son fórmulas bien formadas:

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

No son fórmulas bien formadas las siguientes:

- $q\neg$ (error en la ubicación del operador)
- $p \wedge \neg q)$ (error en el uso del símbolo de paréntesis)
- $\neg((\neg p \vee \neg q) \rightarrow (\neg r \leftrightarrow \neg s$ (uso del símbolo de paréntesis)
- $\neg p \vee \vee p$ (no es correcto el uso de los operadores)
- $\leftrightarrow\rightarrow (\neg r \leftrightarrow \neg s)$ (no es correcto el uso de los operadores)

La sintaxis definida se extenderá para la representación de fórmulas proposicionales, mediante el uso de letras mayúsculas. Por ejemplo las siguientes fórmulas se pueden representar de la siguiente manera.

- $A : \neg(p \rightarrow q \leftrightarrow r)$
- $B : (p \vee q) \wedge \neg(q \vee q)$
- $\neg B : \neg p \wedge q$

Un átomo también puede ser representado mediante la letra mayúscula.

- $\neg A : \neg\neg p$

- $B : \neg q$
- $A : p$

Una fórmula completa también puede representar una combinación de fórmulas, por ejemplo:

- $(B \wedge A)$
- $(A \rightarrow \neg C) \wedge B$
- $(A \rightarrow B) \rightarrow (C)$

La interpretación de la fórmula parte de su interpretación semántica la cual se presenta en la siguiente sección.

2.2 Descomposición única del conectivo principal

Como hemos visto, toda fórmula tiene un conectivo principal, el cual es importante para realizar otros cálculos como el árbol de formación y la notación polaca. Para ello, veremos el teorema de descomposición única presentado por [Caicedo, 1990]:

.:Teorema 2.1 (Teorema de descomposición única de fbf)**.** *Toda fbf α es una letra proposicional o es una fórmula de la forma $\neg(\alpha')$ con α' una fbf o se puede expresar de manera única en la forma $\alpha = \alpha' \bigstar \alpha''$ con α' y α'' fbf y $\bigstar$ un conectivo binario.*

Este teorema nos muestra las tres formas básicas de una fbf, donde la fórmula puesde ser una letra proposicional, una fórmula negada o la unión de dos fbf_s unidas por un conectivo lógico.

Para el tratamiento posterior de este tema, usaremos la forma estricta de los paréntesis, Así por ejemplo, si α es una fbf su negación se escribe $\neg(\alpha)$; si α y β son fbf_s y $\bigstar$ es un operador binario, entonces lo escribiremos como $(\alpha)\bigstar(\beta)$. Todo esto con el fin de encontrar operador lógico principal.

El teorema 2.1 nos dice que podemos descomponer una fbf hasta su conectivo principal. Esto hecho se puede reducir en el siguiente algoritmo ([Caicedo, 1990]):

Algoritmo de descomposición:

1. Si la fórmula comienza por negación $\neg$, ese símbolo es el conectivo principal. De lo contrario, de lo contrario la fórmula debe empezar con un paréntesis izquierdo.
2. Recorra la fórmula de izquierda a derecha, asignando al primer paréntesis izquierdo el número 1.
3. Asigne a cada paréntesis izquierdo el valor del paréntesis anterior más 1.
4. Asigne a cada paréntesis derecho, el valor del paréntesis anterior menos 1.
5. El conectivo principal, en el símbolo inmediatamente después del primer paréntesis al que se le asigna como número el 0.

Tomemos como ejemplo la fórmula:

$$(((p) \rightarrow (q)) \rightarrow \neg((r) \wedge (q))) \vee \neg((p) \leftrightarrow ((r) \wedge (q)))$$

Como no empieza con una negación, entonces numeramos los paréntesis:

$$\underset{123\ \ 2}{(((p)} \rightarrow \underset{3\ \ 21}{(q))} \rightarrow \neg\underset{23\ \ 2}{((r)} \wedge \underset{3\ \ 210}{(q)))} \vee \neg((p) \leftrightarrow ((r) \wedge (q)))$$

Luego, el conectivo principal es $\vee$. Esto nos permite encontrar el árbol sintáctico de la fórmula: (figura 2.1)

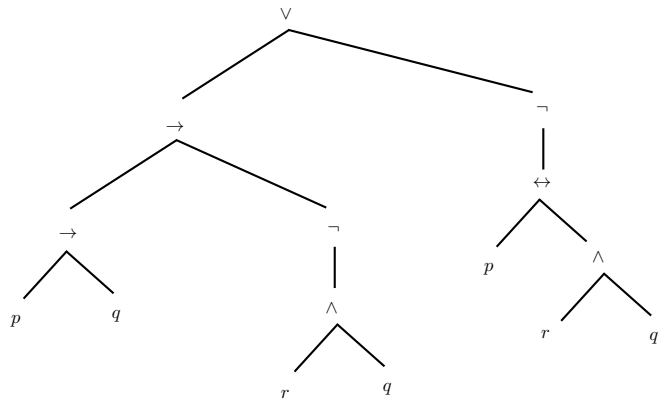


Figura 2.1: Árbol sintáctico de la fórmula $(((p) \rightarrow (q)) \rightarrow \neg((r) \wedge (q))) \vee \neg((p) \leftrightarrow ((r) \wedge (q)))$

2.3 Semántica de la lógica proposicional

Si una fórmula proposicional está bien formada, es decir es su sintaxis es correcta, es posible asignar o atribuir semánticamente el valor ya sea de verdadero o falso. Cuando a cada uno de los átomos de una fórmula A, se le asignan valores (verdadero o falso), entonces se está dando una interpretación a la fórmula, lo cual se puede representar en términos de la función:

$$v : \{p, q, \dots\} \leftrightarrow \{V, F\}$$

La semántica asigna valores de verdad a las fórmulas proposicionales, dando una interpretación a estas. Los siguientes son algunos ejemplos de interpretación:

- $v(\neg A) = F$
- $v(\neg B) = V$
- $v(A \vee B) = F$
- $v(\neg C \wedge B) = V$
- $v(A \vee \neg B) \vee (C \vee C) = V$

Por ejemplo si se tiene la fórmula $A : (\neg p \vee \neg q)$, la interpretación $v(\neg p) = F$ y $v(\neg q) = F$, asigna valores para cada uno de los átomos. La evaluación de $v(A) = F$.

Cuando a la fórmula se le asigna verdadero o falso, se afirma que se realiza una interpretación de la fórmula. Si una fórmula lógica se interpreta para todas las posibles combinaciones de sus componentes, se dice que se realiza la evaluación semántica de dicha fórmula (Castel de Haro, 2011). El número de interpretaciones para un número n de átomos que constituyen una fórmula es 2^n. En el caso de las tablas de verdad, cuando se tienen un número alto de átomos en la fórmula proposicional, el tamaño

de las tablas crece de forma exponencial, por lo que puede resultar en un método poco eficiente.

De cada uno de los operadores lógicos existen diferentes interpretaciones las cuales están dadas en función de las fórmulas. La tabla 2.1 muestra la interpretación de los operadores para las fórmulas A y B.

A	B	$\neg A$	$\neg B$	$A \wedge B$	$A \vee B$	$A \rightarrow B$	$A \leftrightarrow B$
V	V	F	F	V	V	V	F
V	F	F	V	F	V	F	V
F	V	V	F	F	V	V	V
F	F	V	V	F	F	V	F

Tabla 2.1: Interpretaciones de los operadores para las fórmulas A y B

Las interpretaciones son una analogía de los valores que se dan a cada una de las líneas que se generan en una tabla de verdad. Por ejemplo si se tiene la fórmula: $A : \neg p \vee q \wedge r$, esta puede tener la interpretación:

- $v(\neg p) = V$
- $v(q) = V$
- $v(r) = F$

Dados los valores para la fórmula A, la interpretación v apara cada uno de los átomos, permite afirmar que $v(A) = F$.

Una fórmula proposicional es posible representarla mediante un único árbol de formación. Por ejemplo si se tiene la fórmula $A : p \vee q \rightarrow p$ y las interpretaciones:

- $v(p) = F$
- $v(q) = V$

Una posible interpretación puede ser: $v(p \vee (q \rightarrow p)) = V$, la cual tendría como árbol de formación: (figura 2.2)

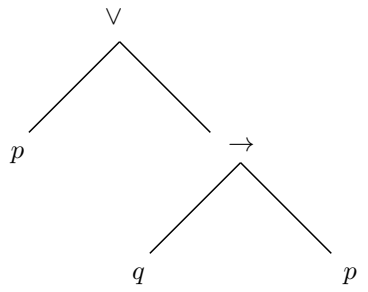


Figura 2.2: Árbol de formación de $v(p \vee (q \rightarrow p)) = V$

Esta misma fórmula se podría interpretar como: $v((p \vee q) \rightarrow p) = F$, la cual tendría como árbol de formación: (figura 2.3)

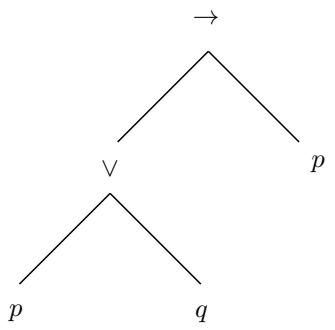


Figura 2.3: Interpretación de la fórmula $v((p \vee q) \rightarrow p) = F$

Cuando una fórmula está definida sin ambigüedades, es posible realizar una representación mediante un único árbol de formación. Por lo tanto, establecer la prioridad de los operadores y el uso de los paréntesis, garantiza una adecuada interpretación v para una fórmula proposicional.

Por ejemplo, la fórmula proposicional $A : (p \wedge q) \vee (q \wedge p)$ y las interpretaciones $v(p) = F$ y $v(q) = V$, la fórmula proposicional tiene como interpretación: $v((p \wedge q) \vee (q \wedge p)) = F$. Cuyo árbol de formación se puede representar de la siguiente manera:

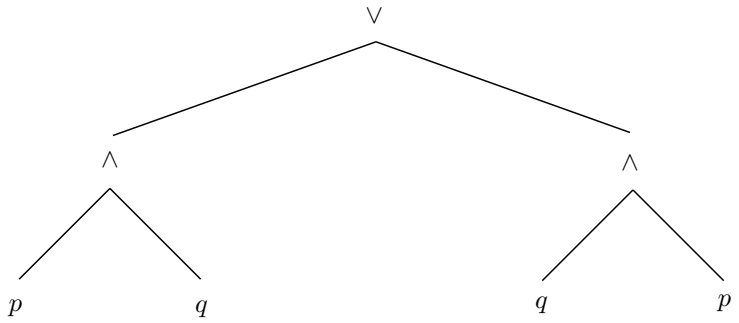


Figura 2.4: Interpretación de la fórmula $v((p \wedge q) \vee (q \wedge p)) = F$

.:Actividad 2.1.

1. Dadas las interpretaciones: $v(p) = F$, $v(q) = F$, $v(r) = V$, $v(s) = V$. Represente su árbol de formación y su interpretación final.

   - $A = p \vee q \rightarrow r \rightarrow s$
   - $B = p \wedge q \leftrightarrow \neg p \rightarrow \neg q$
   - $C = p \vee q \vee s \leftrightarrow p \wedge s \vee q$
   - $D = p \rightarrow q \leftrightarrow \neg p \wedge s$
   - $E = p \vee q \rightarrow q \wedge \neg q \vee p$
   - $F = p \wedge q \rightarrow q \vee s$

2. Dado el siguiente árbol de formación, construya su fórmula proposicional y determine su interpretación final si: $v(p) = V$, $v(q) = V$, $v(r) = F$. (ver figura 2.5)

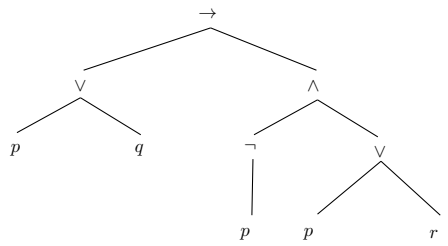


Figura 2.5

2.4 Funciones en LP

2.4.1 Función Valuación

Cuando se encuentran correctamente estructuradas las fórmulas proposicionales, es posible atribuirles valores de verdad del conjunto $\{0,1\}$ (1 equivale a Verdadero y 0 a falso), que permiten dar un significado a tales fórmulas. La valuación es tal que: $v : \xi \curvearrowright \{0,1\}$, donde ξ es la fórmula.

2.4.1.1 Negación

$$H_{\neg}(i) = \begin{cases} 1 & \text{si } v(\neg i) = 0 \\ 0 & \text{si } v(\neg i) = 1 \end{cases}$$

2.4.1.2 Conjunción

$$H_{\wedge}(i,j) = \begin{cases} 1 & \text{si } v(i) = v(j) = 1 \\ 0 & \text{cc - cualquier otro caso} \end{cases}$$

2.4.1.3 Disyunción

$$H_{\vee}(i,j) = \begin{cases} 0 & \text{si } v(i) = v(j) = 0 \\ 1 & \text{cc} \end{cases}$$

2.4.1.4 Condicional

$$H_{\rightarrow}(i,j) = \begin{cases} 0 & \text{si } v(i) = 1 \quad \text{y} \quad v(j) = 0 \\ 1 & \text{cc} \end{cases}$$

2.4.1.5 Bicondicional

$$H_{\leftrightarrow}(i,j) = \begin{cases} 1 & \text{si } v(i) = v(j) \\ 0 & \text{cc} \end{cases}$$

Tomemos por ejemplo la fórmula $(p \wedge q) \rightarrow r$, y la función $H_{\rightarrow}$, entonces:

$$H_{\wedge}((p \wedge q), r)$$

Sólo tiene dos posibles valores, 1 (verdadero), sólo cuando ambas subfórmulas sean verdaderas, en cualquier otro caso será 0 (falsas).

Si tenemos la fórmula $\neg p$, la función $H_{\neg}(p)$, retorna verdadero si p es falsa y retornará falso si p es verdadera.

2.4.2 Propiedades de la valuación

i. $v(\neg p) = 1 - v(p)$

ii. $v(p \wedge q) = min(v(p), v(q))$

iii. $v(p \vee q) = max(v(p), v(q))$

iv. $v(p \to q) = max(1 - v(p), v(q))$

vi $v(p \leftrightarrow q) = 1 - |v(p) - v(q)|$

2.4.3 Función grado

Esta función arroja el número de operadores lógicos que aparecen en una fórmula proposicional, se tiene que $gr : \xi \curvearrowright \mathbb{N}$, y esta definida de la siguiente manera:

$$gr(\alpha) = \begin{cases} 0 & \text{si } \alpha = p \\ 1 + gr(\alpha^{`}) & \text{si } \alpha = \neg(\alpha^{`}) \\ 1 + gr(\alpha^{`}) + gr(\alpha^{``}) & \text{si } \alpha = (\alpha^{`})\bigstar(\alpha^{``}) \end{cases}$$

Ejemplo. Hallar el grado de $p \to (r \vee \neg q)$

$$\begin{aligned} gr(p \to (r \vee \neg q)) &= 1 + gr(p) + gr(r \vee \neg q) \\ &= 1 + 0 + 1 + gr(r) + gr(\neg q) \\ &= 2 + 0 + 1 + gr(q) \\ &= 3 + 0 \\ &= 3 \end{aligned}$$

2.4.4 Función Subfórmula

Esta función da como resultado el conjunto de todas las subfórmulas que conforman la fórmula. se tiene que $sub : \xi \curvearrowright E \subset \xi$, y se define así:

$$sub(\alpha) = \begin{cases} \{p\} & \text{si } \alpha = p \\ sub(\alpha^{`}) \cup \{\neg(\alpha^{`})\} & \text{si } \alpha = \neg(\alpha^{`}) \\ sub(\alpha^{`}) \cup sub(\alpha^{``}) \cup \{(\alpha^{`})\bigstar(\alpha^{``})\} & \text{si } \alpha = (\alpha^{`})\bigstar(\alpha^{``}) \end{cases}$$

Hallar el conjunto subfórmula de $p \to (r \vee \neg q)$

$$\begin{aligned} sub(p \to (r \vee \neg q)) &= sub(p) \cup sub(r \vee \neg q) \cup \{p \to (r \vee \neg q)\} \\ &= \{p\} \cup sub(r) \cup sub(\neg q) \cup \{r \vee \neg q\} \cup \{p \to (r \vee \neg q)\} \\ &= \{p\} \cup \{r\} \cup sub(q) \cup \{\neg q\} \cup \{r \vee \neg q\} \cup \{p \to (r \vee \neg q)\} \\ &= \{p\} \cup \{r\} \cup \{q\} \cup \{\neg q\} \cup \{r \vee \neg q\} \cup \{p \to (r \vee \neg q)\} \\ &= \{p, q, r, \neg q, r \vee \neg q, p \to (r \vee \neg q)\} \end{aligned}$$

2.4.4.1 Ábol subfórmula

Los nodos en este árbol son todas las fórmulas que conforman la fórmula inicial y la raíz es la fórmula completa.

Por ejemplo, hallar el árbol de $p \to (r \vee \neg q)$ (figura 2.6)

2.5 Principio de inducción en fómulas proposicionales

Sea λ una propiedad que se aplica a ciertas sucesiones de simbolos tal que:

- Caso base. Cada letra proposicional tiene la propiedad λ

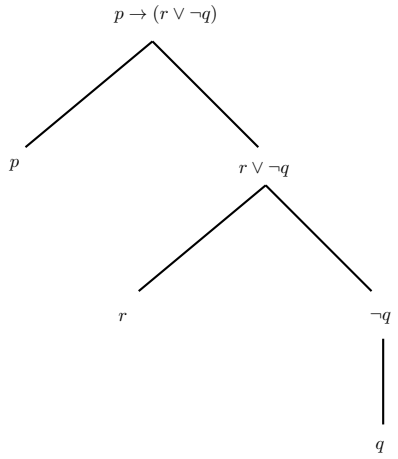


Figura 2.6: El árbol de la fórmula $p \to (r \vee \neg q)$

- Paso Inductivo.

  a. Si α es fbf que tiene la propiedad λ , entonces $\neg(\alpha)$ también la tiene.

  b. Si α y β son fbf_s que tienen la propiedad λ , entonce$(\alpha)\bigstar(\beta)$ también la tiene[1] .

Por ejemplo: verifique, utilizando inducción en fórmulas, que en toda fórmula el número de parentesis izquierdos es igual al número de parentesis derechos.

Solución.

Sea:
$PI(\alpha)$: Número de parentesis izquierdos de α
$PD(\alpha)$: Número de parentesis derechos de α

- Caso base. Sea α una letra proposicional, luego,
  $PI(\alpha) = 0$ y $PD(\alpha) = 0$, así,
  $PI(\alpha) = PD(\alpha)$

- Paso inductivo

  a. Sea α una fbf, tal que $PI(\alpha) = PD(\alpha)$.
  ahora,
  $PI(\neg(\alpha)) = 1 + PI(\alpha)$
  $PD(\neg(\alpha)) = 1 + PD(\alpha)$
  así,

$$\begin{aligned} PI(\neg(\alpha)) &= 1 + PI(\alpha) \\ &= 1 + PD(\alpha) \\ &= PD(\neg(\alpha)) \end{aligned}$$

  b. Sea α y β fbf_s, tal que $PI(\alpha) = PD(\alpha)$ y $PI(\beta) = PD(\beta)$.
  ahora,
  $PI((\alpha)\bigstar(\beta)) = 2 + PI(\alpha) + PI(\beta)$

[1]Recuerde que $\bigstar$ es cualquiera de los operadores binarios vistos anteriormente

$PD((\alpha)\bigstar(\beta)) = 2 + PD(\alpha) + PD(\beta)$
así,

$$\begin{aligned} PI((\alpha)\bigstar(\beta)) &= 2 + PI(\alpha) + PI(\beta) \\ &= 2 + PD(\alpha) + PD(\beta) \\ &= PD((\alpha)\bigstar(\beta)) \end{aligned}$$

2.6 Fórmulas lógicamente equivalentes

Dos fórmulas A y B son lógicamente equivalentes cuando para toda interpretación v, todos los valores de verdad o falsedad son iguales ($v(A) = v(B)$). Para representar que dos fórmulas son lógicamente equivalentes se utiliza el símbolo $\Leftrightarrow$, el cual no es considerado para los autores como un operador booleano.

Por ejemplo, se desea verificar que las fórmulas $A : (\neg p \vee \neg q)$ y $B : (\neg q \vee \neg p)$, son lógicamente equivalentes. Un procedimiento básico para determinar esa equivalencia es elaborar la tabla de verdad para la fórmula A y para la fórmula B. (tabla 2.2)

p	q	$\neg p$	$\neg q$	$\neg p \vee \neg q$	$\neg q \vee \neg p$
V	V	F	F	F	F
V	F	F	V	V	V
F	V	V	F	V	V
F	F	V	V	V	V

Tabla 2.2: Tabla de verdad para la fórmula A y para la fórmula B.

Se observa para cada una de las interpretaciones (línea por línea) de las fórmulas A y B, que estas son idénticas, y por lo tanto se puede afirmar que son lógicamente equivalentes.

Por ejemplo si se tienen las fórmulas: $A : (p \vee q)$ y $B : (q \wedge p)$, y se desea verificar si son lógicamente equivalentes. (tabla 2.3)

p	q	$\neg p$	$\neg q$	$(p \vee q)$	$\Leftrightarrow$	$(q \wedge p)$
V	V	F	F	V		V
V	F	F	V	V		F
F	V	V	F	V		F
F	F	V	V	F		F

Tabla 2.3: Equivalencia entre las fórmulas $A : (p \vee q)$ y $B : (q \wedge p)$.

Del ejemplo se puede afirmar que las fórmulas A y B no son lógicamente equivalentes, debido a que tanto la interpretación de la segunda y tercera línea, tienen un valor de verdad diferente tanto para la fórmula A como para la fórmula B.

Otra forma de determinar si dos fórmulas A y B son lógicamente equivalentes es determinando si el bicondicional $A \leftrightarrow B$ es una tautología. Es decir, $A \Leftrightarrow B$, tendrán los mismos valores de verdad (verdadero o falso) y por lo tanto el valor de verdad de $A \leftrightarrow B$ sea siempre verdadero, es decir, una tautología.

Por ejemplo si se tienen las fórmulas $A : \neg(p \vee q)$ y $B : (\neg p \wedge \neg q)$. Se va a verificar si $A \Leftrightarrow B$, y por lo tanto se analiza si $A \leftrightarrow B$ es una tautología. (tabla 2.4)

p	q	$\neg p$	$\neg q$	$\neg(p \vee q)$	$(\neg p \wedge \neg q)$	$\neg(p \vee q) \leftrightarrow (\neg p \wedge \neg q)$
V	V	F	F	F	F	V
V	F	F	V	F	F	V
F	V	V	F	F	F	V
F	F	V	V	V	F	V

Tabla 2.4: Equivalencia entre las fórmulas $\neg(p \vee q)$ y $(\neg p \wedge \neg q)$.

La fórmula $A \leftrightarrow B$, es una tautología y las fórmulas son lógicamente equivalentes.

Las siguientes son algunas equivalencias lógicas, las cuales son de común uso en el cálculo proposicional:

A_1 **Conmutativa**

- $p \vee q \Leftrightarrow q \vee p$
- $p \wedge q \Leftrightarrow q \wedge p$

A_2 **Identidad**

- $P \vee C \Leftrightarrow p \quad p \vee V \Leftrightarrow V$
- $P \wedge V \Leftrightarrow p \quad p \wedge C \Leftrightarrow C$

A_3 **Distributiva**

- $p \vee (q \wedge r) \Leftrightarrow (p \vee q) \wedge (p \vee r)$
- $p \wedge (q \vee r) \Leftrightarrow (p \wedge q) \vee (p \wedge r)$

A_4 **Complemento**

- $p \vee \neg p \Leftrightarrow V \quad p \wedge \neg p \Leftrightarrow C$
- $\neg(\neg p) \Leftrightarrow p$

A_5 **Idempotencia**

- $p \wedge p \Leftrightarrow p$
- $p \vee p \Leftrightarrow p$

A_6 **Asociativa**

- $(p \wedge q) \wedge r \Leftrightarrow p \wedge (q \wedge r)$
- $(p \vee q) \vee r \Leftrightarrow p \vee (q \vee r)$

A_7 **D'Morgan**

- $\neg(p \vee q) \Leftrightarrow \neg p \wedge \neg q$
- $\neg(p \wedge q) \Leftrightarrow \neg p \vee \neg q$

A_8 **Ley del Condicional**

- $(p \rightarrow q) \Leftrightarrow (\neg p \vee q)$

A_9 **Ley del Bicondicional**

- $(p \leftrightarrow q) \Leftrightarrow (p \rightarrow q) \wedge (q \rightarrow p)$

A_{10} **Absorción**

- $p \wedge (p \vee q) \Leftrightarrow p$
- $p \vee (p \wedge q) \Leftrightarrow p$
- $p \wedge (\neg p \vee q) \Leftrightarrow p \wedge q$
- $p \vee (\neg p \wedge q) \Leftrightarrow p \vee q$

Tomemos por ejemplo la siguientes fórmulas: $\neg((\neg p \vee \neg q) \vee \neg q)$ y $p \wedge q$. Ahora, determinemos is ambas fórmulas son equivalentes:

$$\begin{array}{lcll} \neg((\neg p \vee \neg q) \vee \neg q) & \Leftrightarrow & \neg(\neg p \vee (\neg q \vee \neg q)) & A_6 \\ & \Leftrightarrow & \neg(\neg p \vee \neg q) & A_5 \\ & \Leftrightarrow & \neg(\neg p) \wedge \neg(\neg q) & A_7 \\ & \Leftrightarrow & p \wedge q & A_4 \end{array}$$

Luego $\neg((\neg p \vee \neg q) \vee \neg q) \Leftrightarrow p \wedge q$.

Ahora, tomemos las fórmulas $p \leftrightarrow q$ y $\neg p \leftrightarrow \neg q$; comprobemos que son equivalentes:

$$\begin{array}{lcll} \neg p \leftrightarrow \neg q & \Leftrightarrow & (\neg p \rightarrow \neg q) \wedge (\neg q \rightarrow \neg p) & A_9 \\ & \Leftrightarrow & (\neg\neg p \vee \neg q) \wedge (\neg\neg q \rightarrow \neg p) & A_8 \\ & \Leftrightarrow & (p \vee \neg q) \wedge (q \rightarrow \neg p) & A_4 \\ & \Leftrightarrow & (negq \vee p) \wedge (\neg p \vee q) & A_1 \\ & \Leftrightarrow & (q \rightarrow p) \wedge (p \rightarrow q) & A_8 \\ & \Leftrightarrow & (p \rightarrow q) \wedge (q \rightarrow p) & A_1 \\ & \Leftrightarrow & p \leftrightarrow q & A_9 \end{array}$$

.:Actividad 2.2.

1. Verificar si las siguientes pares de fórmulas (A y B) lógicamente equivalentes:
   - $A : (p \leftrightarrow q) \leftrightarrow s$, $B : s \vee (p \leftrightarrow q)$
   - $A : (p \vee q) \wedge q$, $B : (\neg q \vee q) \leftrightarrow (p \wedge q)$
   - $A : (\neg r \wedge \neg t)$, $B : (t \vee \neg r)$
2. Indicar si las siguientes fórmulas son lógicamente equivalentes
   - $A \leftrightarrow B \Leftrightarrow B \leftrightarrow (A \leftrightarrow B)$
   - $A \leftrightarrow \neg B \Leftrightarrow (A \vee B) \rightarrow (A \wedge B)$
   - $\neg A \vee \neg B \Leftrightarrow \neg A \rightarrow B$
   - $A \leftrightarrow B \Leftrightarrow (A \vee B) \rightarrow (A \wedge B)$
   - $A \wedge \neg B \Leftrightarrow (\neg A \leftrightarrow B) \leftrightarrow (A \vee \neg B)$
   - $A \leftrightarrow B \Leftrightarrow (A \vee B) \wedge (B \rightarrow \neg A)$
   - $A \leftrightarrow B \Leftrightarrow (\neg A \vee B) \wedge (\neg B \rightarrow A)$
   - $\neg A \leftrightarrow B \Leftrightarrow (B \leftrightarrow A) \vee (A \rightarrow B)$
   - $\neg A \rightarrow B \Leftrightarrow A \leftrightarrow (A \wedge B)$

2.7 Satisfacibilidad e insatisfacibilidad

Una fórmula proposicional A es satisfacible si su valor es verdadero en alguna interpretación. Una interpretación satisfecha es llamada un modelo para A (Ben-Ari, 2012). La satisfacibilidad de una fórmula proposicional puede ser determinada mediante diferentes métodos, uno de ellos el de las tablas de verdad, dado que existe un número finito de átomos y un número finito de interpretaciones.

La fórmula $A : p \vee \neg q$, es satisfacible debido a que en varias de sus interpretaciones en la tabla de verdad se evalúan como verdaderas. (tabla 2.5)

p	q	$\neg q$	$p \vee \neg q$
V	V	F	V
V	F	V	V
F	V	F	F
F	F	V	V

Tabla 2.5: Tabla de verdad de $A : p \vee \neg q$.

Una interpretación verdadera se considera un modelo para la fórmula proposicional y por lo tanto la está en correspondencia con la satisfacibilidad. Para la fórmula $A : (p \vee \neg q)$, un modelo puede ser: $v(p) = V$ y $v(\neg q) = F$.

La fórmula $A : (p \wedge q) \vee \neg q$ es satisfacible porque en su tabla de verdad se tienen interpretaciones verdaderas. Un modelo para la fórmula es: $v(p) = F$ y $v(q) = F$, el cual hace verdadero a la expresión: $(p \wedge q) \vee \neg q$. (tabla 2.6)

p	q	$\neg q$	$p \wedge q$	$(p \wedge q) \vee \neg q$
V	V	F	V	V
V	F	V	V	V
F	V	F	F	F
F	F	V	V	V

Tabla 2.6: Tabla de verdad de $A : (p \wedge q) \vee \neg q$.

Una fórmula proposicional es insatisfacible si no es satisfacible, es decir, si su valor es falso en toda interpretación. A una fórmula proposicional con esta propiedad no es posible definir un modelo que la haga satisfacible.

Por ejemplo, la fórmula $A : (q \wedge \neg q)$, es insatisfacible debido a que para cada una de sus interpretaciones estas son falsas, es decir no existe ninguna interpretación verdadera. (tabla 2.7)

q	$\neg q$	$q \wedge \neg q$
V	F	F
F	V	F

Tabla 2.7: Tabla de verdad de $q \wedge \neg q$.

Por ejemplo la fórmula A: $A = (\neg p \vee \neg q) \wedge (p \wedge q)$. (tabla 2.8)

p	q	$(\neg p \vee \neg q)$	$(p \wedge q)$	$(\neg p \vee \neg q) \wedge (p \wedge q)$
V	V	F	V	F
V	F	V	F	F
F	V	V	F	F
F	F	V	F	F

Tabla 2.8: Tabla de verdad de $(\neg p \vee \neg q) \wedge (p \wedge q)$.

Es insatisfacible puesto que cada línea de la tabla de verdad evalúa a faso.

La relación existente entre las fórmulas satisfacible se muestra en la figura, en la cual el rectángulo denota todo el universo de fórmulas proposicionales. Las fórmulas que se encuentran dentro del círculo exterior son las insatisfacibles y aquellas que se encuentran por fuera del círculo son las insatisfacibles. (figura 2.7)

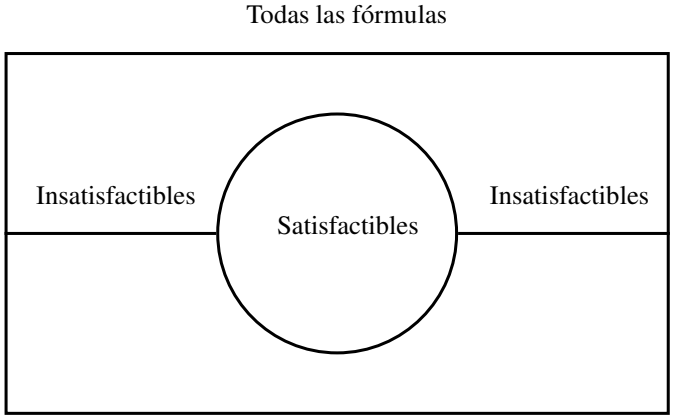


Figura 2.7

Las fórmulas externas al círculo, tienen todas sus interpretaciones falsas y las del área interna al círculo al menos una interpretación verdadera.

2.8 Validez y no validez

Una fórmula proposicional A es válida si su valor es verdadero para toda interpretación. Se dice que una fórmula F es válida (o que F es una tautología), y se representa por F, si todas las interpretaciones son modelos de F (Alonso & Borrero, 2003).

Por ejemplo la fórmula $A : (p \vee \neg p)$ es una fórmula válida dado que cada línea de la tabla de verdad se evalúa como verdadera. (tabla 2.9)

p	$\neg p$	$p \vee \neg p$
V	F	V
F	V	V

Tabla 2.9: Tabla de verdad de $p \vee \neg p$.

Por ejemplo, en la fórmula A: $(p \to q) \to (\neg q \to \neg p)$. Cada una de las líneas de la fórmula son evaluadas como verdaderas, por lo tanto la fórmula es válida. A las fórmulas que cumplen con esta condición se les llama tautologías. (tabla 2.10)

p	q	$(p \to q)$	$(\neg q \to \neg p)$	$(p \to q) \to (\neg q \to \neg p)$
V	V	V	V	V
V	F	F	F	V
F	V	V	V	V
F	F	V	V	V

Tabla 2.10: Tabla de verdad de $(p \to q) \to (\neg q \to \neg p)$.

Una fórmula proposicional es no-válida, si su valor es falso en alguna interpretación.

Por ejemplo la fórmula A: $(p \vee q)$ es una fórmula no válida dado que existe una interpretación en su tabla de verdad que es evaluada como falsa. (tabla 2.11)

p	q	$p \vee q$
V	V	V
V	F	V
F	V	V
F	F	F

Tabla 2.11: Tabla de verdad de $(p \vee q)$.

La fórmula A: $(p \to q) \vee (\neg q \to \neg p)$. Es una fórmula no-válida, pues existe una interpretación que es evaluada como falsa. (tabla 2.12)

p	q	$(p \to q)$	$(\neg q \to \neg p)$	$(p \to q) \vee (\neg q \to \neg p)$
V	V	V	V	V
V	F	F	F	F
F	V	V	V	V
F	F	V	V	V

Tabla 2.12: Tabla de verdad de $(p \to q) \vee (\neg q \to \neg p)$.

Por ejemplo la fórmula A:$(p \wedge q)$ es satisfacible pero no válida, puesto que evalúa tanto un valor verdadero como tres valores falsos. (tabla 2.13)

p	q	$p \wedge q$
V	V	V
V	F	F
F	V	F
F	F	F

Tabla 2.13: Tabla de verdad de $(p \wedge q)$.

Las fórmulas no válidas incluyen fórmulas insatisfacibles así como aquellas que son verdad en alguna interpretación pero no en otras representadas por el anillo externo del círculo (ver figura 2.8) interior.

El área del círculo más grande representa aquellas fórmulas satisfacibles y el área del círculo interno representa aquellas fórmulas satisfacibles no exactamente en una interpretación, son en todas, esto es las fórmulas válidas (Ben-Ari, 2012).

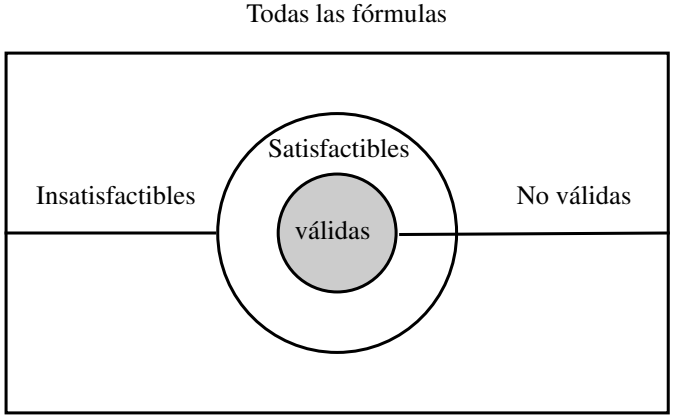


Figura 2.8

Existe una relación directa tanto entre validez e Insatisfacibilidad y entre satisfacibilidad y no validez.

Una fórmula A es válida si y solo si $\neg A$ es insatisfacible. A es satisfacible si y solo si $\neg A$ es no válida. Para decidir si A es válida, aplique el procedimiento de decisión para satisfacibilidad de $\neg A$. Si esto reporta que $\neg A$ es satisfacible, entonces A es no válida y si reporta que $\neg A$ es no satisfacible, entonces A es válida (Ben-Ari, 2012). Tal procedimiento de decisión es llamado usualmente procedimiento de decisión de refutación porque se prueba la validez, refutando la negación.

.:Actividad 2.3.

1. Determine si las siguientes fórmulas proposicionales son: válidas, no válidas, satisfacibles o insatisfacibles. (tabla 2.14)

Fórmula	**Satisfacible**	**Insatisfacible**	**Válida**	**No-válida**
$\neg p \rightarrow q$				
$(p \vee q) \wedge (p \vee q)$				
$\neg(\neg p \vee \neg q)$				
$p \leftrightarrow (q \vee p)$				
$p \wedge \neg p$				
$\neg(\neg p \rightarrow \neg q)$				
$p \leftrightarrow (p \wedge q)$				
$q \vee \neg\neg q$				

Tabla 2.14

2. Dada la siguiente fórmula A: $\neg(\neg p \wedge \neg q) \leftrightarrow (\neg p \vee q)$, determine de que tipo es: válida, satisfacible, insatisfacible o no válida. (tabla 2.15)

   Determine para cada fórmula si es válida, satisfacible, insatisfacible o no válida. Para aquellas en las cuales sea posible, defina un modelo.

   - $A = \neg p \wedge p \rightarrow q \vee q \leftrightarrow p \rightarrow \neg r$

p	q	$\neg(\neg p \wedge \neg q)$	$(\neg p \vee q)$	$\neg(\neg p \wedge \neg q) \leftrightarrow (\neg p \vee q)$
V	V			
V	F			
F	V			
F	F			

Tabla 2.15: Tabla de verdad de $(p \rightarrow q) \vee (\neg q \rightarrow \neg p)$.

- $A = (p \rightarrow \neg q) \rightarrow \neg(\neg q \wedge s)$
- $A = p \wedge \neg((q \rightarrow \neg p) \rightarrow \neg r)$
- $A = (\neg(p \rightarrow q) \vee s) \leftrightarrow q \rightarrow s$
- $A = p \rightarrow q \leftrightarrow \neg p \wedge s$
- $A = ((p \vee q) \rightarrow q \wedge \neg q) \vee p$

2.9 Tableros semánticos

El método de los tableros semánticos es un método de deducción automática que permite representar los sistemas lógicos de forma precisa, basados en un procedimiento de decisión. Se atribuye este método a Beth (Beth, 1955), el cual la define como una búsqueda sistemática de contraejemplos para determinar si una fórmula es consecuencia de otras, disminuyendo la satisfacibilidad de las fórmula proposicional a un subconjunto de literales. En resumen, el método hace una búsqueda sistemática de modelos. La satisfacibilidad para una fórmula proposicional A queda asociada a la satisfacibilidad de un conjunto de literales de la fórmula y por lo tanto, un conjunto de literales es satisfacible si y sólo si no contiene un par complementario de literales.

Un literal es una variable proposicional o la negación de una variable proposicional. Son literales: p, q, r, $\neg q$, $\neg s$, $\neg t$. Un conjunto de literales es satisfacible si y sólo si, no contiene un par complementario de literales.

- Si p es un átomo, el conjunto $\{p, \neg p\}$ es un par complementario de literales.
- Si A es una fórmula, $\{A, \neg A\}$ es un par complementario de fórmulas. A es el complemento de $\neg A$ y $\neg A$ es el complemento de A.

A continuación se muestran fórmulas proposicionales con su respectivo complemento: (tabla 2.16)

Fórmula	**Complemento de la fórmula**
$(p \rightarrow q)$	$\neg(p \rightarrow q)$
$\neg((p \vee q) \vee s)$	$((p \vee q) \vee s)$
$\neg((q \leftrightarrow \neg p) \rightarrow \neg q)$	$(q \leftrightarrow \neg p) \rightarrow \neg q$
$(\neg r \wedge \neg s)$	$\neg(\neg r \wedge \neg s)$
$(((p \vee q)\neg s) \leftrightarrow (\neg p \wedge s))$	$\neg(((p \vee q) \vee s) \leftrightarrow (\neg p \wedge s))$

Tabla 2.16: Fórmulas proposicionales con su respectivo complemento.

En la tabla 2.17 se muestran una serie de casos en las cuales se define cada fórmula proposicional un conjunto de literales y la propiedad de la fórmula.
Considere la fórmula $A : q \wedge (p \vee \neg q)$, A es verdadera si $v(q) = V$ y $v(p \vee \neg q) = V$. Entonces $v(A) = V$, si:

Fórmula	**Conjunto de literales**	**Par complementario**	**Modelo**	**Tipo de fórmula**
$A : (p \vee \neg q)$	$\{p, \neg q\}$	No	$v(p) = V$, $v(q) = F$	Satisfacible
$A : (p \wedge q) \rightarrow \neg r$	$\{p, q, \neg r\}$	No	$v(p) = V$, $v(q) = F$, $v(\neg r) = V$	Satisfacible
$A : p \wedge \neg p$	$\{p, \neg p\}$	Si	No tiene, $v(\neg r) = V$	insatisfacible

Tabla 2.17: Fórmulas proposicional, un conjunto de literales y la propiedad de la fórmula.

1. $v(q) = V$ y $v(p) = V$ o
2. $v(q) = V$ y $v(\neg q) = V$

Para el conjunto de literales $\{q, p\}$, no es un par complementario y se puede establecer un modelo para la fórmula A, $v(q) = V$ y $v(p) = V$, y por lo tanto la fórmula es satisfacible. El conjunto de literales $\{q, \neg q\}$ es un par complementario, por lo tanto no es satisfacible y no se puede definir un modelo.

2.9.1 Árbol semántico

Un árbol semántico es una sucesión de sucesiones de fórmulas llamadas ramas, generadas a partir de un conjunto (no vacío) de fórmulas, por aplicación a éstas de las reglas (y a las fórmulas resultantes que sean complejas) (Nepomuceno, 2002). Un árbol semántico se puede representar a partir de la raíz de un árbol, en donde cada una de las hojas que se generen como hijas, son producto de una búsqueda sistemática de modelos.

Para realizar la búsqueda sistemática de modelos, existen una serie de reglas que deben ser aplicadas de acuerdo al conector u operador booleano de la fórmula proposicional. Estas dos reglas son (Ben-Ari, 2012):

1. α (alfa) - son conjuntivas y estas satisfacen solo si ambas subfórmulas α_1 y α_2 son satisfacibles.
2. β (beta) - son disyuntivas y estas satisfacen aun usando solo una de las dos subfórmulas β_1 o β_2 es satisfacible.

Las reglas se aplican de acuerdo al conectivo principal. Para las α-fórmulas, se deben satisfacer sus dos componentes α_1 y α_2. (tabla 2.18)

α	α_1	α_2
$\neg\neg A$	A	
$A_1 \wedge A_2$	A_1	A_2
$\neg(A_1 \vee A_2)$	$\neg A_1$	$\neg A_2$
$\neg(A_1 \rightarrow A_2)$	A_1	$\neg A_2$
$A_1 \leftrightarrow A_2$	$A_1 \rightarrow A_2$	$A_2 \rightarrow A_1$

Tabla 2.18: α-Fórmulas

Para las β-fórmulas es necesario que al menos uno de sus componentes β_1 y β_2, se satisfaga. (tabla 2.19)

β	β_1	β_2
$B_1 \vee B_2$	B_1	B_2
$\neg(B_1 \wedge B_2)$	$\neg B_1$	$\neg B_2$
$B_1 \rightarrow B_2$	$\neg B_1$	B_2
$\neg(B_1 \leftrightarrow B_2)$	$\neg(B_1 \rightarrow B_2)$	$\neg(B_2 \rightarrow B_1)$

Tabla 2.19: β-Fórmulas

2.9.2 Construcción del tablero semántico

Un tablero semántico se considera completo cuando las hojas que este son etiquetadas. Una hoja que contiene un conjunto complementario de literales se marcará con una X y por lo tanto será insatisfacible, es decir no tiene un modelo. Una hoja que no contenga un conjunto complementario de literales será marcada con una O y será por lo tanto satisfacible, es decir tiene un modelo. El procedimiento del método finaliza cuando todas las fórmulas se encuentran debidamente marcadas. Cando sean etiquetadas todas las hojas del árbol, se llamará un tablero semántico. Este método también es usado para hacer demostraciones por refutación, es decir, se puede determinar la validez negando la formula proposicional.

Por ejemplo, para la fórmula $A : p \wedge \neg p$. Esta tiene la estructura $A_1 \wedge A_2$, por lo cual se aplica una α-fórmula que se deriva en α_1 y α_2. El árbol semántico contiene dos pares complementarios $\{p, \neg p\}$ y se puede etiquetar con una X. Por lo tanto la fórmula proposicional es insatisfacible. El árbol semántico resultante se muestra en la figura 2.9:

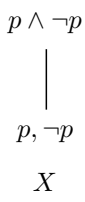


Figura 2.9: Árbol semántico de $A : p \wedge \neg p$

Por ejemplo un tablero para la fórmula $A : (p \vee \neg q) \wedge q$. Esta fórmula cuenta con la estructura $A_1 \wedge A_2$, por lo cual se aplica una α-fórmula que se deriva en α_1 y α_2. En donde quedan dos subfórmulas $(p \vee \neg q)$ y q. En este caso el conector principal es el $\vee$, la cual responde a la estructura $B_1 \vee B_2$, y por lo tanto se aplica una β-fórmula que se deriva en β_1 y β_2. Una vez aplicada surgen los conjuntos $\{q, p\}$ y $\{q, \neg q\}$. El conjunto $\{q, p\}$ es satisfacible y se etiqueta con O. El conjunto $\{p, \neg q\}$ es un par complementario de literales y se etiqueta con X. La fórmula A es satisfacible puesto que el conjunto $\{q, p\}$ es abierto y se puede encontrar por lo tanto un modelo: es $v(p) = V$ y $v(q) = V$. El tablero semántico se muestra en la figura 2.10.

Por ejemplo, para la fórmula $A : (p \vee q) \wedge (\neg p \wedge \neg q)$, se identifica que la fórmula cuenta con la estructura $A_1 \wedge A_2$, por lo cual se aplica una α-fórmula que se deriva en α_1 y α_2. Se elimina del árbol el operador $\wedge$ y se separan las subfórmulas con una coma (,), queda el árbol con las subexpresiones: $p \vee q, \neg p \wedge \neg q$. Teniendo en cuenta que en este punto cada subexpresión tiene un conectivo principal, se puede entonces aplicar ya sea una α fórmula o una β fórmula. Se aplica la α fórmula, y se genera un nuevo nodo quedando el árbol con: $p \vee q$, $\neg p$, $\neg q$. Finalmente se aplica una β fórmula, teniendo en cuenta que solo queda una disyunción. En este caso, se generan dos subexpresiones β_1 y β_2. Los dos hojas del árbol quedan etiquetadas con X, por lo tanto es cerrado y la fórmula proposicional es insatisfacible. No es posible encontrar un modelo para la fórmula. El árbol semántico resultante se

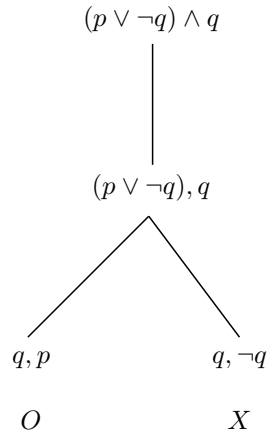


Figura 2.10: Árbol semántico de $A : (p \vee \neg q) \wedge q$

muestra en la figura 2.11:

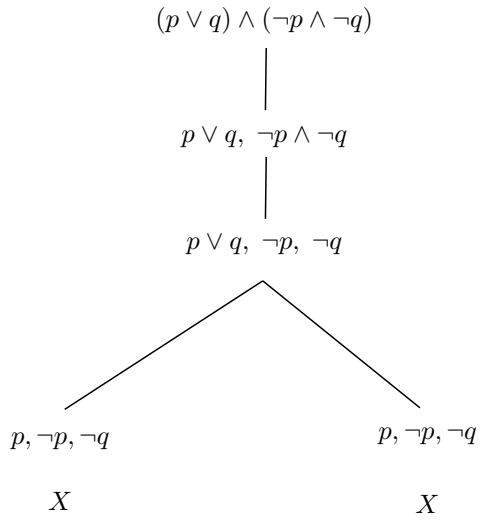


Figura 2.11: Árbol semántico de $A : (p \vee q) \wedge (\neg p \wedge \neg q)$

Por ejemplo, para la fórmula $A : (p \rightarrow ((q \vee r) \rightarrow (p \wedge r)))$, corresponde con la estructura $B_1 \rightarrow B_2$, por lo tanto se aplica una β-fórmula que se deriva en β_1 y β_2. La subfórmula β_1 queda con un literal $\neg p$, el cual se marca como abierto. A continuación se analiza la subfórmula β_2 $(q \vee r) \rightarrow (p \wedge r)$, a la cual se le aplica una β-fórmula que genera dos fórmulas β_1 y β_2. La subfórmula β_1 queda como: $\neg(q \vee r)$ y la subfórmula β_2 queda como: $(p \wedge r)$. Tanto a β_1 como a β_2, se les aplica una α fórmula. El árbol en este punto se etiqueta ambas hojas como abiertas, teniendo en cuenta que no tienen pares complementarios de literales. Como al menos una de las hojas del árbol es abierta, entonces la fórmula A es satisfacible y se puede obtener un modelo. El árbol semántico resultante se muestra en la figura 2.12:

Generalizando, se puede afirmar que una fórmula proposicional es satisfacible si y solo si el tablero

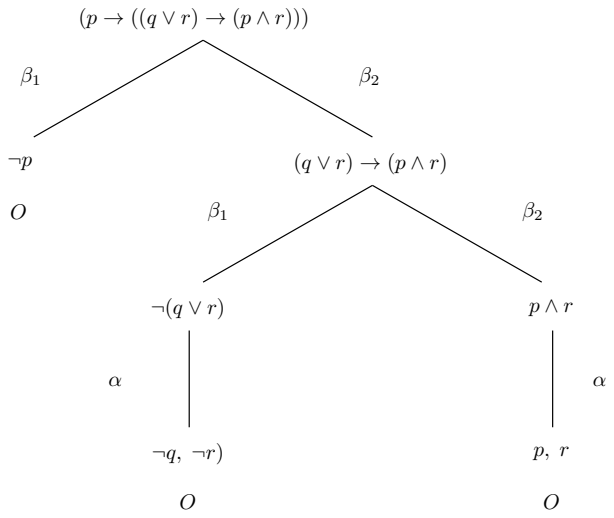


Figura 2.12: Árbol semántico de $A : (p \to ((q \vee r) \to (p \wedge r)))$

semántico es abierto. Un tablero semántico es abierto cuando al menos una de sus hojas es marcada como abierta. Una fórmula proposicional es insatisfacible si y solo si el tablero semántico es cerrado. Un tablero semántico es cerrado cuando todas sus hojas son marcadas como cerradas.

Los tableros semánticos pueden ser usados como procedimientos de decisión para verificar la validez de una fórmula proposicional. Para probar si una fórmula A es válida, entonces se refuta negando a A. Una fórmula A es válida si y solo si $\neg A$ es insatisfacible si y solo si el tablero semántico para $\neg A$ es cerrado.

Por ejemplo, se desea verificar si la fórmula $A : (p \vee \neg p)$ es válida. Inicialmente se niega la fórmula: $\neg(p \vee \neg p)$. En este caso se aplica una α-fórmula. La cual genera una única hoja que se etiqueta con X, pues se tiene el par complementario de literales. $\{p, \neg p\}$. (figura 2.13)

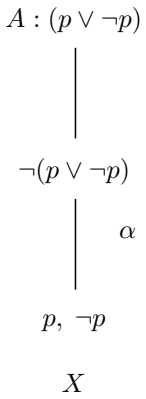


Figura 2.13: Árbol semántico de $\neg(p \vee \neg p)$

Para la fórmula A, se prueba la validez refutando la fórmula original, como $\neg A$ es cerrado, entonces $\neg A$ es insatisfacible y por lo tanto A es válido. Mediante una tabla de verdad se puede verificar que no es posible definir un modelo para la fórmula proposicional. (tabla 2.20)

p	$\neg p$	$(p \vee \neg p)$	$\neg(p \vee \neg p)$
V	F	V	F
F	V	V	F

Tabla 2.20: Tabla de verdad de $\neg(p \vee \neg p)$

Por ejemplo, se desea verificar la fórmula $A : ((p \vee q) \rightarrow (\neg p \vee \neg q))$, es una fórmula válida. El operador principal es el $\rightarrow$, por lo que la estructura es $\neg(A_1 \rightarrow A_2)$, por ello tanto se aplica una α-fórmula que se deriva en α_1 y α_2. La subfórmula α_1 queda como $(p \vee q)$, y la subfórmula α_2 queda como $\neg(\neg p \vee \neg q)$. Se aplica una α-fórmula a $\neg(\neg p \vee \neg q)$,quedando conformado el árbol con las subfórmulas: $(p \vee q)$, p, q. Se aplica una β-fórmula la cual genera dos nodos hoja. Los conjuntos de literales son: $\{p, p, q\}$ y $\{q, p, q\}$. No se tiene un par complementario de literales y por lo tanto las hojas se etiquetan como abiertas. La forma en que se etiqueto el árbol muestra que $\neg A$ es satisfacible, por lo tanto A es no válida. El árbol semántico resultante se en la figura 2.14:

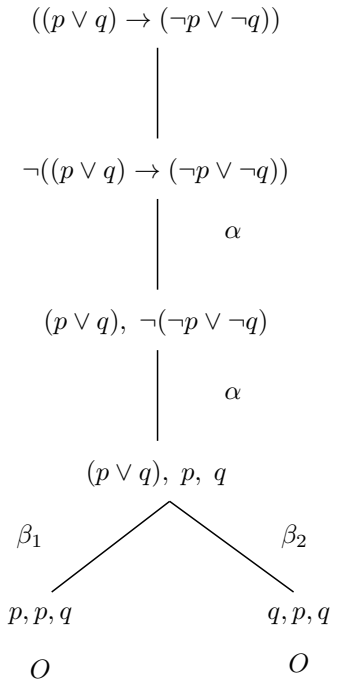


Figura 2.14: Árbol semántico de $A : ((p \vee q) \rightarrow (\neg p \vee \neg q))$

Se desea saber si al fórmula $A = (\neg s \rightarrow \neg q) \rightarrow (q \rightarrow s)$, es válida. Inicialmente para probar validez se niega fórmula. El operador principal es el $\rightarrow$, por lo que la estructura es $\neg(A_1 \rightarrow A_2)$, por ello tanto se aplica una α-fórmula que se deriva en α_1 y α_2. La subfórmula α_1 queda como $(\neg s \rightarrow \neg q)$, y la subfórmula α_2 queda como $\neg(q \rightarrow s)$. Se aplica una β-fórmula la cual genera dos nodos hoja y finalmente se aplica a cada nodo una α-fórmula. Los conjuntos tienen pares complementarios, por lo tanto las hojas son cerradas y A es un fórmula válida. El árbol semántico resultante se muestra en la figura 2.15:

Por ejemplo si se tiene la fórmula proposicional: $p \wedge (p \rightarrow (q \rightarrow \neg p))$, y se aplican las reglas, es posible

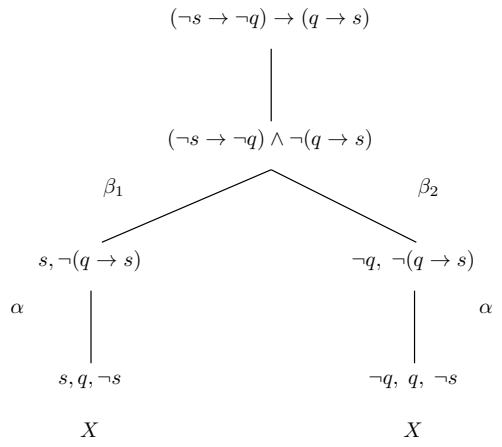


Figura 2.15: Árbol semántico de $A = (\neg s \to \neg q) \to (q \to s)$

la siguiente representación del árbol. El árbol etiquetado, tiene hojas abiertas y cerradas. La fórmula proposicional $A : p \wedge (p \to (q \to \neg p))$, es satisfacible y el tablero semántico es abierto. (figura 2.16)

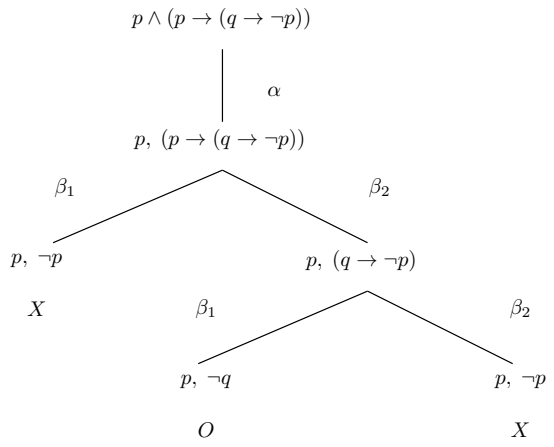


Figura 2.16: Árbol semántico de $p \wedge (p \to (q \to \neg p))$

Por ejemplo, la fórmula proposicional $A : p \wedge \neg q \wedge (p \to ((q \vee r) \to (p \wedge r))$. Su árbol etiquetado, tiene dos hojas abiertas y una hoja cerrada, entonces la fórmula $A : p \wedge \neg q \wedge (p \to ((q \vee r) \to (p \wedge r)))$, es satisfacible y se puede definir un modelo en las hojas abiertas. El árbol semántico resultante se muestra en la figura 2.17:

El método de los tableros semánticos también puede ser extendido a fórmulas. Por ejemplo, si se tiene la fórmula $A : ((p \vee q) \wedge \neg(p \vee q))$, se aplica una α-fórmula y se observa que se existe un tiene un par complementario de fórmulas. Por lo tanto se puede afirmar directamente que el tablero semántico es cerrado y la fórmula es insatisfacible. (figura 2.18)

.:Actividad 2.4.

1. Determine si los siguientes conjuntos de fórmulas son satisfacibles;

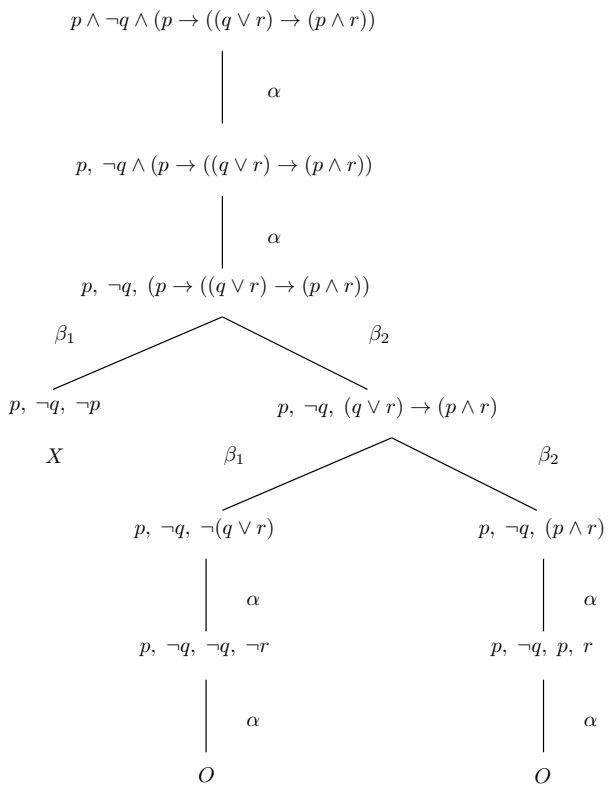


Figura 2.17: Árbol semántico de $A : p \wedge \neg q \wedge (p \to ((q \vee r) \to (p \wedge r))$

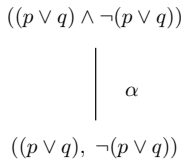


Figura 2.18: Árbol semántico de $A : ((p \vee q) \wedge \neg(p \vee q))$

- $\{\neg p, p, q\}$
- $\{q, p, \neg q\}$
- $\{q, r, p, \neg s\}$

2. Dadas las siguientes fórmulas A, determine por el método de los tableros semánticos si son fórmulas: satisfacibles o insatisfacibles, y cuando sea posible determine un modelo.

   - $A = (p \to q) \wedge (\neg p \wedge q)$
   - $A = (q \wedge \neg p) \to q)$
   - $A = ((p \vee q)) \leftrightarrow (\neg p \vee q)$
   - $A = (p \to q) \leftrightarrow (\neg p \to \neg q)$
   - $A = \neg((p \vee q) \to q) \wedge \neg((\neg q \vee p) \wedge \neg p)$
   - $A = (q \to q) \vee \neg(q \vee p)$

3. Dadas las fórmulas proposicionales, determine por el método de los tableros semánticos si son válidas o no válidas.

   - $A = \neg(p \leftrightarrow q) \wedge (p \wedge q)$
   - $A = \neg((q \rightarrow \neg p) \rightarrow \neg q)$
   - $A = \neg(p \vee q) \leftrightarrow (\neg p \leftrightarrow s)$
   - $A = (\neg p \rightarrow \neg q) \rightarrow (\neg q \rightarrow \neg p)$
   - $A = (p \rightarrow q) \rightarrow (\neg q \vee p)$
   - $A = (q \rightarrow q) \leftrightarrow \neg q \vee p)$

2.10 Resolución proposicional

El método de resolución proposicional fue desarrollado por Robinson (Robinson, 1965). Es un método de demostración automática fundamentado también en la búsqueda. El método se fundamenta en una regla de inferencia denominada regla de resolución. Tiene como elemento particular que trabaja con cláusulas y no directamente con fórmulas proposicionales.

A diferencia de otros métodos de deducción como los tableros semánticos, en donde se manejan cantidad de reglas para inferir el resultado, el método de resolución solo utiliza una regla para deducir. Así mismo, es un método que utiliza exclusivamente clausulas.

2.10.1 Formas normal conjuntiva y disyuntiva

Una fórmula proposicional está en forma normal cuando únicamente está constituida de disyunciones o conjunciones, y la negación solo afecta a los literales, es decir en ella no existen operadores booleanos condicionales, bicondicionales, ni negación de la fórmula. Una forma normal puede estar en forma normal disyuntiva o forma normal conjuntiva.

Una fórmula está en Forma Normal Disyuntiva (FND) si está constituida por una disyunción de conjunciones, es decir el operador principal es la disyunción y las conjunciones son las dependientes. Donde $C_1 \vee C_2 \vee C_3 \vee C_4 \vee \ldots C_n$, y cada C_i una clausula y $B_{i1} \vee B_{i2} \vee B_{i3} \vee \ldots B_{ini}$, es un literal de la cláusula. Se representa formalmente como.

$$\bigvee_{i=1}^{m} \left(\bigwedge_{j=1}^{n_i} l_{ij} \right)$$

Son ejemplos de FND las siguientes expresiones:

- $p \vee q$
- $(\neg p \wedge \neg q \wedge r) \vee (p \wedge s)$
- $(q \wedge p) \vee (\neg p) \vee (q)$
- $\neg p$

Las siguientes fórmulas no están en FND:

- $\neg(p \vee s)$ (el operador de negación afecta la fórmula)
- $((\neg p \vee q \vee s) \wedge r)) \vee \neg s$ (tiene una disyunción)

Una fórmula está en Forma Normal Conjuntiva (FNC) si está constituida por una conjunción de disyunciones, es decir el operador principal es la conjunción y las disyunciones son las dependientes. Donde $C_1 \wedge C_2 \wedge C_3 \wedge C4 \wedge \ldots C_n$, y cada C_i una clausula y $B_{i1} \wedge B_{i2} \wedge B_{i3} \wedge \ldots B_{ini}$, es un literal de la cláusula. Se representa formalmente como:

$$\bigwedge_{i=1}^{m} \left(\bigvee_{j=1}^{n_i} l_{ij} \right)$$

Son ejemplos de FNC las siguientes expresiones:

- $\neg p \wedge s$
- $(p \vee q) \wedge (p \vee \neg s)$
- $(\neg p \vee \neg q \vee \neg s) \wedge (p \vee s) \wedge (\neg p)$
- q

Las siguientes fórmulas no están en forma normal conjuntiva FCN

- $\neg(p \wedge s)$ (el operador de negacion afecta la fórmula)
- $((\neg p \wedge q) \vee \neg r)) \wedge (\neg p)$ (tiene una disyunción en la fórmula).

Una fórmula en FND será insatisfacible si todas sus disyunciones son falsas, de lo contrario, se debe establecer si es una tautología o una fórmula satisfacible. Una fórmula en FNC será una tautología si todas sus conjunciones son verdaderas, de lo contrario, se debe establecer si la fórmula es satisfacible o insatisfacible.

Toda fórmula proposicional tiene su equivalente en FNC. Para convertir una fórmula a FNC, es posible establecer una serie de pasos, los cuales garantizan la propiedad de equivalencia lógica.

1. Eliminar todos los conectivos diferentes a: negación, conjunción y disyunción. (se eliminan los operadores $\leftrightarrow$ y $\rightarrow$).
2. Aplicar las leyes de De Morgan, para llevar todas las apariciones de $\neg$ hasta los átomos.
   - $\neg(A \wedge B) \leftrightarrow (\neg A \vee \neg B)$
   - $\neg(A \vee B) \leftrightarrow (\neg A \wedge \neg B)$
3. Eliminar las dobles negaciones.
4. Aplicar de forma repetida las leyes de Distributividad, para llevar todas las apariciones de $\vee$ hasta los átomos.
   - $A \vee (B \wedge C) \leftrightarrow (A \vee B) \wedge (A \vee C)$
   - $(A \wedge B) \vee C \leftrightarrow (A \vee C) \wedge (B \vee C)$

En los siguientes ejemplos se muestra la aplicación de los pasos para transformar a FNC. Inicialmente se desea transformar la fórmula: $((p \rightarrow q) \wedge p) \rightarrow q$
El siguiente ejemplo muestra cómo se trasforma la fórmula: $(p \rightarrow q) \rightarrow (\neg p \rightarrow \neg q)$, a su equivalente en FNC.

.:Actividad 2.5.

1. Dadas las siguientes fórmulas, transformar a su equivalente en forma normal conjuntiva.

Pasos	Fórmula
1. Se eliminan todas la apariciones de $\rightarrow$	$\neg((\neg p \vee q) \wedge p) \vee q$
2. Llevar todas las apariciones de $\neg$ hasta los átomos y así disminuir su alcance	$(p \wedge \neg q) \vee \neg p \vee q$
3. Aplicar de forma repetida las leyes de Distributividad	$(p \vee \neg p \vee q) \wedge (\neg q \vee \neg p \vee q)$

Tabla 2.21: FNC. de $((p \rightarrow q) \wedge p) \rightarrow q$

Pasos	Fórmula
1. Se eliminan todas la apariciones de $\rightarrow$	$\neg(\neg p \vee q) \vee (\neg\neg p \vee \neg q)$
2. Llevar todas las apariciones de $\neg$ hasta los átomos y así disminuir su alcance	$(\neg\neg p \wedge \neg q) \vee (\neg\neg p \vee \neg q))$
3. Eliminar las dobles negaciones.	$(p \wedge \neg q) \vee (p \vee \neg q))$
4. Aplicar de forma repetida las leyes de Distributividad	$(p \vee p \vee \neg q) \wedge (\neg q \vee p \vee \neg q)$

Tabla 2.22: FNC. de $(p \rightarrow q) \rightarrow (\neg p \rightarrow \neg q)$

- $(p \rightarrow q) \vee \neg(\neg p \wedge q)$
- $\neg((q \rightarrow \neg p) \rightarrow \neg q)$
- $\neg((p \vee q) \leftrightarrow (\neg p \vee q)$
- $\neg(p \rightarrow q) \vee (\neg p \rightarrow \neg q)$
- $((p \vee q) \rightarrow q) \rightarrow \neg p \rightarrow q$

2. Dadas las siguientes fórmulas, transformar a su equivalente en forma normal conjuntiva.

- $(p \rightarrow q) \vee \neg q$
- $(q \vee \neg p \wedge s) \leftrightarrow (\neg q \wedge \neg s)$
- $(p \vee q) \rightarrow (\neg p \vee q)$
- $\neg(p \vee q) \rightarrow (\neg p \vee \neg q)$
- $\neg((p \vee q) \vee q) \leftrightarrow (\neg p \rightarrow q)$

2.10.2 Literales, cláusulas y Forma clausal

Un literal es una proposición atómica p su negación $\neg p$, es decir una variable proposicional o la negación de una variable proposicional. Todo literal tiene un complemento. Si s es un literal, $\neg s$ es su complemento. Si $\neg s$ es un literal, entonces s es su complemento. Los literales se constituyen las clausulas.

Una cláusula se compone de literales que se encuentran unidos por medio del operador de disyunción, esta disyunción se puede dar entre cero o más literales. Una clausula es una disyunción de literales. Una cláusula que no contenga literales se denomina vacía, y su representación es mediante el símbolo $\circledast$. Una clausula vacía es insatisfacible.

Una fórmula está en Forma Clausal si está conformada exclusivamente por un conjunto de cláusulas. La transformación de una fórmula en FNC a Forma Clausal es inmediata sustituyendo las conectivas $\wedge$

Fórmula proposicional	Conjunto de clausulas
$(\neg p \vee q \vee \neg q)$	$\{\neg pq\neg q\}$
$(p \vee \neg p \vee q \vee p) \wedge (\neg q \vee p) \wedge \neg s$	$\{p\neg pq, \neg qp, \neg s, \neg p\}$
$(p \vee q \vee \neg q) \wedge (\neg q \vee p \vee q)$	$\{pq\neg q, \neg qpq\}$
$(p \vee q) \vee (p \rightarrow r) \vee \neg r$	$\{pq, \neg pr, \neg r\}$

Tabla 2.23

por comas y englobando las disyunciones entre llaves (Blanco, Smith, & Damián, 2001). A continuación se presentan unas fórmulas proposicionales con su equivalente expresión en términos de cláusulas. Algunos aspectos a destacar en los conjuntos de cláusulas son:

- Existen clausulas con un solo literal, a las cuales se les conoce como cláusulas unitarias.
- Cuando un literal se repite en la cláusula es posible aplicarla equivalencia lógica $(p \vee p) \Leftrightarrow p$. En este caso la cláusula no pierde sus propiedades.
- Cuando una clausula no contiene ningún literal, se denomina clausula vacía y como se ha mencionado, es insatisfacible y no puede ser satisfecha.
- Una cláusula que tiene un literal verdadero, se denomina cláusula de Horn.

2.10.3 Método de resolución

El método de resolución es un procedimiento automático es un método que trabaja únicamente con cláusulas que estén en FNC. En este método se aplica una única regla de inferencia, denominada regla de resolución.

La regla de resolución se aplica cuando existan literales complementarios entre pares de cláusulas. Cuando en dos cláusulas C_1 y C_2, se contenga en cada una literales tales que $l \in C_1$ y $\neg l \in C_2$, se afirma que son clausulas resolubles. El resolvente entre dos cláusulas C_1 y C_2, da lugar a una nueva cláusula C. El resolvente entre C_1 y C_2, da lugar a una nueva cláusula respecto al literal l. A este proceso se le denomina paso de resolución. La aplicación del resolvente se denota de la siguiente manera:

$$R_l(C_1, C_2) = (C_1 - \{l\}) \cup (C_2 - \{\neg l\})$$

Por ejemplo, si se tienen las cláusulas: $C_1 = pq\neg r$ y $C_2 = qr\neg t$, el literal c, está en la cláusula $C_1 = \neg r$ y en la cláusula $C_2 = r$. El resolvente entre C_1 y C_2, se obtienen de la siguiente manera:

$$R(C_1, C_2) = (pq\neg r - \{\neg r\}) \cup (qr\neg t - \{r\}) = pq \cup q\neg t = pq\neg t$$

Los siguientes son ejemplos en los cuales se aplica el resolvente a dos clausulas:
El resolvente de dos cláusulas es satisfacible cuando los pares de cláusulas C1 y C2, son mutuamente satisfacibles. Un conjunto de cláusulas es satisfacible si y sólo si existe una misma interpretación que satisface a cada cláusula del conjunto. Cuando el resolvente es $C = \square$ el conjunto de cláusulas es insatisfacible.

Son diversos autores (Ben-Ari, 2012), (Labra & Fernández, 1998), que presentan una serie de estrategias o casos para simplificar clausulas, las cuales están orientadas a eliminar clausulas antes de ser usadas en el resolvente. Con ello se busca un ahorro en la búsqueda de la solución.

Clausulas	Resolvente
$C_1 = pq$, $C_2 = \neg q \neg r$	$p \neg r$
$C_1 = r$, $C_2 = \neg r$	$\square$ (clausula vacía)
$C_1 = \neg p \neg q$, $C_2 = \neg q \neg r$	No se puede aplicar resolvente entre C_1 y C_2, (No existen literales complementarios)
$C_1 = p \neg q r$, $C_2 = q r \neg t$	$pr \neg t$

Tabla 2.24

Caso 1 : Si en todo el conjunto de cláusulas, existe una cláusula que no contiene un literal complementario a él en todo el conjunto, es un literal al cual no se le podrá aplicar el resolvente. En este caso esta cláusula puede ser eliminada del conjunto.

Caso 2 : Si existe una cláusula con un literal repetido, esto es una tautología y por lo tanto a esa cláusula se le puede eliminar uno de esos literales.

Caso 3 : Si una clausula contiene a otra clausula, entonces es posible eliminar del conjunto de cláusulas, la cláusula que está contenida en el conjunto.

A continuación se ejemplifica la aplicación de estos casos:

Conjunto	Estrategia de borrado
$\{\neg pq \neg q,\ ppq,\ pqr,\ st\}$	$\{\neg pq \neg q,\ pq,\ p \neg q \neg r,\ st\}$ (Caso 2, se elimina el literal p de la segunda cláusula
$\{p \neg pq, \neg qp, \neg s, \neg p, t\}$	$\{p \neg pq, \neg qp, \neg s, \neg p, t\}$ (caso 1, se elimina el literal t del conjunto)
$\{pq \neg q, \neg qpr, pq\}$	$\{pq \neg q, \neg qpr\}$ (caso 3, se elimina la cláusula pq)

Tabla 2.25

Con fundamento en todo lo expuesto anteriormente, a continuación se presenta el algoritmo de resolución proposicional (Blanco et al., 2001).

Entrada: Conjunto de Cláusulas C
Salida: Determina si el conjunto de cláusulas es insatisfacible

1. Buscar en C dos cláusulas C_1 y C_2, en la cual exista un literal l que cumpla que $l \in C_1$ y $\neg l \in C_2$.
2. Si se encuentran:
   2.1 Calcular el resolvente entre las dos cláusulas C_1, C_2 y añadirlo al conjunto C.
   2.1 Si $R(C_1, C_2) = \square$, entonces terminar e indicar que es insatisfacible.
   2.2 De lo contrario, volver al paso 1
3. Si no se encuentran: Terminar indicando que no es insatisfacible.

Por ejemplo, se desea aplicar el algoritmo de resolución proposicional para la fórmula: $(p \wedge \neg p)$. Inicialmente el cada una de las cláusulas será numeradas para aplicar el procedimiento de resolución. Cuando sea aplicado el resolvente se indicará sobre que cláusulas se está trabajando. El conjunto de cláusulas es:

1. p
2. $\neg p$
3. $R(1, 2)$

Teniendo en cuenta que el resolvente de entre las cláusulas 1 y 2, llegó a $\square$ se puede afirmar que la fórmula proposicional es insatisfacible. Esto se puede verificar desde la definición misma de interpretación proposicional, pues si $v(p) = V$, entonces $v(\neg p) = F$, y por lo tanto no es posible encontrar un modelo para la fórmula $(p \wedge \neg p)$.

Los siguientes son ejemplos de aplicación del método de resolución proposicional.

Considere el siguiente conjunto de cláusulas: $\{(1)p, (2)\neg q, (3)\neg pq\}$

1. p
2. $\neg q$
3. $\neg pq$
4. $q \quad R(1, 3)$
5. $\square \quad R(2, 4)$

El conjunto de cláusulas es insatisfacible. Este mismo conjunto de cláusulas puede tener una serie de pasos a los que se llega a la misma deducción de insatisfacibilidad.

1. p
2. $\neg q$
3. $\neg pq$
4. $\neg p \quad R(2, 3)$

5. $\square \quad R(1,4)$

Dado el siguiente conjunto de cláusulas, determine si es insatisfacible o no: $\{(1)pq, (2)\neg pq, (3)p\neg q, (4)\neg p\neg q\}$.

1. pq
2. $\neg pq$
3. $p\neg q$
4. $\neg p\neg q$
5. $q \quad R(1,2)$
6. $\neg q \quad R(3,4)$
7. $\square \quad R(5,6)$

El conjunto es insatisfacible.

Retomando el ejemplo de la sección 2.8 se puede también utilizar el método de resolución para verificar la validez de una fórmula mediante el procedimiento de refutación. Esto se verifica negando la conclusión obtenida de los argumentos.

La Universidad Autónoma es privada, o la Universidad Nacional lo es. Si la Universidad Autónoma es privada, entonces es sin ánimo de lucro. La Universidad Católica con ánimo de lucro tiene acreditación de calidad. Por lo tanto, la Universidad Nacional es privada.

Utilicemos las siguientes proposiciones atómicas:

p Universidad Autónoma es privada. q Universidad del Nacional es privada. r Universidad Autónoma es sin ánimo de lucro.
La representación de los argumentos será:

$p \wedge q$
$p \rightarrow r$
$\neg r$

Para este caso se va a negar la conclusión para verificar la validez por medio de la refutación de ella. El conjunto de cláusulas quedarían: $\{pq, \neg pr, \neg r, \neg q\}$. Donde $\neg q$ es la negación de la conclusión.

1. pq
2. $\neg pr$
3. $\neg r$
4. $\neg q$
5. $\neg p \quad R(2,3)$
6. $p \quad R(1,4)$
7. $\square \quad R(5,6)$

Con base en esta deducción, se puede afirmar que la argumentación obtenida es valida por el procedimiento de refutación.

.:Actividad 2.6.

1. Dadas las siguientes fórmulas, transformar a su equivalente en forma normal conjuntiva y aplicar cuando sea posible la regla de resolución repetidamente.

   - $(p \to \neg q) \to (\neg p \land \neg q)$
   - $\neg((q \to \neg p) \to \neg q) \lor (\neg p \to \neg q)$
   - $((p \lor q) \leftrightarrow (\neg p \lor \neg q)$
   - $(p \to q) \leftrightarrow (\neg p \to \neg q)$
   - $\neg(((p \lor q) \to q) \to p) \to q$

2. Obtener el resolvente de los siguientes pares de cláusulas cuando sea posible.

   - $C_1 = p\neg q\neg r$ y $C_2 = qr\neg p$
   - $C_1 = \neg r$ y $C_2 = qrp$
   - $C_1 = p\neg q\neg r$ y $C_2 = \neg pqr$
   - $C_1 = \neg pq$ y $C_2 = \neg q\neg p$
   - $C_1 = pq\neg r$ y $C_2 = qr\neg t$

3. Dados los siguientes conjuntos de cláusulas, terminar si este es satisfacible o insatisfacible.

   - $\{pqr, \neg p\neg qr, \neg r, \neg q\}$
   - $\{\neg p\neg q, p\neg qr, r, q, s, \neg ps\}$
   - $\{r, \neg p\neg qr, \neg pq, \neg qs\}$
   - $\{pq, qr, \neg r, q, p\}$

4. Determinar por el método de resolución si las siguientes fórmulas son insatisfacibles.

   - $((p \to \neg q) \to \neg p) \to q$
   - $((q \lor \neg p) \to q) \land (\neg p \to \neg q)$
   - $((p \lor q) \to (\neg p \lor \neg q \lor r)) \to \neg r$
   - $(p \to \neg q) \to (\neg p \to \neg q)$
   - $(((p \lor q \lor r) \to \neg q) \to p) \to \neg q$

2.11 Pruebas deductivas

En las temáticas tratadas en las secciones anteriores, se han presentado técnicas y métodos para determinar las propiedades de las proposiciones. Estas técnicas se basan en el significado semántico de las expresiones, es decir, basadas en la interpretación de su estructura semántica. La lógica proposicional, proporciona técnicas y métodos alternativos que se basan en la manipulación a nivel sintáctico de la formula proposicional, sin considerar sus posibles valores semánticos. Esa manipulación sintáctica parte de una serie de reglas y axiomas, a partir de los cuales se pueden realizar conclusiones sobre una o varias fórmulas proposicionales.

A continuación se presenta dos tipos de pruebas deductivas: Sistemas de Gentzen y Sistemas de Hilbert.

2.11.1 Sistema de Gentzen

Este sistema deductivo fue propuesto por Gerhard Gentzen en "Untersuchungen Über Das Logische Schliessen" (Investigaciones sobre la deducción lógica)', "Mathematische Zeitschrift", (1934). Este sistema deductivo se basa en un axioma y un conjunto de reglas de inferencia. Las reglas de inferencia son un conjunto de procedimientos que permiten concluir o determinar alguna situación basada en un conjunto de afirmaciones o premisas. Es decir, el sistema deductivo permite llegar a una conclusión a partir de un conjunto de premisas definidas.

En el sistema de Gentzen el axioma establece que un conjunto de fórmulas debe contener al menos un plan complementario de literales.

Las reglas de inferencia parten de la estructura: premisas – conclusión.

$$\frac{\text{Premisas}}{\text{Conclusión}}$$

Para obtener una conclusión a partir de un conjunto de premisas, es necesario aplicar reglas de inferencia que permiten la manipulación sintáctica de las premisas. Las reglas de inferencia en el sistema de Gentzen son de tipo α (alfa) o β (beta). Las reglas de inferencia son presentadas en Mordechai Ben-Ari, Mathematical Logic for Computer Science (Rehovot: Springer-Verlag London, 2012). y retomadas en este texto. (tabla 2.26)

α	α_1	α_2
A	$\neg\neg A$	
$A_1 \vee A_2$	A_1	A_2
$\neg(A_1 \wedge A_2)$	$\neg A_1$	$\neg A_2$
$(A_1 \rightarrow A_2)$	$\neg A_1$	A_2
$\neg A_1 \leftrightarrow A_2$	$\neg(A_1 \rightarrow A_2)$	$\neg(A_2 \rightarrow A_1)$

Tabla 2.26: α - fórmulas Ben-Ari.

Las β-fórmulas para el sistema de Hilbert se presentan a continuación. (tabla 2.27)

β	β_1	β_2
$B_1 \wedge B_2$	B_1	B_2
$\neg(B_1 \vee B_2)$	$\neg B_1$	$\neg B_2$
$\neg(B_1 \rightarrow B_2)$	B_1	$\neg B_2$
$(B_1 \leftrightarrow B_2)$	$(B_1 \rightarrow B_2)$	$(B_2 \rightarrow B_1)$

Tabla 2.27: β - fórmulas Ben-Ari.

Con ese conjunto de reglas de inferencia es posible probar una formula proposicional a partir de un conjunto de premisas. Por ejemplo se desea demostrar $(p \vee q) \vee p$. Entonces la prueba de Gentzen se realiza así: (tabla 2.28)

Por ejemplo si se desea demostrar $(\neg q \rightarrow \neg p) \vee (p \rightarrow q)$. Entonces la prueba de Gentzen se puede realizar de la siguiente manera: (tabla 2.29)

1. p, q, $\neg p$	Axioma
2. p, q, $\neg q$	Axioma
3. $(p \vee q)$, p	α - $\rightarrow$ (1)
4. $(p \vee q) \vee p$	α - $\rightarrow$ (3)

Tabla 2.28: Prueba de Gentzen para $(p \vee q) \vee p$

1. $\neg q$, $\neg p$, q	Axioma
2. q, $\neg p$, p	Axioma
3. $(\neg q \rightarrow \neg p)$, p, q	α - $\rightarrow$ (1)
4. $(\neg q \rightarrow \neg p)$, $(p \rightarrow q)$	α - $\rightarrow$ (2)
5. $(\neg q \rightarrow \neg p) \vee (p \rightarrow q)$	α - $\vee$ (2)

Tabla 2.29: Prueba de Gentzen para $(\neg q \rightarrow \neg p) \vee (p \rightarrow q)$

2.11.2 Sistemas de Hilbert

Un sistema de Hilbert es un sistema deductivo que también se constituye de un conjunto de axiomas y una regla de Inferencia denominada el Modus Ponens. El Modus Ponens es una regla de inferencia que la cual a partir de un condicional y el antecedente del condicional, es posible deducir el consecuente del condicional. Si se tiene el condicional $(p \rightarrow q)$ y el antecedente del condicional p, entonces la regla del Modus Ponens , concluye en consecuente.

1.	$p \rightarrow q$	P
2.	p	P
3.	q	$MP_{1,2}$ (2)

El sistema de Hilbert parte de los siguientes axiomas:

Axioma 1: $\vdash (p \rightarrow (q \rightarrow p))$

Axioma 2: $\vdash (p \rightarrow (q \rightarrow r)) \rightarrow ((p \rightarrow q) \rightarrow (p \rightarrow r))$

Axioma 3: $\vdash (\neg q \rightarrow \neg p) \rightarrow (p \rightarrow q)$

Por ejemplo si se desea demostrar $(r \wedge s)$, a partir de la formula $(p \wedge \neg q) \rightarrow (r \wedge s)$ y $(p \wedge \neg q)$, entonces se puede aplicar Modus Ponens, llegando a la conclusión $(r \wedge s)$.

1.	$(p \wedge \neg q) \rightarrow (r \wedge s)$	P
2.	$(p \wedge \neg q)$	P
C.	$(r \wedge s)$	$MP_{1,2}$ (2)

.:Actividad 2.7.

1. Dadas las siguientes fórmulas, demostrar si es posible por medio del sistema de Gentzen.
   - $(p \rightarrow q) \rightarrow (\neg p \wedge q)$
   - $\neg((q \rightarrow \neg p) \rightarrow \neg p)$
   - $\neg((p \vee q) \rightarrow (\neg p \vee q)$

- $\neg(p \to q) \vee (\neg p \to \neg q)$
- $((p \vee q) \to q) \to (p \to q)$

2. Deducir por medio del sistema deductivo de Hilbert.

- $p \to (\neg p \to q)$
- $\neg((q\neg\neg p) \to q$
- $\neg(p \to q) \to (\neg p \to \neg q)$
- $((p \to q) \to q) \to (p \to q)$

CAPÍTULO 3

Argumentos, reglas de inferencia y límites de la lógica

> El argumento se asemeja al disparo de una ballesta, es igual de efectivo dirigido a un gigante que a un enano.
>
> Francis Bacon.
> 1561 – 1626

Al finalizar el estudio de éste capítulo el estudiante estará en condiciones de:

- Identificar los principales métodos de razonamiento con sus componentes fundamentales: premisas, conclusión y proceso de inferencia.
- Aplicar los conceptos fundamentales del análisis de razonamientos en el cálculo proposicional.
- Determinar cuándo un argumento tiene propiedades de validez y no validez.
- Verificar la validez o no de un argumento mediante el uso de diferentes reglas de inferencia.
- Aplicar las principales reglas de inferencia en el cálculo proposicional.

3.1 Introducción

En este capítulo se trabajará con elementos fundamentales de la lógica proposicional para establecer las propiedades de argumento a partir de un conjunto de afirmaciones. Para esa deducción, se presentan diferentes reglas de inferencia, las cuales determinan las propiedades de las proposiciones basadas en una manipulación sintáctica. El análisis de estas propiedades no se infiere a partir de los posibles valores de verdad que puede tomar la fórmula proposicional. Con esto se muestra en este capítulo una diferencia en la forma como se han trabajado las proposiciones, los operadores booleanos y las deducciones.

Inicialmente se presentan el concepto de análisis de razonamiento, para posteriormente mostrar las principales reglas de inferencia que se trabajan en la lógica. Como elemento diferenciador en este capítulo, no se analizará verdad o falsedad de una proposición a partir de los operadores booleanos. Se analizará si la veracidad de las premisas tiene relación directa con la verdad de la conclusión.

3.2 Análisis de razonamientos y silogismos

Un argumento se compone de un conjunto de enunciados en un lenguaje formal. Los enunciados son un conjunto de premisas a partir de las cuales se puede establecer una conclusión. El concepto de argumento y razonamiento se trabajará como sinónimos en este libro. Los silogismos constan de dos premisas y una conclusión, siendo la conclusión el resultado de una inferencia deductiva ente las premisas.

Mediante los razonamientos es posible resolver diversos problemas que se plantean desde la lógica, teniendo como referentes proposiciones que responden a la estructura premisas-conclusión.La conclusión se deriva a partir de las premisas y las premisas sirven como justificación o evidencia de la conclusión. Si el razonamiento es válido, no puede aceptarse la verdad de las premisas sin aceptar también la validez de la conclusión (Baldwin & Scragg, 2004). La lógica no estudia la verdad o falsedad de las premisas y la conclusión, sino que estudia si la verdad de las premisas implica la verdad de la conclusión (Ramírez, 2003).
A continuación se presentan razonamientos concebidos como válidos, en los cuales se presentarán las premisas y posteriormente la conclusión. Para denotar las premisas se antepone la notación P_i para cada una de ellas y la letra C para denotar la conclusión. Para los siguientes ejemplos se asume que si las premisas son verdaderas, también lo será la conclusión. (tabla 3.1)

Premisas	**Conclusión**
P_1. Todos los hombres son mortales P_2. Pedro es hombre	Pedro es mortal
P_1. Los deportistas son personas constantes P_2. Las personas constantes son exitosas	Los deportistas son exitosos
P_1. Si descanso lo suficiente estaría feliz P_2. No estoy feliz	No descanso lo suficiente
P_1. Los grillos son insectos P_2. Los insectos no tienen cola	Los grillos no tienen cola

Tabla 3.1

También es común trabajar con razonamientos considerados no válidos. (tabla 3.2)

Premisas	**Conclusión**
P_1. Todos los tigres son carnívoros P_2. Todos los tigres son cuadrúpedos	Algunos cuadrúpedos son carnívoros

Tabla 3.2

La conclusión tiene un contenido existencial y se menciona que solo algunos, es decir, que existe cuadrúpedos que son carnívoros, pero en las premisas se afirma que todos los tigres son carnívoros.

El siguiente caso no se puede afirmar que la persona no está alegre porque no se compró una vivienda, puede no estar alegre por otro motivo. Por lo tanto si la primera premisa es válida, lo serán ambas y

no la conclusión. (tabla 3.3)

Premisas	**Conclusión**
P_1. Si comprara una vivienda estaría feliz P_2. No me compre una vivienda	No estoy feliz

Tabla 3.3

El razonamiento lógico es un conjunto de juicios que mantienen entre sí relaciones lógicas de tal forma que partiendo de juicios denominados premisas es posible llegar deductivamente a un nuevo juicio denominado conclusión. La obtención de la conclusión, asegura la validez de la juicios que componen las premisas (Muñoz, 2011).

Un razonamiento es válido si todas las premisas son válidas y por lo tanto su conclusión también lo es. Si un razonamiento tiene la estructura:

$$p_1 \wedge p_2 \wedge p_3 \wedge p_4 \wedge \cdots \wedge p_n \rightarrow q$$

En donde la conclusión q es verdadera y también lo son las premisas p_i, entonces se puede afirmar que es un razonamiento válido. Así mismo, se podría afirmar que el razonamiento $p_1 \wedge p_2 \wedge p_3 \wedge p_4 \wedge \cdots \wedge p_n \rightarrow q$, es una tautología.

Para determinar si un razonamiento es válido, se representan las premisas y la conclusión mediante fórmulas proposicionales. Una vez formalizados se establece el esquema completo del razonamiento y se procede a aplicar el método de las tablas de verdad. A continuación se presenta un ejemplo en donde se determina la validez del razonamiento.

P_1. Si tuviera beca estaría estudiando
P_2. No estoy estudiando

C. No tengo beca

La representación de las premisas en forma de proposición es:

p. Tengo beca
q. Estoy estudiando

La representación formal de las premisas y la conclusión sería:

$p \rightarrow q$
$\neg q$

$\neg p$.

Una forma de comprobar la validez del razonamiento es mediante la tabla de verdad. La fórmula: $((p \rightarrow q) \wedge \neg q) \rightarrow \neg p$, es una tautología y por lo tanto, es un razonamiento válido. (tabla 3.4)
Por ejemplo, si se tiene el siguiente razonamiento con sus premisas y conclusión, se desea verificar si este es válido.

P_1. Si tuviera beca estaría estudiando
P_2. No tengo beca

C. No estoy estudiando

p	q	$\neg p$	$\neg q$	$p \to q$	$(p \to q) \wedge \neg q$	$((p \to q) \wedge \neg q) \to \neg p$
V	V	F	F	V	F	V
V	F	F	V	F	F	V
F	V	V	F	V	F	V
F	F	V	V	V	V	V

Tabla 3.4: Tabla de verdad de $((p \to q) \wedge \neg q) \to \neg p$

Para ello es necesario es pasar las premisas a proposiciones.

p. Tengo beca
q. Estoy estudiando

Formalizando tenemos:

$p \to q$
$\neg p$

$\neg q$

La validez del razonamiento se comprueba analizando su tabla de verdad. La fórmula: $((p \to q) \wedge \neg q) \to \neg p$. De la tabla de verdad se puede afirmar que no es una tautología y por lo tanto no es un razonamiento válido. (tabla 3.5)

p	q	$\neg p$	$\neg q$	$p \to q$	$(p \to q) \wedge \neg p$	$((p \to q) \wedge \neg p) \to \neg q$
V	V	F	F	V	F	V
V	F	F	V	F	F	V
F	V	V	F	V	V	F
F	F	V	V	V	V	V

Tabla 3.5: Tabla de verdad de $((p \to q) \wedge \neg q) \to \neg p$

Se desea determinar si el siguiente argumento es válido.

La biblioteca municipal es pública, o lo es la biblioteca universitaria. Si la biblioteca municipal es pública, entonces tiene subsidio del gobierno. La biblioteca municipal no tiene subsidio del gobierno. Por lo tanto, la biblioteca universitaria es pública.

Utilicemos las siguientes proposiciones atómicas:

bmp biblioteca municipal es pública
ep biblioteca universitaria es pública
ca biblioteca municipal tiene subsidio

La representación de los proposicional de las premisas y la conclusión es:

$bmp \vee ep$
$bmp \to ca$
$\neg ca$
---
ep.

Al analizar la conclusión, se observa que todas las interpretaciones son verdaderas, por lo tanto es una fórmula válida. (tabla 3.6)

bmp	ep	ca	$bmp \vee uqp$	$bmp \rightarrow ca$	$\neg ca$	$(bmp \vee uqp) \wedge (bmp \rightarrow ca) \wedge \neg ca \rightarrow ep$
V	V	V	V	V	F	V
V	V	F	V	F	V	V
V	F	V	V	V	F	V
F	F	F	V	F	V	V
F	V	V	V	V	F	V
F	V	F	V	V	V	V
F	F	V	F	V	F	V
F	F	F	F	V	V	V

Tabla 3.6: Tabla de verdad de $(bmp \vee uqp) \wedge (bmp \rightarrow ca) \wedge \neg ca \rightarrow ep$

.:Actividad 3.1.

1. Determine cuáles de los siguientes razonamientos corresponden a razonamientos válidos ó a razonamientos no válidos.

2. Determine si las siguientes expresiones proposicionales son razonamientos válidos o no válidos.

   - $(\neg p \wedge \neg(p \rightarrow q)) \rightarrow \neg q$
   - $\neg(\neg p \wedge (p \rightarrow q)) \rightarrow q$
   - $(\neg q \wedge (p \rightarrow \neg q)) \rightarrow p$
   - $\neg(\neg q \wedge \neg(p \neg q)) \rightarrow \neg p$
   - $((p \vee \neg q) \wedge \neg q) \rightarrow p$
   - $((\neg p \vee \neg q) \wedge \neg p) \rightarrow q$

3. Determine si las siguientes premisas y la conclusión, son válidas o no válidas.

   Si usted es disciplinado, tendrá reconocimiento
   Si usted tiene reconocimiento, será recompensado

   Si usted es disciplinado, será recompensado

   Si no tuviera deudas estaría ahorrando dinero
   Estoy ahorrando dinero

   No tengo deudas

3.3 Reglas de Inferencia

Una inferencia permite determinar una situación de acuerdo a un conjunto de premisas. Esas situaciones surgen de un conjunto de proposiciones (premisas), a partir de las cuales se genera una proposición conocida como conclusión. Es decir, la conclusión debe estar contenida con base en las premisas preestablecidas.

Premisas	**Conclusión**	**válido**	
		Si	**No**
P_1. Los deportistas son disciplinados P_2. Los disciplinados cumplen con los compromisos	Los deportistas cumplen con los compromisos		
P_1. Los deportistas no son disciplinados P_2. Los disciplinados no cumplen con los compromisos	Los deportistas no cumplen con los compromisos		
P_1. Los deportistas son disciplinados P_2. Los disciplinados no cumplen con los compromisos.	Los deportistas cumplen con los compromisos		
P_1. Si no es disciplinado, tendrá reconocimiento P_2. Si tiene reconocimiento, será feliz	Si es disciplinado, será feliz		
P_1. Si usted no es disciplinado, no tendrá reconocimiento P_2. Si usted no tiene reconocimiento, será feliz	Si es disciplinado, será feliz		
Si tuviera reconocimiento estaría contento, No estoy contento	No tengo reconocimiento		

Tabla 3.7

Son diversos los mecanismos de inferencia que pueden ser usados para demostrar propiedades de las formulas en el cálculo proposicional, tales como el método de resolución, los tableros semánticos, las tablas de verdad, las propias reglas de inferencias, entre otros. A través de las reglas de inferencia, es posible establecer la validez de un argumento sin usar tablas de verdad.

Los tres elementos que fundamentan los razonamientos y permiten realizar las inferencias son:

- Premisas: hipótesis que se plantean, supuestos básicos, proposiciones, fórmulas, enunciados, siempre considerados verdaderos.
- Conclusión: proposición que se desea probar a partir de las premisas.
- Reglas de Inferencia: conjuntos de pasos que se utilizar para obtener otras conclusiones de acuerdo a las premisas.

En las siguientes subsecciones se presentan reglas de inferencias comúnmente utilizadas en el cálculo proposicional. El esquema para mostrar cómo se aplican las reglas de inferencia parte de las

premisas numeradas y posteriormente se presentan las reglas de inferencia aplicadas para llegar a una determinada conclusión.

3.3.1 Modus Ponens

La regla de Modus Ponens (MP), es también conocida como método de afirmación (afirmando-afirmo). Esta regla establece que dado un condicional y el antecedente del condicional, se puede establecer el consecuente del condicional. Si se tiene la premisa verdadera $(p \rightarrow q)$ y se asume que se cumple la premisa p, entonces se puede concluir q. La regla de MP permite afirmar solo en el caso que el antecedente se esté afirmando.
Partiendo del esquema planteado en la sección anterior para aplicar reglas de inferencia, se tiene como primera premisa $(p \rightarrow q)$ y como segunda premisa p, entonces se aplica la regla MP, y se obtiene q.

1	$p \rightarrow q$	P
2	p	P
C	q	$MP_{1,2}$

Este esquema puede ser aplicado a un conjunto de premisas que estén expresadas en lenguaje natural. Por ejemplo si se tienen las premisas: Si x es un número es par, entonces x es divisible por 2. La regla del MP se puede aplicar así:

1	Si x es un número par, entonces x es divisible por 2	P
2	x es un número par	P
C	x es divisible por 2	$MP_{1,2}$

La regla del MP puede ser también aplicada a proposiciones compuestas, es decir, su uso no está limitado exclusivamente a proposiciones atómicas. Por ejemplo si se tienen las premisas: Si trabajo y no gano ahorro dinero, entonces me desmotivo y me endeudo. Es el caso que trabajo y no ahorro dinero.

1	Si trabajo y no gano ahorro dinero,entonces me desmotivo y me endeudo	P
2	Es el caso que trabajo y no ahorro dinero	P
C	Por lo tanto,me desmotivo y me endeudo	$MP_{1,2}$

Formalizando el caso anterior, la representación es la siguiente:

p : trabajo

q : ahorro

r : me desmotivo

s : me endeudo

1	$(p \wedge \neg q) \rightarrow (r \wedge s)$	P
2	$(p \wedge \neg q)$	P
C	$(r \wedge s)$	$MP_{1,2}$

La regla del Modus Ponens se puede aplicar repetidas veces en un conjunto de premisas. Por ejemplo se desea demostrar $(p \vee q)$, considerando las premisas P_1, P_2, P_3, P_4. Se aplica MP entre P_2 y P_3, obteniéndose una primera conclusión C_1 la cual es adicionada al conjunto y se enumera con la línea 4. Se vuelve aplicar MP entre la premisa P_1 y C_1, pudiendo demostrar $(p \vee q)$ a partir de las premisas

dadas. La aplicación de MP se puede aplicar en un orden indistinto a las premisas. En el ejemplo la P_4 nunca es usada para obtener alguna conclusión:

1	$r \to (p \vee q)$	P
2	$(q \vee p) \to r$	P
3	$(q \vee p)$	P
4	q	P
C, 5	r	$MP_{2,3}$
C	$(p \vee q)$	$MP_{1,4}$

.:Actividad 3.2.

1. Determinar la validez del argumento aplicando tablas de verdad.

p	q	r	s	$(p \wedge \neg q) \to (r \wedge s)$	$((p \wedge \neg q) \to (r \wedge s) \wedge (r \wedge s)) \to \neg(r \wedge s)$
V	V	V	V		
V	V	V	F		
V	V	F	V		
V	V	F	F		
V	F	V	V		
V	F	V	F		
V	F	F	V		
V	F	F	F		
F	V	V	V		
F	V	V	F		
F	V	F	V		
F	V	F	F		
F	F	V	V		
F	F	V	F		
F	F	F	V		
F	F	F	F		

Tabla 3.8

2. Dada la siguiente premisa aplique Modus Ponens para obtener la conclusión

1	Si Alberto esta en su trabajo,entonces Alberto está en la Universidad	P
2	Alberto está en su oficina trabajando	P
C		$MP_{\square}$

3. Dada las siguientes premisas, demostrar r si:

1	$p \to q$	P
2	$q \to r$	P
3	q	P
4		
C		

4. Dada la siguiente lista de premisas, demostrar $\neg\neg r$ si:

1	$p \to \neg q$	P
2	$\neg q \to \neg\neg r$	P
3	p	P
4	p	P
C		

3.3.2 Tollendo Tollens

La regla del Tollendo Tollens (TT), es conocida como la regla del negando - niego. Es posible aplicarla a expresiones condicionales, en la cual a partir de un condicional y la negación del consecuente del condicional, se puede concluir la negación del antecedente del condicional. Es decir, si en un condicional el consecuente esta negado, es posible negar el antecedente, debido a que si el consecuente no es verdadero, su antecedente tampoco lo es.

Si en un conjunto de premias se tiene como la primera $p \to q$, y como segunda premisa $\neg q$, se pueda aplicar TT, obteniéndose $\neg p$. Con esta estructura de la implicación, solo se puede realizar una afirmación a partir del antecedente y la negación solo a partir del consecuente.

1	$p \to q$	P
2	$\neg q$	P
C	$\neg p$	$TT_{1,2}$

Por ejemplo, si se tiene la premisa: Si Alfonso es una persona mayor de edad, entonces tiene cédula de ciudadanía. Se puede aplicar TT así:

1	Si Alfonso es una persona mayor de edad,entonces tiene cédula de ciudadanía	P
2	Alfonso no tiene cédula de ciudadanía	P
C	Alfonso no es una persona mayor de edad	$TT_{1,2}$

Por ejemplo, si se tienen la premisa: Si los síntomas de la enfermedad continúan, entonces se debe realizar el tratamiento. La regla de TT se aplica como se presenta a continuación:

1	Si los síntomas de la enfermedad continúan,entonces la persona se debe realizar el tratamiento	P
2	La personas no se debe realizar el tratamiento	P
C	Por lo tanto,los síntomas de la enfermedad no continúan	$TT_{1,2}$

La regla de Tollendo Tollens como en el caso de las reglas de inferencia se puede aplicar a una lista de premisas. Por ejemplo, se desea demostrar $p \neq 0$, a partir de un conjunto de premisas. Inicialmente se aplica MP entre las premisas 1 y 4, generándose la conclusión $p = r$. Luego se aplica MP ente 2 y 5, generándose la conclusión $p = 1$. Finalmente se aplica Tollendo Tollens entre las líneas 3 y 6, obteniendo la conclusión $(p \neq 0)$.

1	$(p = q) \rightarrow (p = r)$	P
2	$(p = r) \rightarrow (p = 1)$	P
3	$(p = 0) \rightarrow (p \neq 1)$	P
4	$p = q$	P
C, 5	$p = r$	$MP_{1,4}$
C, 6	$p = 1$	$MP_{2,5}$
C	$p \neq 0$	$TT_{3,6}$

.:Actividad 3.3.

1. De acuerdo a la siguiente expresión, formalícelas en lógica proposicional y aplique la regla de Tollendo Tollens.
   - Si tiene problemas económicos, entonces no podrá acceder al crédito hipotecario.
   - Si trabaja en cantidad, no tendrá tiempo para terminar el posgrado.
   - Si la moto no prende, entonces tiene problemas eléctricos.
2. Aplique las reglas de inferencia a las siguientes premisas, para demostrar r.

1	$p \rightarrow q$	P
2	$\neg q \rightarrow \neg\neg r$	P
3	$\neg p$	P
4		
5		
C		

3. Aplique las reglas de inferencia a las siguientes premisas, para demostrar si es posible $p = 1$.

1	$(p = s) \rightarrow (p = w)$	P
2	$(p = w) \rightarrow (p = 0)$	P
3	$(p = 1) \rightarrow (p \neq 0)$	P
4	$p = s$	P
5		
6		
C		

3.3.3 Doble negación

La regla de la Doble Negación (DN) establece que la doble negación en una proposición equivale a la misma proposición. Cuando se aplica esta regla de inferencia es posible pasar directamente de una premisa a la conclusión. Los siguientes ejemplos muestran la doble negación en expresiones de lenguaje natural.

- Nadie sabe nada.
- No voy nunca a viajar en avión.
- No es verdad que la empresa no cuente con certificación de calidad.

La regla de la DN puede adquirir dos formas, en el primer caso la conclusión se deriva de la negación de su negación, es decir la premisa se niega doblemente.

1	q	P
C	$\neg\neg q$	DN_1

Ejemplos de la aplicación para la primera forma se muestra a continuación:

1	Luis estudia para la certificación	P
C	No sucede que Luis no estudia para la certificación	DN_1

1	Es cierto que tengo efectivo para pagar	P
C	No es cierto que no tenga efectivo para pagar	DN_1

En la segunda forma, la proposición doblemente negada, es equivalente a una proposición afirmativa. Por lo tanto ambas negaciones son eliminadas.

1	$\neg\neg q$	P
C	q	DN_1

Ejemplos de aplicación de estas formas se presentan a continuación:

1	No es cierto que Felipe no pagó los impuestos	P
C	Felipe pagó los impuestos	DN_1

1	No ocurre que Juliana no es una buena programadora	P
C	Juliana es una buena programadora	DN_1

Cuando se tiene un conjunto de premisas, es posible aplicar diferentes reglas de inferencia con el propósito de demostrar una determinada conclusión. Por ejemplo, si se desea demostrar $\neg\neg r$, partiendo de las premisas:

1	$q \to r$	P
2	q	P
C_1, 3	$q \to r$	$MP_{1,2}$
C	$\neg\neg r$	DN_3

En este caso se aplica MP entre las premisas 1 y 2, a partir de las cuales se obtiene una primera conclusión r, posteriormente se aplica la doble negación y se obtiene la conclusión $\neg\neg r$.

.:Actividad 3.4.

1. Aplique doble negación a las siguientes proposiciones:
   - No ocurre que Luz no sabe hacer traducciones.
   - No es cierto que ella no tenga incentivos económicos para viajar.
   - Es verdad que el programa académico está acreditado

- No voy nunca a estudiar física.

2. Aplique las reglas de inferencia a las siguientes premisas, para demostrar si es posible $\neg\neg p$.

1	$p \rightarrow \neg q$	P
2	$\neg q \rightarrow p$	P
3	p	P
4		
5		
C		

3. Aplique las reglas de inferencia a las siguientes premisas, para demostrar si es posible $\neg\neg p$.

1	$(q \rightarrow p)$	P
2	$(p \rightarrow q)$	P
3	q	P
4		
5		
C		

3.3.4 Regla de adjunción

La regla de Adjunción (A), es una regla de inferencia en la cual se establece que si dos premisas son verdaderas, su conjunción también lo es. El orden en el cual aparecen las premisas en el razonamiento es indiferente. La representación básica puede tomar dos formas:

1	p	P
2	q	P
C	$p \wedge q$	$A_{1,2}$

ó

1	p	P
2	q	P
C	$q \wedge p$	$A_{1,2}$

En el siguiente conjunto de premisas se muestra mediante la conjunción, como se conserva la propiedad de validez en la conclusión. (tabla 3.9)
Cuando se aplica la Adjunción se obtiene una conclusión compuesta que también es verdadera. Es decir, mediante esta regla llega a la conclusión directamente desde las premisas.

A continuación se muestra otro ejemplo con la estructura planteada.

Premisa 1 (P_1)	Premisa 2 (P_2)	Adjunción A_{p_1,p_2}
El plan de estudios de la carrera.	La formación de los maestros es pertinente.	El plan de estudios de la carrera y la formación de los maestros es pertinente
El promedio de notas del estudiante es sobresaliente	La actitud del estudiante no es la ideal.	El promedio de notas del estudiante es sobresaliente, sin embargo, la actitud del estudiante no es la ideal.
Está lloviendo	Es posible salir a caminar	Está lloviendo, pero es posible salir a caminar.
El Decano tiene un plan de gestión.	El Decano cuenta con un regular grupo de trabajo.	El Decano tiene un buen plan de gestión, no obstante, cuenta con un regular grupo de trabajo

Tabla 3.9

1	Los Ingenieros tienen tarjeta profesional	P
2	La tarjeta profesional es indispensable para participar en licitaciones	P
C	Los ingenieros tienen tarjeta profesional y la tarjeta es indispensable para participar en las licitaciones	$A_{1,2}$

La regla de Adjuncion también puede ser aplicada a un listado de premisas. Por ejemplo se desea verificar que $(p \wedge q) \wedge (q \wedge r)$, es una conclusión valida a partir de las premisas de las líneas 1,2 y 3.

1	p	P
2	q	P
2	r	P
C_1, 4	$p \wedge q$	$A_{1,2}$
C_2, 5	$q \wedge r$	$A_{2,3}$
C	$(p \wedge q) \wedge (q \wedge r)$	$A_{1,2}$

Por medio de la tablas de verdad se puede asumir que $(p \wedge q) \wedge (q \wedge r)$, es válida a partir de las premisas válidas.

.:Actividad 3.5.

1. Dadas las siguientes premisas, aplicar entre ellas la regla de Adjunción.
   - Andres es Arquitecto de Software
   - Las empresas de tecnología contratan personal calificado.

- El gobierno incentiva la formación de las personas en certificaciones de calidad
- El país necesita desarrolladores de software.

2. Demostrar $\neg\neg(\neg p \wedge (q \wedge r))$ si se tienen las siguientes premisas. Se debe indicar el nombre de la regla aplicada.

1	$\neg p$	P
2	$(p \wedge r)$	P
3		
4		
C		

3. Demostrar $\neg\neg(\neg\neg((\neg p \wedge q) \wedge (q \wedge r) \wedge s))$ si se tienen las siguientes premisas. Se debe indicar el nombre de la regla aplicada.

1	$\neg p$	P
2	q	P
3	r	
4	s	
5		
C		

3.3.5 Conmutativa

La regla Conmutativa (C) establece que el orden en el cual aparecen las proposiciones en una conjunción o en una disyunción, no altera las características de la fórmula proposicional. No sucede igual con la implicación, pues el alterar el orden las proposiciones modifica las propiedades de la formula.

Para la disyunción, cambiar el orden de los premisas, permite pasar directamente a la conclusión.

1	$p \wedge q$	P
C	$q \wedge p$	C_1

Por ejemplo, a la siguiente premisa $(p \vee q \vee r) \wedge (\neg p \vee r)$, se le puede aplicar la regla conmutativa, y en la conclusión aparece alterado el orden de las premisas.

1	$(p \vee q \vee r) \wedge (\neg p \vee r)$	P
C	$(\neg p \vee r) \wedge (p \vee q \vee r)$	C_1

Para la conjunción, cambiar el orden de los premisas, permite pasar directamente a la conclusión.

1	$p \vee q$	P
C	$q \vee p$	C_1

Por ejemplo, a la siguiente premisa $(p \vee q \vee \neg r) \vee (r)$, se le puede aplicar la regla conmutativa, y en la conclusión aparece alterado el orden de las premisas, conservando las propiedades de la conclusión.

$$\begin{array}{lll} 1 & (p \vee q \vee \neg r) \vee (r) & P \\ \hline C & (r) \vee (p \vee q \vee \neg r) & C_1 \end{array}$$

Igual sucede con el siguiente ejemplo al cual se le puede aplicar la regla conmutativa:

1	Alfredo es gerente de Tecnología y practica deportes extremos	P
C	Alfredo practica deportes extremos y es gerente de tecnología	C_1

La regla Conmutativa también puede ser aplicada a un conjunto de premisas. Para el siguiente caso se observa que en la línea 6 se obtiene la conclusión $p \vee s$. En línea 6 se observa que a la conclusión $s \vee p$, se le puede aplicar la regla Conmutativa y se puede verificar $p \vee s$.

$$\begin{array}{rll} 1 & r \to \neg q & P \\ 2 & \neg\neg q & P \\ 3 & \neg(s \vee p) \to r & P \\ C_1,\, 4 & \neg r & TT_{1,2} \\ C_2,\, 5 & \neg\neg(s \vee p) & TT_{3,4} \\ C_3,\, 5 & s \vee p & DN_5 \\ \hline C & p \vee s & C_6 \end{array}$$

.:Actividad 3.6.

1. Dadas las siguientes premisas, aplique la regla conmutativa de manera que la conclusión sea verdadera.

Premisa 1 (P_1)	Premisa 2 (P_2)	Conmutativa C_{p_1,p_2}
La carrera tiene muchos kilómetros de montaña.	Los ciclistas se han preparado durante un mes.	
El promedio de notas del estudiante es bajo	El estudiante tiene malos fundamentos en matemáticas	
Java es un lenguaje de programación	Son diversas las plataformas para java	
La empresa de tecnología tiene muchos ingenieros de sistemas	La ingeniería de sistemas se está expandiendo a nivel mundial.	

Tabla 3.10

2. Dadas las siguientes premisas, formalicelas y aplique entre otras la regla conmutativa.

1	9 es un número perfecto y 9 es un número par	P
C		

3. Dadas las siguientes premisas, demuestre que $s \vee p$. Debe aplicar las reglas de inferencia necesarias.

1	$x \rightarrow \neg q$	P
2	$\neg\neg q$	P
3	$\neg(p \vee s) \rightarrow x$	P
C		

3.3.6 Tollendo Ponens

La regla de Tollendo Ponens (TP) establece que cuando una de las proposiciones de la disyunción esta negada, entonces la otra proposición afirma a la otra proposición, es decir, que cuando en la disyunción se niega una proposición, la otra se asume como verdadera. Esta regla también es conocida como la regla negando – afirmo.

Si uno de los miembros de una disyunción es negado, el otro miembro queda automáticamente afirmado, ya que uno de los términos de la elección ha sido descartado (Sanchéz, 2007).

1	$p \vee q$	P
2	$\neg p$	P
C	q	$TP_{1,2}$

Mediante la regla también se puede concluir:

1	$p \vee q$	P
2	$\neg q$	P
C	p	$TP_{1,2}$

En el siguiente conjunto de premisas se muestra como mediante la aplicación de la regla de Tollendo Ponens, la conclusión continua siendo verdadera. (tabla 3.11)
A continuación se muestra otro ejemplo donde se aplica Tollendo Ponens bajo la estructura planteada.

1	6 es un número perfecto o 6 es múltiplo de 4	P
2	6 no es un número multiplo de 4	P
C	6 es un numero perfecto	$TP_{1,2}$

.:Actividad 3.7.

Premisa 1 (P_1)	Premisa 2 (P_2)	Conmutativa TP_{p_1,p_2}
El estudiante hizo la pasantía en México o en Chile.	El estudiante no ha estudiado en Chile.	Por lo tanto, el estudiante hizo la pasantía en México.
El avión tiene la cabina económica o ejecutiva	El avión no tiene cabina ejecutiva	Por lo tanto, el avión tiene la cabina económica.
Los proyectos de investigación son en convenio o son propios	Los proyectos de investigación no son en convenio.	Los proyectos de investigación son propios.
Al menos debe pagar el crédito de vivienda o el de libranza.	No paga el crédito de vivienda.	Por lo tanto, paga el crédito de libranza.

Tabla 3.11

Premisa 1 (P_1)	Premisa 2 (P_2)	Conmutativa TP_{p_1,p_2}
La persona declaro renta o no cumplía con los requisitos	La persona no pago la renta.	
El estudiante conocía el lenguaje de programación o tenía experiencia profesional.		
Los planes turísticos estaban baratos o el destino era muy atractivo.		

Tabla 3.12

1. Dadas las siguientes premisas, aplicar entre ellas la regla de Tollendo Ponens.

2. Demostrar $p < 4$ si se tienen las siguientes premisas.

1	$(p < w) \vee (p = w)$	P
2	$(p = w) \rightarrow (w \neq 4)$	P
3	$((p < w) \wedge (w = 4)) \rightarrow p < 4$	P
4	$w = 4$	P

C

3.3.7 Reglas de simplificación

En esta sección se presentan dos reglas de simplificacion, la primera esta asociada a la simplificacion en la conjuncion y la segunda a la simplificación en la disyunción.

La regla Simplificación I (SI), establece que de la conjunción de dos proposiciones asumidas como verdaderas, es posible concluir cada una de ellas. Si la premisa dada en la conjunción es cierta, entonces cada una de ellas se puede concluir como verdadera. Esto se puede validar desde la definición misma del operador de la conjunción. La conclusión para esta regla puede tomar las siguientes dos formas:

1	$p \wedge q$	P
C	p	S_1

O también se puede concluir:

1	$p \wedge q$	P
C	q	S_1

En el siguiente conjunto de premisas se muestra como mediante la simplificación se conserva la propiedad de validez para la conclusión. (tabla 3.13)

Premisa 1 (P)	Simplificación (S_1)
El numero 10 es divisible por 5 y por 2.	El número 10 es divisible por 5. O El numero 10 es divisible por 2.
El deportista aprobó el test físico y los exámenes de laboratorio.	El deportista aprobó el test físico. O El deportista aprobó los exámenes de laboratorio
La cédula de mi hermana es de Armenia y la de mi hermano de Calarcá	La cédula de mi hermana es de Armenia O La cédula de mi hermano es de Calarcá.

Tabla 3.13

Como se muestra en el ejemplo, de cada una de las premisas es posible obtener dos conclusiones, las cuales son consideradas verdaderas de forma independiente.
Un ejemplo con la estructura de premisas y conclusión, se presenta a continuación:

1	n estudiante con promedio de 4,5 es sobresaliente y tiene beca	P
C	Un estudiante con promedio de 4,5 es sobresaliente	S_1

1	Un estudiante con promedio de 4,5 es sobresaliente y tiene beca	P
C	Un estudiante como promedio de 4,5 tiene beca	S_1

La regla de Simplificacion I también puede ser aplicada a un listado de premisas. Por ejemplo se desea verificar que $(q \wedge r)$, es una conclusión valida a partir de las premisas dadas.

1	p	P
2	q	P
3	r	P
4	$(p \wedge q)$	P
C_1, 5	$(q \wedge r)$	$A_{2,3}$
C_2, 6	$(p \wedge q) \wedge (q \wedge r)$	$A_{4,5}$
C	$(q \wedge r)$	$SI_{5,6}$

La regla de Simplificación II (SII) establece que si una premisa es verdadera, la disyunción con cualquier otra premisa garantiza que esta continúe siendo verdadera. Eso se puede verificar mediante la definición propia del operador booleano de disyunción, el cual define que para que una proposición sea verdadera, basta con que solo una de las proposiciones sea verdadera.

La aplicación de la regla de Simplificación se puede representar de dos maneras diferentes:

1	p	P
C	$p \vee q$	$II.S_1$

O también se puede concluir

1	q	P
C	$p \vee q$	$II.S_1$

A continuación se muestra otro ejemplo con la estructura planteada.

1	6 es un número perfecto	P
C	6 es un número perfecto ó 6 es divisible por 2	$II.S_1$

También se puede afirmar lo siguiente con base en la Simplificación:

1	25 es divisible por 5	P
C	25 es divisible por 5 ó 3 es un número perfecto	$II.S_1$

Es posible aplicar la regla II de simplificación a un grupo de premisas. Por ejemplo si se tiene la proposición verdadera $(\neg r \vee q)$ y la proposición falsa $\neg r$, la disyunción entre ambas permite dar una conclusión verdadera.

1	$\neg r \vee q$	P
1	$\neg r$	P
C	$(\neg r \vee q) \vee \neg r$	$II.S_1$

La regla II de simplificación se puede usar en una lista de premisas. Si se quiere demostrar $q \vee s$, a partir de las premisas 1,2, y 3. Se puede observar que a $\neg\neg q$ se le aplica la regla de la doble negación quedando solo q, y posteriormente se aplica la regla II de simplificación con la cual se sigue garantizando que la conclusión sigue siendo verdadera.

1	$\neg s \rightarrow [(s \vee \neg\neg q) \wedge (s \wedge \neg q)]$	P
2	$\neg(p \wedge h)$	P
3	$s \rightarrow (p \wedge h)$	P
C_1, 4	$\neg s$	$TT_{2,3}$
C_2, 5	$(s \vee \neg\neg q) \wedge (s \wedge \neg q)$	$MP_{1,4}$
C_3, 6	$s \vee \neg\neg q$	IS_5
C_4, 7	$\neg\neg q$	$TP_{4,6}$
C_5, 8	q	DN_7
C	$q \vee s$	IIS_8

.:Actividad 3.8.

1. Dadas las siguientes premisas en lenguaje natural, aplicar la regla de simplificación

Premisa 1 (P)	Simplificación (S_1)
Las personas mayores de edad tienen cédula y pueden votar.	
LAS TIC pueden ser usadas en la educación y en la formación del talento humano.	
En clase ejecutiva se tiene centro de entretenimiento y bebidas ilimitadas.	

Tabla 3.14

2. Demostrar s dadas las siguientes premisas.

1	$\neg q \rightarrow [(\neg q \vee \neg s) \wedge (q \wedge \neg\neg s)]$	P
2	$\neg(r \wedge t)$	P
3	$q \rightarrow (r \wedge t)$	P

C

3. Demostrar $\neg\neg(p \wedge q)$ q dadas las siguientes premisas.

1	$\neg[(p \wedge q) \wedge r] \rightarrow (p \wedge r)$	P
2	$\neg(p \vee r)$	P

C

4. Dadas las siguientes premisas, formalicelas y aplique la regla de simplificación.

5. Aplique las reglas de inferencia a las siguientes premisas, para demostrar $(p \wedge q) \vee q$.

Premisa 1 (P)	Simplificación SII
El estudiante cancela el curso de algoritmia o de matemáticas. El estudiante no cancela el curso de algoritmia.	
El estudiante no cancela el curso de matemáticas.	
El estudiantes cancea el curso de matemáticas.	
El estudiante cancela el curso de algoritmia.	

Tabla 3.15

$$\begin{array}{lll} 1 & \neg p \rightarrow [(\neg p \vee q) \wedge (p \wedge q)] & P \\ 2 & \neg(t \wedge r) & P \\ 3 & p \rightarrow (t \wedge r) & P \\ \hline C & & \end{array}$$

3.3.8 Transitiva

La regla Transitiva (T) parte de dos proposiciones condicionales, a partir de la cual se genera otra proposición. Esa nueva proposición es otro condicional el cual se compone del antecedente de la primera proposición y el consecuente, es el consecuente de la segunda expresión condicional. La regla transitiva se presenta a continuación:

$$\begin{array}{lll} 1 & p \rightarrow q & P \\ 2 & q \rightarrow h & P \\ \hline C & p \rightarrow h & T_{1,2} \end{array}$$

La regla Transitiva se puede ejemplificar mediante las siguientes premisas: Si realizo la pasantía en México entonces me alojaré en vivienda familiar; si me alojo en vivienda familiar ahorrare dinero; por lo tanto si realizo la pasantía en México ahorraré dinero.

Formalizando el caso anterior, su representación queda de la siguiente manera:

p: Realizo la pasantía en México.
q: Me alojaré en vivienda familiar.
r: Ahorraré dinero.

$$\begin{array}{lll} 1 & p \rightarrow q & P \\ 2 & q \rightarrow r & P \\ \hline C & p \rightarrow r & T_{1,2} \end{array}$$

A continuación se muestra otro ejemplo con la aplicación de la regla Transitiva. La conclusión obtenida a partir de las premisas sigue siendo válida.

1	Si 4 es un número par entonces el cuadrado de 4 es numero par	P
2	Si el cuadrado de 4 es un número par entonces es divisible por 4	P
C	Si 4 es un número par entonces es divisible por 4	$T_{1,2}$

La regla Transitiva se puede usar en una lista de premisas. Si se quiere demostrar $p \to r$, a partir de las premisas 1,2, y 3, es posible aplicar las siguientes reglas:

1	$(q \to r) \wedge p$	P
2	$p \to q$	P
3	$\neg r$	P
4	$q \to r$	IS_1
C	$p \to r$	$T_{2,4}$

.:Actividad 3.9.

1. Dadas las siguientes premisas aplique la regla transitiva garantizando que la conclusión sea verdadera.

   - Si apruebo las asignaturas del ciclo básico entonces puedo cursar las del ciclo profesional; si curso las asignaturas del ciclo profesional podré presentar el trabajo de grado.
   - Si Alberto no aprende inglés entonces no puede viajar; si Alberto no puede viajar pierde los tiquetes aéreos.
   - Si las personas tienen buena lógica, entonces tendrán éxito en los cursos de programación; si tienen éxito en los cursos de programación, entonces podrán realizar un buen trabajo de grado.

2. Aplique las reglas de inferencia a las siguientes premisas, para demostrar $\neg p$.

1	$(q \to r) \wedge p$	P
2	$p \to q$	P
3	$\neg r$	P
4		
C		

3.3.9 Regla De Morgan

La regla de De Morgan (M) es utilizada para realizar transformación entre formulas proposicionales. Esos cambios permiten transformar una formula cuyo aperador principal es una disyunción a una formula cuyo operador principal es una disyunción, y viceversa. Cuando se pasa de una a otra, se cambian los valores de afirmación y negación de los términos de la disyunción/conjunción así como de la propia operación en conjunto (Cornejo, 2012). Para la regla de Morgan, se pueden aplicar dos reglas particulares, las cuales se presentan a continuación:

3.3.9.1 Regla I

La regla I De Morgan establece negando una conjunción se obtiene la disyunción de las negaciones de cada una de las proposiciones que esta contiene. La regla I se presenta a continuación:

$$\begin{array}{ccc} 1 & \neg(p \wedge q) & P \\ \hline C & \neg p \vee \neg q & M_1 \end{array}$$

Por ejemplo si se tiene la premisa: Es falso que, un menor tiene cédula y puede ejercer el derecho al voto. Se encuentra que las expresiones: una persona tiene cédula y puede ejercer el derecho al voto, son afirmativas, pero están unidas por la conjunción e influenciadas por la negación.

Cuando se aplica la regla De Morgan, cada una de las proposiciones queda negadas, por lo tanto ambas ya no son afirmativas. El operador original de conjunción se transforma a conjunción. Para el ejemplo anterior entonces la expresión queda de la siguiente forma:

1	Es falso que, un menor tiene cédula y puede ejercer el derecho al voto	P
C	Un menor no tiene cédula o no puede ejercer el derecho al voto	M_1

Al igual que en las anteriores reglas de inferencia, es posible aplicar la regla de De Morgan a un conjunto de premisas. Por ejemplo, en el siguiente listado se puede observar que en la línea 9 se tiene la expresión: $\neg(p = 1 \wedge q = 8)$, en donde se observa que es posible aplicar la regla de Morgan. La conclusión continua siendo verdadera, pero ambas expresiones quedan negadas así: $p \neq 1 \vee q \neq 8$.

1	$p = 1 \rightarrow q > 5$	P
2	$\neg(q \neq 8)$	P
3	$p + 5 = 7 \rightarrow p = 1$	P
4	$q > 5 \rightarrow q \neq 8$	P
5	$(p = 1 \wedge q = 8) \rightarrow p + 5 = 7$	P
C_1, 6	$p = 1 \rightarrow q \neq 8$	$T_{1,4}$
C_2, 7	$p \neq 1$	$TT_{6,2}$
C_3, 8	$p + 5 \neq 7$	$TT_{3,7}$
C_4, 9	$\neg(p = 1 \wedge q = 8)$	$TT_{5,8}$
C	$p \neq 1 \vee q \neq 8$	M_9

3.3.9.2 Regla II

La regla II De Morgan establece que negando una disyunción de proposiciones, se obtiene la negación de cada una de estas unidas por el operador de conjunción. La regla II se presenta a continuación.

$$\begin{array}{ccc} 1 & \neg(p \vee q) & P \\ \hline C & \neg p \wedge \neg q & M_1 \end{array}$$

Por ejemplo si se tiene la premisa: Es falso que, un menor tiene cédula o puede ejercer el derecho al voto. Se tiene que las expresiones: un menor tiene cédula o puede ejercer el derecho al voto, son afirmativas, están unidas por una disyunción y están negadas. Cuando se aplica la regla De Morgan, para el ejemplo se puede observar que cada una de las proposiciones ya no es afirmativa, por lo tanto, cada una de ellas queda negadas y son unidas por el operador de conjunción. La conclusión queda expresada de la siguiente manera:

1	Es falso que,un menor tiene cédula o puede ejercer el derecho al voto	P
C	Un menor de edadvoto no tiene cédula y no puede ejercer el derecho al	M_1

Diferentes reglas de inferencia han sido a las premisas que se presentan en el ejemplo. Se observa que en la línea 6, es posible aplicar la regla De Morgan a la conclusión $\neg(s \vee p)$.

1	$\neg q$	P
2	$\neg(p \wedge q)$	P
3	$\neg[(p \wedge q) \vee \neg(s \vee p)] \rightarrow q$	P
C_1, 4	$\neg\neg[(p \wedge q) \vee \neg(s \vee p)]$	$TT_{3,1}$
C_2, 5	$(p \wedge q) \vee \neg(s \vee p)$	DN_4
C_3, 6	$\neg(s \vee p)$	$TP_{2,5}$
C	$\neg s \wedge \neg p$	M_6

.:Actividad 3.10.

1. Dadas las siguientes premisas, formalicelas y aplique entre ellas la regla de Morgan I y II de manera que la conclusión sea verdadera.
   - Es falso que, un león es un animal carnívoro o herbívoro.
   - Es falso que, las redes sociales son nocivas o no estén reguladas adecuadamente.
   - No es cierto que, los dueños de los vehículos no los revisen y no quieran someterlos a la revisión de calidad.
   - Es falso que, las instituciones no se quieran acreditar y que no se quieren someter al proceso de verificación.
   - Es falso que, un cuadrado tiene 5 lados y 5 ángulos internos
2. Dadas las siguientes premisas, formalícelas y aplique reglas de inferencia que permitan demostrar que: $p \neq 1 \vee q \neq 9$.

1	$p = 1 \rightarrow q > 3$	P
2	$\neg(q \neq 9)$	P
3	$p + 3 = 7 \rightarrow p = 1$	P
4	$q > 3 \rightarrow q \neq 9$	P
5	$(p = 1 \wedge q = 9) \rightarrow p + 3 = 7$	P
C		

3.3.10 Bicondicional

La regla Bicondicional (B) permite derivar de una expresión bicondicional, dos condicionales, cuyas forma de representación se presentan a continuación.

a)

$$\begin{array}{ccc} 1 & p \leftrightarrow q & P \\ \hline C & p \rightarrow q & B_1 \end{array}$$

O también se puede expresar.

b)

$$\begin{array}{ccc} 1 & p \leftrightarrow q & P \\ \hline C & q \rightarrow p & B_1 \end{array}$$

La regla del bicondicional también establece que de una expresión bicondicional, se puede concluir los condicionales con las formas (a) y (b), unidos por el operador de disyunción, tal como se representa a continuación.

c)

$$\begin{array}{ccc} 1 & p \leftrightarrow q & P \\ \hline C & (q \rightarrow p) \wedge (p \rightarrow q) & B_1 \end{array}$$

Otra aplicación de la regla del bicondicional permite que a partir de dos expresiones condicionales independientes se pueda concluir un bicondicional.

d)

$$\begin{array}{ccc} 1 & p \leftrightarrow q & P \\ 2 & q \leftrightarrow p & P \\ \hline C & (p \leftrightarrow q) & B_{1,2} \end{array}$$

O también se puede expresar

e)

$$\begin{array}{ccc} 1 & q \leftrightarrow p & P \\ 2 & p \leftrightarrow q & P \\ \hline C & (p \leftrightarrow q) & B_{1,2} \end{array}$$

Por ejemplo si se tiene las siguientes dos premisas:

1	Si Juan puede votar ,entonces tiene cédula de ciudadania	P
2	Si juan tiene cédula de ciuadadania,entonces puede votar	P
C	Juan tienen cédula de ciudadania si y solo si puede votar	$B_{1,2}$

Se puede observar que a partir de sus condicionales, es posible concluir el condicional de forma tal que este sigue siendo verdadero. También se podría expresar así:

1	Si Juan tiene cédula de ciudadania ,entonces puede votar	P
2	Si juan puede votar ,entonces tiene cédula de ciudadania	P
C	Juan puede votar si y solo si tiene cédula de ciudadania	$B_{1,2}$

A continuación se presenta un listado de premisas al cual se le aplicaron diferentes reglas de inferencia. En la línea 6 se puede observar que la regla del bicondicional se aplica a la primera de las premisas, obteniendo se la conclusión $s \wedge q \rightarrow q$.

1	$q \leftrightarrow (s \wedge q)$	P
2	$\neg[\neg(s \wedge q) \vee r]$	P
C_1, 3	$\neg\neg(s \wedge q) \wedge \neg r$	M_2
C_2, 4	$\neg\neg(s \wedge q)$	IS_3
C_3, 5	$s \wedge q$	DN_4
C_4, 6	$s \wedge q \rightarrow q$	B_1
C	q	$PP_{5,6}$

.:Actividad 3.11.

1. Dadas las siguientes premisas, formalicelas y aplique diferentes reglas de inferencia incluyendo necesariamente el bicondicional.
   - El número x es impar par si y sólo si no es divisible por 2.
   - Un número x es compuesto si y sólo si tiene más de dos divisores.
   - Un número es divisible por 3 si y sólo si al sumar sus cifras el resultado es múltiplo de 3.
2. Dadas las siguientes premisas, complete de acuerdo a la regla del bicondicional

1	Si $x + 4 = 8$ entonces $x = 4$	P
2	Si $x = 4$ entonces $x + 8 = 8$	P
C		

3. Dadas las siguientes premisas demuestre $p \wedge 2 = y \leftrightarrow y = p$, aplicando entre otras la regla bicondicional.

1	Si $p2 = q \rightarrow q = p$	P
2	Si $p = q \rightarrow q > p$	P
	Si $(p = q \rightarrow q > p) \rightarrow (q = p \rightarrow p^2 = q)$	P

C

3.3.11 Regla de dilema

La regla del Dilema (D) parte de tener tres proposiciones: dos proposiciones condicionales y una proposición con una disyunción. La proposición con la disyunción se compone de los antecedentes de los condicionales. La proposición que se obtiene de la regla del dilema es una disyunción que se compone de los consecuentes de los condicionales. La regla del dilema se puede deducir de la siguiente manera:

1	$p \rightarrow q$	P
2	$s \rightarrow h$	P
3	$p \vee s$	P
C	$q \vee h$	$D_{1,2,3}$

Por ejemplo si se tienen las siguientes premisas: Si Lorena aprueba el curso, entonces gana la beca. Si el estudiante gana la beca, entonces representa a la universidad. A partir de los condicionales se obtiene la disyunción con los antecedentes quedando la premisa: Lorena aprueba el curso o la estudiante gana la beca. A partir de la regla del Dilema, se obtiene la conclusión: La estudiante gana la beca o representa a la Universidad.

Por ejemplo, si se tienen tres premisas, se puede obtener una nueva premisa aplicando la segunda regla de simplificación. De acuerdo a la configuración que se tiene, es posible entonces aplicar la regla del dilema a las premisas.

1	$s \rightarrow (s \wedge t)$	P
2	$q \rightarrow p$	P
3	s	P
C_1, 3	$s \vee q$	IIS_3
C	$p \vee (s \wedge t)$	$D_{1,2,4}$

.:Actividad 3.12.

1. Aplique las reglas de inferencia necesarias a las siguientes premisas, para demostrar (s v q).

1	$p \to \neg(s \vee q)$	P
2	$h \to q$	P
3	$p \to s$	P
4	h	P
C		

2. Dada la siguiente proposición complete las premisas faltantes y su correspondiente conclusión.

1	Si un número entero termina en dos entonces es par	P
2		
3		
C		

3. Dada la siguiente proposición complete las premisas faltantes y su correspondiente conclusión.

1	Si un número termina en cero entonces es divisible por 10	P
2	Si un número termina en cinco entonces es divisible por 5	P
3		P
C		$D_{1,2,3}$

3.3.12 Simplificación disyuntiva

La regla de Simplificación Disyuntiva (SD) establece que de la disyunción entre dos mismas proposiciones, es posible simplificarla en una sola proposición.

1	$p \vee p$	P
C	p	SD_1

Por ejemplo si se tiene la premisa: Java es un lenguaje de programación o Java es un lenguaje de programación, entonces se puede simplificar en una solo proposición: : Java es un lenguaje de programación.

1	Java es un lenguaje de programación o Java es un lenguaje de programación	P
C	Java es un lenguaje de programación	SD_1

Por ejemplo en la siguiente lista de premisas se aplican diferentes reglas de inferencia. Se observa que la premisa de la línea 6 tiene la estructura $r \vee r$, en este caso es posible aplicar la regla de simplificación disyuntiva, obteniendo como simplificación la conclusión n.

1	$\neg(p \wedge q)$	P
2	$\neg q \rightarrow r$	P
3	$\neg p \rightarrow r$	P
4	$m \rightarrow \neg r$	P
C_1, 5	$\neg p \vee \neg q$	M_1
C_2, 6	$r \vee r$	$D_{3,2,5}$
C	r	SD_6

Otra definición de simplificación disyuntiva es la propuesta por (Flores, 2011) "Si disponemos de dos premisas que corresponden a dos implicaciones con el mismo consecuente, y sus antecedentes se corresponden con los dos miembros de una disyunción, podemos concluir con el consecuente de ambas implicaciones". Por ejemplo si se tienen las premisas:

- El vuelo es por aerolínea nacional o extranjera.
- Si vuela por aerolínea nacional, entonces se acumulan millas a la cuenta del viajero
- Si viajas por aerolínea extranjera, entonces se acumulas millas a la cuenta del viajero.
- Entonces, acumulan millas a la cuenta del viajero.

Formalizando las anteriores expresiones, se tiene respectivamente:

- $p \vee q$
- $p \rightarrow r$
- $q \rightarrow r$

De lo cual se puede concluir r.

.:Actividad 3.13.

1. Aplique las reglas de inferencia a las siguientes premisas, para demostrar h

1	$\neg(negp \vee \neg q)$	P
2	$p \rightarrow (h \wedge t)$	P
3	$q \rightarrow (h \wedge t)$	P

C

2. Aplique las reglas de inferencia a las siguientes premisas, para demostrar p.

1	$p \rightarrow \neg q$	P
2	$\neg\neg q$	P
3	$\neg(s \vee p) \rightarrow p$	P

C

3.3.13 Condicional Contrarrecíproca

La regla Condicional Contrareciproca (CC) establece que partir de una expresion condicional, es posible concluir otra expresión la cual tiene como antecedente la negación del consecuente del primer condicional y como consecuente la negación del antecedente del primer condicional.

a)

1	$p \to q$	P
C	$\neg q \to \neg p$	CC_1

O también se puede expresar:

b)

1	$\neg q \to \neg p$	P
C	$p \to q$	CC_1

O también se puede tener:

c)

1	$q \to p$	P
C	$\neg p \to \neg q$	CC_1

Finalmente se puede también expresar.

d)

1	$\neg p \to \neg q$	P
C	$q \to p$	CC_1

De las representaciones a) y b) se puede deducir que: $(p \to q) \leftrightarrow (\neg q \to \neg p)$. De las expresiones c) y d) se puede deducir que: $(q \to p) \to (\neg p \to \neg q)$.

Por ejemplo si se tiene la premisa: Si una persona paga los impuestos, entonces tendrá beneficios tributarios. Corresponde a la estructura:
a)

1	$p \to q$	P
C	$\neg q \to \neg p$	CC_1

Por lo tanto premisa como conclusión quedan de la siguiente manera:

1	Si una persona paga los impuestos,entonces tendrá beneficios tributarios	P
C	Si una persona no tiene beneficios tributarios entonces no paga los impuestos	CC_1

O también

1	Si una persona no tiene beneficios tributarios entonces no paga los impuestos	P
C	Si una persona paga los impuestos,entonces tendrá beneficios tributarios	CC_1

En la siguiente lista de premisas se aplican reglas de inferencia. Se observa que en la premisa de la línea 6 tiene la estructura $p \to (p \wedge q)$, se puede aplicar la regla de condicional contrarrecíproca, obteniendo como conclusión $\neg(p \wedge q) \to \neg p$.

1	$p \to t$	P
2	$t \to \neg s$	P
3	$\neg s \to (p \wedge q)$	P
4	$\neg\neg q$	P
C_1, 5	$p \to \neg s$	$T_{1,2}$
C_2, 6	$p \to (p \wedge q)$	$T_{5,3}$
C	$\neg(p \wedge q) \to \neg p$	CC_6

.:Actividad 3.14.

1. Aplique las reglas de inferencia a las siguientes premisas, para demostrar $\neg p \vee q$.

1	$p \to q$	P
2	$\neg q$	P
C_1, 3	$\neg q \to \neg p)$	CC_1
C_2, 4	$\neg p$	$PP_{2,4}$

C	$\neg p \vee q$

2. Aplique las reglas de inferencia a las siguientes premisas, para demostrar $\neg(r \wedge p) \to \neg r$.

1	$r \to t$	P
2	$t \to \neg\neg\neg s$	P
3	$\neg s \to (r \wedge p)$	P
	P	P

C

CAPÍTULO 4

Lógica cuantificada LC

La imaginación nos lleva a
descubrir o inventar nuevos
universos matemáticos.

Newton da Costa - Lógico
brasileño
1929 –

Los aprendizajes que se espera el estudiante obtenga al finalizar este capítulo de la lógica cuantifica o de predicados son:

- Identificar los elementos fundamentales del cálculo de predicados y cuál es su relación con el cálculo proposicional.
- Entender y aplicar los conceptos: término, predicado.
- Representar los predicados y su lista de términos.
- Establecer equivalencias entre proposiciones que utilizan cuantificadores.
- Conocer y aplicar el concepto de argumento constante y argumento variable.
- Conocer las equivalencias lógicas clásicas dentro del cálculo de predicados.
- Conocer la deducción natural de la lógica cuantificada.

4.1 Introducción

En este capítulo se analizarán los elementos fundamentales de la lógica cuantificada (o de predicados), la cual permite extender la capacidad expresiva y de aplicación del cálculo proposicional. La lógica cuantificada es una generalización de la lógica proposicional, en la cual se introducen nuevos elementos del lenguaje, los cuales permiten estudiar y detallar la estructura interna de los enunciados (propiedades y las relaciones entre los objetos), es decir su estructura sintáctica y semántica.

En la lógica proposicional (LP) la formalización de las expresiones se puede realizar sin que existan relaciones entre los objetos, sin embargo, existen situaciones en las cuales el cálculo proposicional no

es suficiente para representar todas las afirmaciones que se puedan expresar. Por ejemplo, si se tienen expresiones como "$y < y + x$ ", "$y = w \times 2$ ", no se puede afirmar si estas expresiones son verdaderas o falsas. Solo cuando se asocien valores a cada una de las variables w, x, y, se podrá establecer un valor de verdad de verdad o falsedad a la expresión.

La lógica cuantificada, es vista desde ahora como una ampliación de la lógica de enunciados que cuenta un lenguaje formal más rico (más expresivo) y con un conjunto de reglas que permiten validar razonamientos expresados utilizando este lenguaje. La lógica de enunciados (LP) debe entenderse a partir de este momento, como un subconjunto de la lógica de predicados.

La lógica cuantificada permite simbolizar enunciados o razonamientos con los cuales se dificulta su simbolización en LP, un ejemplo de estos casos son los siguientes:

Primera situación. Simbolizar el siguiente argumento en LP. (Cardona, Hernández & Jaramillo, 2010) .

Todos los quindianos son caficultores.
Todos los empleados son quindianos.
Todos los empleados son caficultores.

1	Todos los quindianos son caficultores.
2	Todos los empleados son quindianos.
C	Todos los empleados son caficultores.

Este razonamiento es válido, veamos ahora que pasa con este razonamiento en LP.

Sea:

p : Todos los quindianos son caficultores.
q : Todos los empleados son quindianos.
r : Todos los empleados son caficultores.

La expresión proposicional es: $(p \wedge q) \rightarrow r$. En el caso de hacer verdaderas las premisas y falsa la conclusión, la fórmula resulta ser una fórmula no válida. Adicionalmente se puede ver que la validez de esta inferencia depende tanto del valor de verdad de las proposiciones como de la relación de los componentes de la proposición.

La otra situación es poder simbolizar este tipo de enunciados "Todos los estudiantes de lógica tienen carro o no lo tienen ".

Para poder simbolizar este enunciado en LP, se tendría que asignar una letra proposicional a cada uno de los estudiantes del curso de lógica, por ejemplo:

p : Andrés tiene carro.
q : Jorge tiene carro.
r : Valentina tiene carro.

Asumiendo que Andrés, Jorge y Valentina son estudiantes del curso de lógica, el enunciado tomaría la forma:

$$(p \vee \neg p) \wedge (q \vee \neg q) \wedge (r \vee \neg r) \wedge \cdots$$

Si las posibilidades se hacen infinitas, la conjunción se hace infinita.

Retomando la inferencia:

1	Todos los quindianos son caficultores.
2	Todos los empleados son quindianos.
C	Todos los empleados son caficultores.

Se puede analizar la estructura interna de las proposiciones y se puede escribir de una forma general:

1	Todos los A son B
2	Todos los C son A
C	Todos los C son B

Los elementos A, B y C , son objetos que se pueden elegir libremente. Con base en lo anterior, se hace necesario identificar otras estructuras que permitan un análisis más exhaustivo de las relaciones de las proposiciones atómicas, y es en este punto donde la lógica cuantificada se puede aplicar.

Inicialmente, para el análisis más detallado de las proposiciones atómicas se analizarán algunos conceptos preliminares.

4.2 Semántica en LC

Como se evidenció en la anterior sección, es necesario que a cada una de las proposiciones atómicas, se le concrete a que objetos reales se hace referencia.

4.2.1 Términos

El término es una expresión en la que se hace referencia a un objeto en particular dentro de una proposición. Los objetos pueden ser personas, cosas físicas, conceptos. Por ejemplo si se tienen las proposiciones:

- Mario no se divierte
- Andrés practica deporte
- El automóvil es nuevo
- Felipe tiene el mejor promedio

Los términos para estas proposiciones son respectivamente: José, María, El carro, cinco y cuatro.

El término no siempre debe hacer referencia a un nombre en particular, también puede ser una frase de la que se pueda identificar el objeto particular, por ejemplo:

- El primer mandatario de los países bajos

- La isla con mayor extensión
- El capitán del equipo
- El futbolista mejor pago

La siguiente es una propuesta de clasificación de los términos, hecha por algunos autores como Constantino Malagón Luque Profesor – Investigador Universidad Nebrija[1]

Términos singulares o constante: El objeto referenciado es alguien o algo específico. También puede interpretase como una palabra o frase que designe o se refiera a una cosa individual. Por ejemplo: los nombres propios (Jorge, Andrés, etc) y las descripciones definidas (el inventor del computador, el actual presidente de Colombia).

Términos generales o variables: El objeto referenciado es alguien o algo genérico, generalmente, son términos utilizados para caracterizar una clase general de individuos que poseen una propiedad en cuestión. Por ejemplo: Ingenieros, Pintores, programadores, etc.

4.2.2 Predicado

La lógica de predicados utiliza dentro de sus conceptos más importantes, el concepto de término el cual ya se analizó en la anterior sección y el concepto de predicado. El término representa objetos particulares o específicos. Los predicados denotan o expresan las características, cualidades o atributos de los objetos.

En una descripción más general un predicado es una afirmación que expresa una propiedad de un objeto o una relación entre objetos. Estas afirmaciones se hacen verdaderas o falsas cuando se reemplazan las variables (objetos) por valores específicos. (Tomado de [González, 2013])

Por ejemplo si se tiene la proposición:

Mario Vargas es escritor

En la anterior proposición se tiene que Mario Vargas es el sujeto de la misma, por lo tanto es un término y escritor es el resto de la proposición que dice algo del sujeto y por lo tanto este es el predicado.

Las siguientes expresiones atómicas permitirán identificar tanto su término como su predicado: (Cardona, Hernández & Jaramillo, 2010)

Proposición Atómica	**Término**	**Predicado**
Álvaro es un trabajador	Álvaro	es un trabajador
José es Ingeniero Electrónico	José	es Ingeniero Electrónico
María viaja	María	es viaja
El Presidente vive en Bogotá	El presidente	vive en Bogotá
El Real Madrid es campeón	El Real Madrid	es campeón

[1] Más información en `http://www.nebrija.es/~cmalagon`

En el enunciado "El Real Madrid es campeón", "Campeón" es el predicado y se simboliza Cr, donde C representa el predicado "Campeón" y r denota el témino que en este caso es el Real Madrid.

Las siguientes definiciones son planteadas por Páez en [Páez, 2007]:

4.2.3 Funciones proposicionales

Una función proposicional es una expresión abierta de la forma Px, la cual se convierte en un enunciado, cuando se le asigna un valor específico a la variable x.

Por ejemplo, si se tiene Cx, sería una función proposicional, que indica que x tiene la propiedad C, y si nos referimos al ejemplo anterior, indicaría que x es un campeón. Cuando x es remplazado por la constante r, indica que Real Madrid es campeón.

4.2.4 Grado del predicado

Un predicado es de grado n, si utiliza n variables proposicionales. Tenga en cuenta que dichas variables provienen de un conjunto no vacío al que llamamos Universo del discurso (UD). Por ejemplo, si decimos Sx donde x es un sistema operativo, entonces el UD son unos cuantos elementos (Windows, Unix, Linux, etc). Mientras que si Sx representa x es un sistema solar en la galaxia, el UD seria muy grande.

Si el predicado no tiene ninguna variable proposicional, entonces la expresión ya no pertenece a la lógica cuantificada sino a la de predicados. El grado de un predicado también se conoce como **Aridad**, por ejemplo si un predicado tiene grado uno, se dice que es unaria o 1-aria; si el predicado es de grado dos se le conoce como binaria y así sucesivamente.

Por ejemplo, si tenemos que Hxy, donde H representa x es hermano de y, el grado de H es dos, debido a que usa dos variables proposicionales, también diríamos que es un predicado binario. Mientras que Cx representa x es un cantante, es de grado uno (unario).

4.2.5 Enunciados singulares

Solo contienen un término singular que denota un individuo, un predicado que le atribuye una propiedad al individuo.

Si tenemos que Andrés es ingeniero, donde Ándres es el término singular e ingeniero es el predicado.

4.2.6 Enunciados generales

Son enunciados que no se refieren a un individuo en particular, si no a todo aquel individuo que tenga determinada propiedad.

En la frase: Todos los ingenieros de Sistemas saben programar. Se refiere al conjunto de los ingenieros, no a uno en particular.

Nota: Por lo general, los enunciados generales son antecedidos por palabras como: todos, algunos, alguien, cada, ninguno, términos llamados cuantificadores.

.:Actividad 4.1.

1. Dadas las siguientes proposiciones atómicas, identifique los términos y los predicados. Además, determine si son predicados singulares o generales y el grado de cada proposición.

- El primer partido empezó según lo programado
- La temperatura sube constantemente
- El que entra ahora es el canciller
- El presentador habla rápidamente
- La multitud se dispersó lentamente
- El perro de Juan ladra toda la noche
- Andrés está esperando desde ayer

4.2.7 Representación de predicados – forma 1

Anteriormente se ha mencionado que los predicados hacen referencia a afirmaciones sobre los objetos de acuerdo a una expresión. Dentro de esas expresiones se puede hacer referencia a una determinada lista de objetos (lista de argumentos o términos) asociadas a una serie de características o relaciones entre estos objetos mencionados.

Por ejemplo si se tienen las siguientes expresiones:

Luis es Magister en Informática

La lista de argumentos la compone un solo objeto que para este caso es Luis y su predicado es Magister en Informática.

Armenia y Pereira son capitales de departamento.

La lista de términos o argumentos se compone de Armenia y Pereira y el predicado es son capitales de departamento.

En esta segunda forma de representación, continuaremos teniendo en cuenta únicamente los términos constantes individuales. Para este caso, cada predicado tiene un nombre, el cual va seguido por una lista de argumentos (parámetros), los cuales van entre paréntesis.

Por ejemplo si se tienen las expresiones:

Mariana es una ejecutiva.

Entonces su representación sería:

ejecutiva(Mariana)

Armenia es la capital del departamento del Quindío.

Entonces la representación sería:

capitalDepartamento(Armenia,Quindio)

De la misma forma en la cual en el cálculo proposicional las expresiones tienen un valor de verdad o falso, pasa en el cálculo de predicados, en el cual estos valores de verdad dependen de sus argumentos. Un predicado puede ser cierto para un grupo de argumentos pero falso para otro conjunto. Por ejemplo

si se tiene el predicado:

capital (Colombia, Bogotá)

Se puede afirmar que este tiene un valor verdadero, pero si se tiene el mismo predicado pero con otro argumento, no necesarios son verdaderos como por ejemplo:

capital (Colombia, Barranquilla) ó *capital* (Colombia, Medellín)

Los predicados también pueden ser utilizados para asignar una cualidad abstracta a sus términos, o para representar acciones o relaciones de acción entre dos objetos[2]. Por ejemplo:

duerme (Micaela)
clima (martes, lluvioso)
quiere (Alberto, Liliana)
lee (Marcela, periódico)
visito (Gloria, cárcel)

Al igual que en el capítulo de cálculo proposicional donde se mencionaba que la validez de las expresiones únicamente desde el punto de vista de su estructura, sin tener en cuenta el significado semántico de tales expresiones, en el cálculo de predicados se considera la validez de los predicados en relación con el contexto del mundo real, sin embargo, se sigue trabajando con su significado semántico. Estos predicados que se asumen como lógicamente verdaderos se conocen como axiomas.

4.2.8 Representación de predicados – forma 2

Existen otras formas de simbolizar los predicados, por ejemplo si se tiene la expresión atómica:

Barcelona es campeón.

Entonces C es el predicado "es campeón" e i = Barcelona. Entonces Barcelona es campeón se puede simbolizar:

$$Ci$$

A continuación se muestran otros ejemplos en los cuales se representan mediante la anterior notación:

Proposición Atómica	Representación Predicado
María es una enfermera	Em
Federico es un socio del club	Sf
Juan viaja rápidamente	Vr

.:Actividad 4.2.

1. Dadas las siguientes expresiones, simbolizar de acuerdo a las formas de representación de predicados.

[2] `http://exa.unne.edu.ar/informatica/programacion3/public_html/`. Universidad Nacional del Nordeste Argentina. Departamento de Informática. Curso Programación III (Programación Lógica).

Proposición Atómica	Representación Predicado
Mario es primo de Julián	
El día miércoles el clima es seco	
Rosario está enamorada de Roberto	
El perro tiene cola	
Felipe lee la prensa	
El perro de Juan ladra en la noche	
Andrés duerme desde anoche	
Mariana vive en Brasil	

4.3 Argumentos variables y fórmulas atómicas

Cuando los argumentos de los predicados hacen referencia a un objeto específico, estos se denominan constantes, pero también es posible tener argumentos no identificables o determinados a los cuales se les denominan argumentos variables.

Por ejemplo si se tiene la expresión: "7 es un número primo", se está particularizando un elemento que para este caso es el número 7, y se podría decir la expresión es verdadera o es falsa.

Pero si se tiene una expresión como "x es un número perfecto", se está hablando de un argumento variable, y no se puede afirmar si la expresión es verdadera o es falsa porque x no hace referencia a un objeto particular. Si se sustituye x por algún valor, será posible identificar el valor de verdad, por ejemplo:

- 28 es un número perfecto (verdadero) esnumeroperfecto(28)
- 33 es un número perfecto (falso) esnumeroperfecto(33)

También puede suceder un caso como el siguiente: *moneda*(españa,x), la variable x puede tomar el valor euro, permitiendo afirmar que el predicado es verdadero; o puede tomar el valor peso, dando lugar a un predicado falso.

Normalmente en el cálculo de predicados se utilizan las letras "x", "y", "z" para representar argumentos variables. También se utiliza dentro de este cálculo las letras "a", "b", "c" para representar argumentos constantes.

Las fórmulas atómicas corresponden a predicados únicos, asociados a una lista de términos o argumentos. Este tipo de fórmulas son expresiones que también dentro del cálculo de predicados se pueden combinar con los operadores lógicos, tal como se hacía con las proposiciones, con el objetivo de conformar expresiones más complejas. Los operadores lógicos son los mismos operadores utilizados en el cálculo proposicional.

En el caso de la expresión: Andrés y Bernardo son Odontólogos, se tiene el operador lógico y, entonces mediante la primera forma de representación se tendría:

$Oa \wedge Ob$, donde O representa el predicado Odontólogo y las constantes a y b representan a Andrés y Bernardo respectivamente.

Si se tiene la expresión Alexandra y Beatriz no son Médicas, entonces se representa:

$$\neg Ma \wedge \neg Mb$$

Usando la segunda forma de representación, por ejemplo si se tiene la expresión: Natalia es arquitecta y Natalia tiene planos. Entonces se puede formar:

$arquitecta(\text{Natalia}) \wedge tienePlanos(\text{Natalia})$

Otros ejemplos en los cuales se forman expresiones compuestas son:

Lógica y Algoritmia son asignaturas:

$asignatura(\text{lógica}) \wedge asignatura(\text{Algoritmia})$

El edificio no es alto:

$\neg\ alto(\text{edificio})$

Cada uno de los casos anteriores, muestra elementos contantes en su lista de argumentos, pero es de recordar que no siempre se asocian elementos particulares. Para este caso se pueden tener expresiones con argumentos variables tales como:

$color(x) \wedge precio(x)$
$colombiano(y) \vee chileno(y)$
$nota(x) \rightarrow (x \geq 0.0) \wedge (x \leq 5.0)$
$carro(z) \wedge blanco(z)$

De las anteriores expresiones, para poder particularizar, es posible sustituir un término especifico en cada una de las ocurrencias de x, y y z. Lo cual podría representar la expresión por un objeto en concreto de la siguiente forma respectivamente:

color(campero) $\wedge$ precio(campero)
colombiano(Didier) $\vee$ chileno(Didier)
nota(parcial) $\rightarrow$ (parcial $\geq$ 0.0) $\wedge$ (parcial$\leq$ 5.0)
carro(mari) $\wedge$ blanco(z)

.:Actividad 4.3.

1. Dadas las siguientes expresiones, representarlas con argumentos variables y determine su validez.
   - x no es un numero primo, pero es un numero perfecto
   - y esta a una distancia de 250 kilómetros de x
   - w viaja hacia el norte si y solo si y muestra la ruta
   - si w es mayor que tres y tres es mayor que z, entonces w es mayor que z.

4.4 Tipos de cuantificadores

En las anteriores secciones, se trabajo con argumentos constantes y variables, de los cuales se podría determinar o no su validez de acuerdo a un contexto especifico, esta situación implica una particularización de los objetos, es por ello que se hace necesario generalizar de forma que se pueda afirmar cada cosa de un universo determinado.

Hasta ahora si tuviéramos expresiones como:

- Todos los niños del barrio
- Cualquiera de las personas puede responder
- Algunos animales son peligrosos
- Todos los países tienen producto interno bruto
- Cada persona tiene una madre natural
- Ningún camión paso la prueba mecánica
- Nadie tiene un lapicero de color rojo
- Para ningún ejecutivo el tiempo es suficiente

No fuese posible expresarla con los elementos que se han explicado, pues cada una de ellas denota una frecuencia con la cual es verdadera alguna cosa. Por lo anterior, se hace necesario incluir elementos adicionales que permitan generalizar las expresiones. Inicialmente se analizará el cuantificador Universal.

4.4.1 Cuantificador universal

También llamado cuantificador general, establece que cada objeto del universo del discurso tiene alguna propiedad. Se simboliza con $\forall$

Ejemplo.

Todos los estudiantes de lógica saben programar.

ó

Cada ingeniero de Sistemas egresado de la Uniquindio tiene trabajo.

4.4.1.1 Cuantificador universal afirmativo

Se expresa como "todos los A son B". Una forma equivalente a este enunciado es: para cada x, si x es A, entonces x es B. La forma de simbolizar este tipo de enunciado es:

$$(\forall x)(Ax \rightarrow Bx)$$

Si se desea representar por ejemplo la expresión: "Todas las personas tienen una ilusión". Para este caso se identifica el predicado: "tienen una ilusión", entonces Px significa que x tiene una ilusión. La palabra "todas las personas" indica que esto se aplica para todos los x. Se formaliza:

$$(\forall x)(Ax \rightarrow Px)$$

Donde Ax es x es persona. El predicado anterior también se puede escribir de la siguiente forma:

$$(\forall x)(Ax \rightarrow Px)$$

A continuación, se muestra una seria de formalizaciones de expresiones en las cuales se aplica el cuantificador universal afirmativo.

- Todos los Chilenos comen salmón y juegan futbol

$$(\forall x)(Chileno(x) \rightarrow comesalmon(x) \wedge juegafutbol(x))$$

- Todos los Bogotanos son Colombianos, se puede representar: Para todo x, si x es Bogotano, entonces x es colombiano

  $(\forall x)(Bogotano(x) \rightarrow Colombiano(x))$

- Siempre que el equipo gana o empata, todos quedan felices.

  $(\forall x)(feliz(x) \rightarrow equipogana(x) \vee equipoempata(x))$

También es necesario distinguir cuando al cuantificador universal es negativo, a continuación, se muestra esta situación.

4.4.1.2 Cuantificador universal negativo

Se expresa como "ningún A es B". Una forma equivalente a este enunciado es: para cada x, si x es A, x no es B. La forma de simbolizar este tipo de enunciado es:

$$(\forall x)(Ax \rightarrow \neg Bx)$$

Las frases que comúnmente se usan para denotar el cuantificador Universal negativo son:

- Para ningún x
- Ninguno
- No
- Nadie
- Nada

Si por ejemplo se desea representar la expresión: "ningún empleado público es menor de edad", En este caso "ningún empleado público" hace referencia a todo un universo de todos los empleados y el predicado es "es menor de edad" y por lo tanto se usa para expresar una negación.

La anterior expresión se puede modificar de la siguiente manera:

Para todo y, si y es empleado público, entonces y no es menor de edad.

$$(\forall x)(Ex \rightarrow \neg Mx)$$

También los argumentos pueden estar seguidos de los predicados sin necesidad de paréntesis. Pero para efectos de este libro, utilizaremos los paréntesis.

A continuación, se muestra una seria de formalizaciones de expresiones en las cuales se aplica el cuantificador universal negativo.

- Para todo x, x no es empresario.

  $(\forall x)(\neg empresario(x))$

- Nadie ganó el parcial de lenguaje de programación.

  $(\forall x)(\neg gano(x))$

- Para ningún cantante es importante la fama o el dinero.

$$(\forall x)(cantante(x) \rightarrow (\neg gustafama(x) \vee gustadinero(x)))$$

.:Actividad 4.4.

1. Dadas las siguientes expresiones, representarlas con cuantificadores universales afirmativos o negativos, según sea el caso:
   - Nadie de la familia es profesional
   - Todos en la academia han viajado tanto a Ecuador como a Panamá
   - Ninguno de los visitantes conocía el zoológico.
   - Nada es absolutamente caliente.
   - Todas las cosas se componente de materia orgánica o inorgánica.
   - Todos los niños juegan con Mirus
   - Cualquiera de los estudiantes o puede realizar la pasantía o su trabajo de investigación.
   - Siempre que se viaja o se enferma o se pone de mal humor.

4.4.2 Cuantificador existencial

Este cuantificador establece que un único objeto o algunos objetos del universo del discurso, tienen determinada propiedad (pero aclarando que no todos los objetos tiene la propiedad). Se simboliza $\exists$.

Dentro de este cuantificador hay un caso especial, el cual se utiliza para indicar que un único objeto del universo del discurso cumple cierta propiedad, es el cuantificador universal particular y se simboliza $\exists!$.

Algunas de las frases con la que se identifica generalmente este cuantificador son:

- Existe al menos un x Para algún x Para algunos x Existe un x tal que Algunos x Cuando menos un x

Ejemplo:

Alguien esta mintiendo.

ó

Algún profesor de matemáticas es ingeniero.

Por ejemplo si se quiere representar la expresión: “Existe al menos un mexicano que escribe poemas y es político”, se puede formalizar de la siguiente manera:

$$(\exists x)(Mexicano(x) \rightarrow escribepoema(x) \wedge politico(x))$$

4.4.2.1 Cuantificador particular afirmativo

Se expresa como "algunos A son B" o "algún A es B". Una forma equivalente a este enunciado es: Existe por lo menos un x, tal que x es A y x es B. La forma de simbolizar este tipo de enunciado es:

$$(\exists x)(Ax \wedge Bx)$$

Si se desea representar la expresión: "Algunos estudiantes son deportistas", se podría reestructurar y representar de la siguiente manera:

"Existe por lo menos un x tal que, x es universitario y x es deportista".

$$(\exists x)(Universitario(x) \wedge deportista(x))$$

4.4.2.2 Cuantificador particular negativo

Se expresa como "algún A no es B". Una forma equivalente de escribir este enunciado es: existe por lo menos un x, tal que x es A y x no es B. La forma de simbolizar este tipo de enunciado es:

$$(\exists x)(Ax \wedge \neg Bx)$$

Por ejemplo, la frase "Algún número primo no es impar"

$$(\exists x)(primo(x) \wedge \neg impar(x))$$

Otro ejemplo sería "Algunos profesores no tienen Doctorado". Se puede representar de la siguiente manera:
"Existe por lo menos un x tal que, x es profesor y x no tiene Doctorado".

$$(\exists x)(profesor(x) \wedge \neg doctorado(x))$$

.:Actividad 4.5.

1. Dadas las siguientes expresiones, representarlas con cuantificadores existenciales afirmativos o negativos, según sea el caso:
   - Existe al menos un Ingeniero o un veterinario en la familia.
   - Algunos pájaros cantan en la madrugada y de noche
   - Para algunos campesinos el invierno no es un problema.
   - Cuando menos un Español es hincha del Zaragoza.
   - Algunos niños juegan con el gato, a pesar de ser peligroso.

4.4.3 Equivalencias entre cuantificadores

En las anteriores secciones se mostró que las proposiciones que son cuantificadas (universales o existenciales) pueden ser tanto afirmativas como negativas.

Los cuantificadores existencial y universal son complementarios, es decir, que uno se puede escribir en términos del otro. Por ejemplo, si decimos: "Todos los ingenieros saben Java", esta frase se puede escribir de la forma "No existen ingenieros que no sepan java". Veamos la siguiente tabla:

Proposición	Fórmula equivalente
$(\forall x)Px$	$\neg(\exists x)\neg Px$
$(\forall x)\neg Px$	$\neg(\exists x)Px$
$(\exists x)Px$	$\neg(\forall x)\neg Px$
$(\exists x)\neg Px$	$\neg(\forall x)Px$

Por ejemplo si se tienen las expresiones:

- Todos son hombres equivale a decir es falso que algunos no sean hombres.
- Ninguno es hombre equivale a decir es falso que algunos sean hombres
- Algunos son hombres equivale a decir es falso que ninguno sea hombre
- Algunos nos son hombres equivale a decir es falso que todos sean hombres.

Existen otros tipos de equivalencias llamadas de oposición aristotélica, a continuación se muestran las equivalencias.

Proposición	**Fórmula equivalente**
$(\forall x)(Px \to Qx)$	$\neg(\exists x)(Px \wedge \neg Qx)$
$(\forall x)(Px \to \neg Qx)$	$\neg(\exists x)(Px \wedge Qx)$
$(\exists x)(Px \wedge Qx)$	$\neg(\forall x)(Px \to \neg Qx)$
$(\exists x)(Px \wedge \neg Qx)$	$\neg(\forall x)(Px \to Qx)$

Si se quiere verificar la validez de la equivalencia entre:

$$(\forall x)(Px \to Qx) \text{ y } (\exists x)(Px \wedge \neg Qx)$$

Se pueden aplicar los siguientes pasos:

1. $(\forall x)(Px \to Qx)$
2. $\neg(\exists x)\neg(Px \to Qx)$
3. $\neg(\exists x)\neg(\neg Px \vee Qx)$
4. $\neg(\exists x)(Px \wedge \neg Qx)$

Ambas fórmulas entonces son lógicamente equivalentes y conservan su propiedad de validez.

Otras equivalencias lógicas que se usan en el cálculo de predicados son las siguientes:

Proposición	**Fórmula equivalente**
$(\exists x)\neg\neg(Px)$	$(\forall x)\neg\neg(Px)$
$(\forall x)\neg(Px \wedge Qx)$	$(\forall x)(\neg Px \vee \neg Qx)$
$(\forall x)\neg(Px \vee Qx)$	$(\forall x)(\neg Px \wedge \neg Qx)$

.:Actividad 4.6.

1. Exprese las siguientes expresiones en lenguaje natural, de forma que se apliquen las equivalencias explicadas en esta sección.
   - Ninguno es egresado de odontología
   - Todos son profesionales
   - Algunos tienen cédula de ciudadanía
2. Verifique que las siguientes proposiciones son equivalentes:
   - $(\forall x)(Px \to \neg Qx) \Leftrightarrow \neg(\exists x)(Px \wedge Qx)$
   - $(\exists x)(Px \wedge Qx) \Leftrightarrow \neg(\forall x)(Px \to \neg Qx)$

4.4.4 Alcance del cuantificador

Con los ejemplos que hemos visto, es posible definir inicialmente el alcance de los cuantificadores. Si alguno de los cuantificadores no va seguido de un paréntesis entonces su alcance llega hasta la primera letra que se encuentra a la derecha del predicado, por ejemplo:

- $(\exists x)Px$
- $(\forall x)Px \rightarrow Qx \leftrightarrow Sx$
- $(\exists x)Px \vee Sx$

En las tres situaciones anteriores, el alcance solo llega hasta px.

Si el cuantificador antecede los paréntesis, su alcance abarca a toda la expresión que se encuentra entre los paréntesis, por ejemplo:

- $(\forall x)(Qx \leftrightarrow Sx)$
- $(\exists x)(Px \rightarrow Qx \leftrightarrow Sx)$

4.4.4.1 Variables libres y ligadas

Las variables que están bajo el alcance de algún cuantificador, se les denomina ligadas y en caso contrario se les denomina libres.

Por ejemplo son variables libres:

- Px
- $Px \leftrightarrow Qx \leftrightarrow Sx$
- $Qx \vee Sx$

Son variables ligadas:

- $(\forall x)(\exists y)(Qx \leftrightarrow Sy)$
- $(\exists y)(\forall x)(Py \rightarrow Qx \leftrightarrow Sx)$

4.5 Sintaxis de la lógica de predicados LPRED

En las secciones anteriores ya se han revisado como se representan los predicados, cuáles son sus posibles valores de verdad, los operadores lógicos que se pueden aplicar en este contexto y cómo se relacionan los predicados. Es por ello, que en las siguientes secciones formalizaremos un alfabeto y la sintaxis de esta extensión de la lógica.

La lógica cuantificada cuenta con su propio lenguaje formal, sólo que el alfabeto ahora es una extensión del alfabeto definido en LP.

4.5.1 Alfabeto de LC

Letras proposicionales: Son las mismas de LP
p, q, r, s, t, u, v, w, x, y, z
p_1, p_2, ..., p_n
q_1, q_2, ..., q_n

Predicados:. Letras mayúsculas del alfabeto (opcionalmente con super índice para indicar el grado del predicado).

A, B, C, ...

Términos
Variables: denotadas por las últimas letras del alfabeto x, y, z. Permiten representar cosas, elementos o individuos que no están definidos.

Constantes: Se denotan por las primeras letras del alfabeto a, b, c, ... Estas permiten representar cosas, elementos o individuos que están claramente definidos.

Operadores lógicos.
Unarios: $\neg, \forall, \exists$
Binarios: $\wedge, \vee, \rightarrow$ y $\leftrightarrow$

Símbolos de asociación.
$(,)$

Numerales
Se utilizan para indicar, de manera opcional el grado del predicado.

Valores de verdad.
$v(\cdot) = V$ y $v(\cdot) = F$

Forma atómica en LC.

Es toda expresión de LC que sea o una letra proposicional, o un predicado de LC de grado n, seguido de n términos individuales de LC.

4.6 Reglas de Formación en LC

Toda fórmula de LC, es una fórmula bien formada (fbf) si sólo si puede obtenerse por medio de las siguientes reglas de formación:

1. F_1: Todas las formas atómicas de LC son fbf_s.
2. F_2: Si α es fbf en LC, entonces $\neg(\alpha)$ es fbf.
3. F_3: Si α y β son fbf_s en LC, entonces $(\alpha)\bigstar(\beta)$ es fbf.
4. F_4: Si α es fbf en la que la variable x ocurre al menos una vez, y en la que no hay ningún cuantificador que utilice esa variable, entonces $(\forall x)\alpha$ y $(\exists x)\alpha$ son fbf_s.

4.7 Definición de operador lógico y subformula

1. Si α es una fórmula atómica de LC, no contiene ningún operador lógico y α es la única subfórmula.
2. Si α es una fórmula de LC de la forma $\neg(\alpha')$, entonces la negación es el operador lógico principal de α y α' es la subfórmula inmediata de α .
3. Si α es una fórmula de LC de la forma de la forma $\alpha' \bigstar \alpha''$, el operador lógico $\bigstar$ es el operador lógico principal de α. α' y α'' son las subfórmulas inmediatas de α.
4. Si α es una fórmula de LC de la forma $(\forall x)\alpha'$ o $(\exists x)\alpha'$, entonces el cuantificador que ocurre antes α', es el operador lógico principal de α. α' es la subfórmula inmediata de α.

4.8 Deducción en LC

Para verificar la validez de los argumentos en la lógica cuantificada, se mantienen las reglas de inferencia vistas en La lógica proposicional y se agregan cuatro nuevas reglas.

4.8.1 Instancia de sustitución

Sea α una fórmula de LC de la forma $(\forall x)\alpha'$ o $(\exists x)\alpha'$. Sea x una variable y a una constante. Una instancia de sustitución de α denotada $\alpha(a/x)$, es la fórmula que resulta de eliminar el cuantificador y remplazar todas las instancias de x por la constante a.

Espansión semántica del cuantificador universal Es la conjunción de todas sus instancias de sustitución.

$$\alpha(a_1/x) \wedge \alpha(a_2/x) \wedge \cdots \wedge \alpha(a_n/x)$$

Espansión semántica del cuantificador existencial. Es la disyunción de todas sus instancias de sustitución.

$$\alpha(a_1/x) \vee \alpha(a_2/x) \vee \cdots \vee \alpha(a_n/x)$$

A continuación se daran las cuatro reglas de inferencia adicionales, las cuales operan sobre los cuantificadores.

4.8.2 Eliminación del cuantificador universal (E$\forall$)

Cuando una fórmula esta cuantificada universalmente, la variable cuantificada puede ser sustituida por cualquier término o instancia de sustitución (que este en el universo del discurso) y se elimina el cuantificador.

1.	$(\forall x)Px$	P	
C	Pa		E$\forall_1(a/x)$

Por ejemplo:

Todos los estudiantes de lógica saben programar.
Juan es estudiante de lógica.
Juan sabe programar.

Para simbolizar el argumento en LC se tiene que la guía de simbolización es:
UD : Estudiantes.
Lx : x es estudiante de lógica.
Px : x sabe programar

Así, utilizando la regla se tiene:

1.	$(\forall x)(Lx \to Px)$	P
2.	Lj	P
C,3	$Lj \to Pj$	E$\forall_1$ (j/x)
C	Pj	MP_{23}

4.8.3 Introducción del cuantificador universal (I$\forall$)

Cuando se dispone de una fórmula que contiene una variable libre, esta variable puede ser cuantificada.

1.	Pa	P
C	$(\forall x)Px$	I$\forall_1$

Condiciones para usar esta regla.

i. La variable x debe ser arbitraria, esto quiere decir que cuando se ha deducido Px, donde esta x podría haberse puesto cualquier otro término.

ii. No aparece en el encabezamiento (hipótesis) de la subdedución donde la regla se aplica.

iii. La introducción del cuantificador universal no debe provocar capturas involuntarias de variables libres, esto quiere decir que la variable x, no aparece en la fórmula.

Por ejemplo, si tenemos la siguinte situación:

Todo el mundo es amigo de todo el mundo, por lo tanto todo el mundo es amigo de si mismo.
La guía de simbolización para la formalización del argumento es la siguiente:

UD : Personas.
Axy : x es amigo de y.

Se tendría que:

1.	$(\forall x)(\forall y)Axy$	P
C,2.	$(\forall y)Aby$	E$\forall_1$ (b/x)
C,3.	Abc	E$\forall_2$ (c/y)
C	$(\forall z)Azz$	I$\forall_3$

4.8.4 Eliminicación del cuantificador existencial (E$\exists$)

Cuando una fórmula esta cuantificada existencialmente, la variable cuantificada puede ser sustituida por una constante nueva y el cuantificador puede ser eliminado.

1.	$(\exists x)Px$	P
C	Pa	E$\exists_1(a/x)$

Para poder utilizar esta regla se debe tener en cuenta:

i. Que la constante no ocurra en una deducción que no haya sido desechada.

ii. la constante no ocurre en la fórmula que se desea cuantificar.

Veamos el siguiente ejemplo:

Derive $Ac \wedge Bc$ de $(\exists x)(Bx \wedge Ax)$

1.	$(\exists x)(Bx \wedge Ax)$	P
C,2.	$Bc \wedge Ac$	E$\exists_1(c/x)$
C,3.	Ac	S_2
C,4.	Bc	S_2
C	$Ac \wedge Bc$	A_{34}

4.8.5 Introducción del cuantificador existencial (I$\exists$)

Las constantes de una fórmula pueden sustituirse por una variable cuantificada existencialmente.

1.	Pa	P
C	$(\exists x)Px$	I$\exists_1$

Ejemplo.

Derive $(\exists z)(Rz \wedge Mz)$ de $(\forall x)(Ax \rightarrow Rx)$, $(\forall x)(Bx \rightarrow Mx)$ y $Aa \wedge Ba$.

1.	$(\forall x)(Ax \rightarrow Rx)$	P
2.	$(\forall x)(Bx \rightarrow Mx)$	P
3.	$Aa \wedge Ba.$	P
C,4.	$Aa \rightarrow Ra$	E$\forall_1$ (a/x)
C,5.	$Ba \rightarrow Ma$	E$\forall_2$ (a/x)
C,6	Aa	S_3
C,7	Ra	MP_{46}
C,8	Ba	S_3
C,9	Ma	MP_{58}
C,10	$Ra \wedge Ma$	A_{79}
C	$(\exists z)(Rz \wedge Mz)$	I$\exists_{10}$

Veamos estos otros ejemplos, donde se aplican varias de las reglas anteriores:

Deducir $(\forall y)(\exists x)Bxy$ de $(\exists x)(\forall y)Bxy$

1.	$(\exists x)(\forall y)Bxy$	P
C,2.	$(\forall y)Bay$	E$\exists_1$ (a/x)
C,3.	Bac	E$\forall_2$ (c/y)
C,4.	$(\exists x)Bxc$	I$\exists_3$
C	$(\forall y)(\exists x)Bxy$	I$\forall_4$

Veamos este otro ejemplo:

1.	$(\forall x)(Px \to (Qx \vee Rx))$	P
2.	$(\forall x)\neg Qx$	P
3.	$(\forall x)Px$	P
C,4.	Pa	E$\forall_3$ (a/x)
C,5.	$Pa \to (Qa \vee Ra)$	E$\forall_1$ (a/x)
C,6.	$Qa \vee Ra$	PP_{45}
C,7	$\neg Qa$	E$\forall_2$ (a/x)
C,8.	Ra	TP_{67}
C,9.	$(\forall x)Rx$	I$\forall_8$
C.	$(\forall x)Px \to (\forall x)Rx$	Condicional

Ahora, deduzcamos $(\exists x)Bx$ del conjunto de premisas $\{(\forall x)(Ax \to Bx), Ab\}$

1.	$(\forall x)(Ax \to Bx)$	P
2.	Ab	P
C,3	$Ab \to Bb$	E$\forall_1$ (a/x)
C,4	Bb	PP_{23}
C	$(\exists x)Bx$	I$\exists_4$

Finalmente, $(\forall x)Bx \wedge (\forall x)Tx$ deducir $(\forall x)(Bx \wedge Tx)$

1.	$(\forall x)Bx \wedge (\forall x)Tx$	P
C,2.	$(\forall x)Bx$	S_1
C,3.	$(\forall x)Tx$	S_1
C,4.	Ba	E$\forall_2$ (a/x)
C,5.	Ta	E$\forall_3$ (a/x)
C,6.	$Ba \wedge Ta$	A_{45}
C	$(\forall x)(Bx \wedge Tx)$	I$\forall_6$

.:Actividad 4.7.

1. Traducir los siguientes argumentos a LC y comprobar su validez utilizando reglas de inferencia

   - Si todos los códigos son correctos, todos los ejecutalbes funcionan bien. Sin embargo hay un ejecutable que no funciona bien. por lo tanto no todos los códigos estan correctos.
   - Los estudiantes de lógica saben programar, Santiago es estudiante de lógica. Luego Santiago sabe programar.
   - Todos los números positivos son mayores que cero. 3 es un número positivo. 7 es un número positivo. Por consiguiente 3 y 7 son números mayores que cero.
   - Todos los políticos incumplen alguna promesa. Hay políticos que incumplen todas las promesas. Pero si un político es votado, entonces cumple alguna promesa. Por lo tanto hay políticos que no son votados.

2. Verifique que:

   - $(\forall x)(Ax \wedge Sx)$ se deduce de $(\forall x)(Px \to Sx)$ y $(\forall y)(Py \wedge Ay)$
   - Con $(\forall x)(Ax \to Bx)$ y $(\forall x)(Tx \to Ax)$ deducir $(\forall y)(Ty \to By)$

γ	$\gamma(a)$
$(\forall x)Ax$	Aa
$\neg(\exists x)Ax$	$\neg Aa$

Tabla 4.1: Reglas γ para cuantificadores universales

δ	$\delta(a)$
$(\exists x)Ax$	Aa
$\neg(\forall x)Ax$	$\neg Aa$

Tabla 4.2: Reglas δ para cuantificadores existenciales

4.9 Método del Tableaux

Como se mencionó en la sección 2.9 el método de los tableros semánticos se fundamenta en un procedimiento de decisión, el cual realiza una búsqueda de contraejemplos para determinar si una formula bien formada es satisfacible para un conjunto de literales de la formula. Para el cálculo de predicados se incorporan dos reglas para los cuantificadores universales y existenciales. Estas dos reglas son retomadas de (Ben-Ari, 2012), las reglas se denominan gamma (γ) y delta (δ).
Si una hoja contiene un par complementario de literales, el árbol se etiqueta. Por otro lado, si no contiene un par complementario, se aplican las formulas alfa o beta descritas. Si la formula está cuantificada se aplica las formulas gamma o beta de acuerdo al cuantificador principal.

Con los elementos conceptuales de este capítulo, se considera que es posible que el estudiante pueda trabajar con los temas fundamentales de la programación lógica.

CAPÍTULO 5

Programación lógica en Prolog

Regla del noventa-noventa:
El primer 90 % del código corresponde al primer 90 % del tiempo de desarrollo. El 10 % restante corresponde al otro 90 % del desarrollo.

Tom Cargill - Laboratorios Bell.

Al finalizar el estudio de éste capítulo el estudiante estará en condiciones de:

- Entender los principios teóricos en los cuales se fundamenta la programación lógica.
- Comprender y aplicar los elementos sintácticos fundamentales de la programación lógica: secciones, hechos y reglas, con aplicación al lenguaje de programación Prolog.
- Aplicar el principio de unificación en el lenguaje de programación Prolog.
- Solucionar problemas mediante la recursividad, aplicando los principios de la programación lógica.
- Especificar un sistema experto mediante sus elementos sintácticos fundamentales.

5.1 Introducción

Tradicionalmente la formación de los estudiantes en Ingeniería de Sistemas y profesiones afines, se orienta hacia el aprendizaje de lenguajes de programación de carácter comercial. Estos lenguajes de programación trabajan bajo una concepción orientada a aspectos procedimentales e imperativos, los cuales están direccionados a "como" obtener resultados mediante un conjunto de sentencias dadas como órdenes. Sin embargo, existen otros paradigmas de programación como el lógico y el declarativo, que tienen una amplia aceptación en el contexto académico.

Prolog es un lenguaje de programación que trabaja bajo la concepción del paradigma declarativo. Su nombre tiene origen en la oración en Francés "PROgrammation en LOGique". Este lenguaje de programación fue concebido por los investigadores Alain Colmerauer y Philippe Roussel, en la década

de los 70 en los laboratorios de la Universidad del Marseilles, Francia.

El lenguaje de programación Porolog es muy utilizado en el área de la inteligencia artificial, sin embargo, dado su potencial se aplica a nivel académico en el contexto de los sistemas basados en conocimiento y en general de la algoritmia. Este lenguaje trabaja bajo el paradigma declarativo en el cual se especifica que se desea obtener y no cómo llegar a la solución, es decir, el lenguaje define la manera en la cual solucionará el problema. Formalmente, Prolog va buscando en la memoria de trabajo, hechos que se unifiquen con el objetivo (Cubero & Berzal, 2014).

Se esté capítulo se muestra la relación existente entre el cálculo de predicados y el lenguaje de programación Prolog. Se retoma para el ello el concepto de predicado, proposición y expresión, los cuales a partir de la aplicación de un conjunto de reglas buscan llegar a la solución de un problema.

5.2 Primeros pasos en Prolog

Prolog se ha convertido en el lenguaje ideal para los desarrolladores de inteligencia artificial, así como para los muchos otros propósitos dentro de las ciencias de la computación. Por lo anterior, es importante tener a disposición una versión gratuita de este lenguaje para cualquier tipo de plataforma (Windows, Linux y Mac)[1]. En nuestro caso, hemos optado por la versión SWI-Prolog, la cual, se ofrece en forma gratuita en la dirección:

`http://www.swi-prolog.org`

Para otras plataformas como Androit en Tables o dispositivos móviles, es recomendable usar la versión online, la cual, puede ser encontrada en la dirección:

`http://swish.swi-prolog.org/`

En las tres plataformas, la forma de la instalación es muy simple. En Windows, se descarga el archivo *swipl-w32-721.exe* y se instala siguiendo las recomendaciones del programa instalador. (ver figura 5.1)

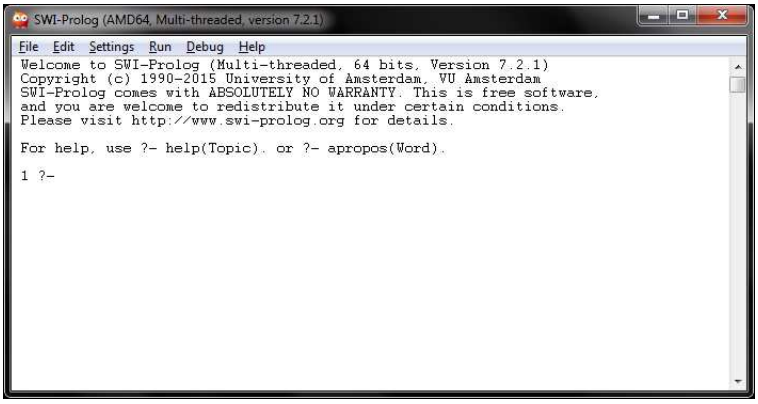


Figura 5.1: Prolog en Ms-Windows

De igual manera para Mac, el archivo se llama *SWI-Prolog-7.2.1.dmg*, el cual contiene a Prolog, este a su vez debe ser copiado a la carpeta Aplicaciones. (ver figura 5.2)

[1]Windows es una marca registrada por Microsoft Corporation. Mac-OS es un amarca registrada por Apple Inc. GNU/Linux es un software bajo distribución de licencia GNU.

Figura 5.2: Prolog en Mac-OS

Para Linux, se debe tener en cuenta la distribución:

Para Debian y Ubuntu (y sus derivados) se escribe en una terminal lo siguiente:

```
% sudo apt-add-repository ppa:swi-prolog/stable
% sudo apt-get update
% sudo apt-get install swi-prolog
```

Una vez terminada la instalación, se abre una terminal y escribimos prolog, luego se da enter. En la misma terminal aparecerá el Prolog cargado y listo para funcionar: (ver figura 5.3)

Figura 5.3: Prolog en Linux-Ubuntu

Para Instalarlo en los derivados de Redhat como Fedora y CentOS, se debe tener en cuenta las instrucciones de compilación, que se muestran en la página de Swi-Prolog:

`http://www.swi-prolog.org/build/Redhat.html`

Para OpenSuse y sus derivados, se aconseja seguir las instrucciones de la página web:

`https://build.opensuse.org/package/show?package=swipl&project=openSUSE%3A11.4`

5.2.1 Fundamentos de Prolog

El lenguaje de programación Prolog usa Lógica de Predicados de Primer Orden (restringida a cláusulas de Horn) para representar datos y conocimiento, utiliza encadenamiento hacia atrás y una estrategia de control retroactiva sin información heurística (backtracking) (Cubero & Berzal, 2014). Una cláusula de Horn, es una disyunción de literales en la cual existe a lo sumo un literal positivo, es decir, un literal que no está negado.

Prolog soporta el uso de caracteres alfanuméricos (letras y números). También permite el uso de caracteres con los cuales se denotan los operadores aritméticos (- , /, *, +), los caracteres relacionales (<,>,=) y caracteres lógicos como &. Prolog tiene un conjunto de palabras clave que restringen el uso de los predicados y las constantes. Estas palabras clave se pueden encontrar en la documentación del sitio web `http://www.swi-prolog.org`.

Las variables en Prolog son cadenas de caracteres que deben estar inicializadas en letra mayúscula. Las variables deben comenzar por mayúscula. Si un identificador comienza por minúsculas se toma como una constante de tipo symbol, la cual tiene una funcionalidad similar a la de una cadena. Por ejemplo: Cuenta es una variable, pero cuenta se considera como una constante de tipo symbol. Por lo general Prolog no diferencia entre mayúsculas y minúsculas. Por ejemplo: hermana, Hermana y HERMANA representan el mismo predicado.

Los comentarios en Prolog son considerados como el texto que se encuentre delimitado por los símbolos de apertura (/*) y por los de cierre (*/.) por ejemplo:

```
/* Esta es una operación de comparación entre las edades */
```

Los comentarios de línea son aquellos que aparecen después del signo de porcentaje (%). Por ejemplo:

```
papa (''Alvaro'', ''Jose'') . % Alvaro es papa de Jose.
```

5.2.2 Metodología básica

La metodología del lenguaje Prolog, se basa en dos cosas:

1. **La Base de datos de conocimiento**: Esta base de datos, es sólo una lista de hechos y relaciones, las cuales "guardan" por así decirlo, todos los hechos y relaciones entre los objetos almacenados.
2. **Las consultas:** Una vez creada la base de datos del conocimiento, se le hacen las preguntas pertinentes a Prolog. Este tratará de hacer las inferencias necesarias con los datos que tiene y retornará una respuesta.

Por ejemplo: la siguiente es una base de datos:

Programa 5.1: Ejemplo en Prolog

```
1  animal(perro).
2  animal(gato).
3  animal(oso).
4  animal(leon).
5
6  animal(gnu).
7  animal(oveja).
8  animal(gallina).
9
10 come(perro, gallina).
11 come(gato,gallina).
```

```
12 come(oso,oveja).
13 come(leon, oveja).
14 come(leon,perro).
15 come(oso, leon).
```

Como se puede apreciar, se tiene se implementan dos predicados:

1. `animal(x)`: Este predicado, sólo dice que x es un animal. En este caso tenemos siete animales, perro, gato, oso, león, Ñu, oveja y gallina.
2. `come(x,y)` Este predicado, relaciona que el animal x se come al animal y. Entonces, tenemos que el perro se come la gallina, el gato a la gallina, el oso a la oveja, el león a la oveja, el león al perro y el oso al león.

Una vez compilado el ejemplo, se puede empezar a realizar la preguntas a Prolog.

```
?- animal(X).
X = perro .
```

En este caso, Prolog, Asigna a la variable X el valor del primer animal almacenado en la base de datos.

También podemos preguntar por un animal en específico, por ejemplo:

```
 ?- animal(leon).
true.
```

En este caso, se indago si el León es una animal, a lo que Prolog contesta con `true`, lo que indica que León si es un animal descrito en la base de datos.

De igual forma, podemos hacer consultas como la siguiente:

```
?- come(gato, gallina).
true.
```

Lo que preguntamos, es que si el gato se come a la gallina, Prolog infiere que dicha afirmación es cierta. Sin embargo, también podemos preguntar:

```
 ?- come(X,leon).
X = oso.
```

Aquí se le pregunta al programa por un animal X que se come al león, lo que infiere que es el oso. De igual forma:

```
?- come(leon,Y).
Y = oveja.
```

Donde se le pide a Prolog que indague por qué animal es comido por el león; donde, se infiere que es la oveja. Así mismo si preguntamos por quien se come al Ñu:

```
 ?- come(X,ngu).
false.
```

Como en la base de datos, no se hace esta definición, Prolog no puede deducirla, por esta razón retorna false.

En las siguientes secciones, se mostrarán ejemplos más elaborados que permitirán al estudiante profundizar en el tema.

5.3 El Lenguaje Prolog

Para hacer una primera aproximación al lenguaje Prolog se presenta la estructura general que puede tener un programa. Se presentan tres secciones Predicates (correspondiente a los predicados), Clauses (correspondiente a las clausulas) y goal (correspondiente a las metas). En las secciones siguientes se presenta una explicación detallada de cada una de las secciones.

El primer ejemplo que se presenta corresponde a un problema orientado a identificar los cuñados de una determinada persona. Se presentan las tres secciones: predicados, cláusulas y metas. Este ejemplo se retoma y adapta de los propuesto en (Cardona, Hernández & Jaramillo, 2010)

Programa 5.2: Primer ejemplo de Prolog

```
1  PREDICATES
2       nondeterm hermana(STRING,STRING)
3       nondeterm hermano(STRING,STRING)
4       nondeterm casado(STRING,STRING)
5       nondeterm cunado(STRING,STRING)
6  CLAUSES
7     casado("Mario","Diana").
8     casado("Juan","Julia").
9     hermano("Juan","Mario").
10    hermana("Julia","Clara").
11    hermana("Julia","Gloria").
12       cunado(A,B):-
13        casado(A,C),
14        hermana(C,B).
15 cunado(A,B):-
16   hermano(A,C),
17   casado(C,B).
18 GOAL
19   cunado("Juan",Z).
```

A continuación, se explican en detalle cada uno de estas secciones.

5.3.1 Predicados (Predicates)

En la sección `predicates` van todas las formas generales de los predicados. Por ejemplo las líneas 2 a 5, se indica que hermana va a ser un predicado que operará sobre dos cadenas de caracteres. La cláusula `nondeterm` indica que el predicado se puede satisfacer varias veces, es decir, que pueden haber muchos pares de personas que pueden ser hermanos.

La misma situación aplica para los predicados `hermano`, `casado` y `cunado`, los cuales trabajaran con argumentos de tipo cadena de caracteres.

Programa 5.3: Predicates

```
1 PREDICATES
2   nondeterm hermana(STRING,STRING)
3   nondeterm hermano(STRING,STRING)
4   nondeterm casado (STRING,STRING)
5   nondeterm cunado (STRING,STRING)
```

La cláusula `nondeterm` indica que el predicado puede satisfacer muchas clases de posibles objetos. Tambien es posible utilizar la cláusula `determ`, que determina que solo puede existir un solo hecho particular en ese momento. Cuando se hace un uso no adecuado de los predicados `determ` o `nondeterm`, el compilador de Prolog produce un mensaje de error indicando la situación.

5.3.2 Hechos (Clauses)

La sección `Clauses` corresponde a todos los hechos que pueden ser considerados como las afirmaciones del programa. En Prolog en esta sección se especifican van los hechos y las reglas del programa. Debe existir una relación entre los predicados y los términos definidos.

Veamos con detalle las diferentes secciones del programa:

Programa 5.4: Clauses y Hechos

```
1  CLAUSES
2    casado("Mario","Diana").
3    casado("Juan","Julia").
4    hermano("Juan","Mario").
5    hermana("Julia","Clara").
6    hermana("Julia","Gloria").
7  cunado(A,B):-
8    casado(A,C),
9    hermana(C,B).
10 cunado(A,B):-
11   hermano(A,C),
12   casado(C,B).
```

La sentencia: `casado(Mario,Diana).`

Tiene como significado que `Mario está casado con Diana`. Aunque, por la sintaxis, casado parece ser una función, en realidad es un predicado. `casado` corresponde a la parte `...está casado con...` de la interpretación en lenguaje natural. Es importante no confundir el concepto de predicado en Prolog con el de función en un lenguaje procedimental.

La sentencia: `cunado(A,B):- casado(A,C), hermana(C,B)`

Es una regla y se interpreta: `A es cuñado de B si A esta casado con C y C es hermana de B`. De nuevo no se puede confundir el concepto de función con el de predicado; `cunado`, `casado` y `hermana` son predicados.

Las reglas se componen de cabeza y cuerpo. Son similares a las expresiones si ... entonces... de los lenguajes de programación procedimentales.

En la sentencia: `cunado(A,B):- casado(A,C), hermana(C,B),`

La cabeza es: `cunado(A,B) y el cuerpo es: casado(A,C), hermana(C,B).`

La cabeza se interpreta como una meta a alcanzar y el cuerpo como submetas. Una forma de alcanzar la meta `A es cuñado de B` es alcanzar primero las metas `A está casado con C` y `C es hermana de B`.

Las variables deben comenzar por mayúscula. Si un identificador comienza por minúsculas se toma como una constante de tipo *symbol*, la cual tiene una funcionalidad similar a la de una cadena. Ejemplo:

Juan es una variable, pero *juan* es una constante de tipo *symbol*. Por lo general Prolog no diferencia entre mayúsculas y minúsculas. Por ejemplo: `hermana`, `Hermana` y `HERMANA` representan el mismo predicado. Una excepción a esta regla es el caso mencionado en el párrafo anterior.

5.3.3 Objetivo (Goal)

En la sección `Goal` se especifica la meta que se pretende alcanzar o el objetivo que se quiere verificar. Cuando la meta tiene variables, Prolog intentará hallar todos los valores de la variable que hagan que se cumpla el predicado.

Programa 5.5: GOAL

```
1 GOAL
2   cunado("Juan",Z).
```

La meta, en este programa, significa: Hallar todos los valores de Z tales que Juan sea cuñado de Z. Es decir, hallar las personas de las que Juan es cuñado.

5.4 Resolución y Unificación

La resolución en el cálculo de predicados es una generalización del algoritmo de resolución proposicional. Sin embargo, existe un paso previo denominado unificación de cláusulas basado en una serie de substituciones. Una substitución es un mecanismo que permite transformar formulas. Su importancia radica en su aplicación a la unificación de un conjunto de expresiones, haciendo que las expresiones resulten sintácticamente idénticas (Labra & Fernández, 1998).

Prolog utiliza un procedimiento llamado unificación para llevar a cabo el proceso de inferencia. El proceso de unificación busca que dos expresiones coincidan. El proceso se realiza mediante el reemplazo de variables por constantes o también por otras variables. Por ejemplo si se desean unificar las siguientes expresiones:

Por ejemplo si se desean unificar las siguientes expresiones:

```
cuñado(josé,Z)
cuñado (A,B),
```

Una posible unificación podría ser:

```
cuñado(josé,Z)
cuñado(A,B) → cuñado(josé,Z)
A/josé
B/Z
```

En la unificación, la primera expresión `cuñado(josé,Z)`, queda expresada de la misma forma. En la segunda expresión, se enlaza A con josé y B con Z. Se evidencia que las dos expresiones resultantes son idénticas.

.:Actividad 5.1.

1. Unifique las siguientes expresiones.

Programa 5.6: Ejercicios unificación de expresiones I

```
1 padre(A,B)
2 padre(X,luis)
3
4 cunado(jose,Z)
5 cunado(luis,B)
6
7 padre(A,B)
8 cunado(jose,luis)
```

2. Señale con cuales de las siguientes expresiones se puede unificar la siguiente expresión:

Programa 5.7: Ejercicios unificación de expresiones II

```
1 recomendar(X, movistar)
2 a) recomendar (juan, movistar)
3 b) recomendar (maria, comcel)
4 c) recomendar (A, B)
5 d) convenir (juan, movistar)
6 e) recomendar (juan, Y)
7 f) recomendar (movistar)
```

5.5 Motor de Inferencia

Cuando se realiza la ejecución del programa, Prolog realiza una búsqueda conocida como razonamiento hacia atrás, el cual es un procedimiento que realiza una búsqueda en la cual hace la selección de un conjunto de reglas. Retomando el caso de la sección anterior, se plantea como meta identificar los valores de `z` que satisfacen:

`cunado(Juan,z).`

Para representar la búsqueda que realiza Prolog, se hará uso de una estructura de árbol, en la cual su raíz es el objetivo que se pretende.

`cunado(Juan,Z).`

Lo primero que intenta Prolog es unificar el objetivo con una cabeza de regla o con un hecho. Unificar dos sentencias es simplemente hacerlas iguales, mediante sustitución de variables. Prolog hará las sustituciones de variable que sean necesarias para lograr que las dos sentencias sean iguales.

Inicialmente Prolog intenta unificar el objetivo con una cabeza de regla o con un hecho. Unificar dos sentencias es hacerlas iguales, mediante un proceso de sustitución de variables. Prolog hará las sustituciones de variable que sean necesarias para lograr que las dos sentencias sean iguales.

Observemos el programa:

Programa 5.8: Unificación de la meta con la primera regla

```
1 CLAUSES
2     casado("Mario","Diana").
3     casado("Juan","Julia").
4     hermano("Juan","Mario").
5     hermana("Julia","Clara").
6     hermana("Julia","Gloria").
7     cunado(A,B):-
```

```
8           casado(A,C),
9           hermana(C,B).
10      cunado(A,B):-
11          hermano(A,C),
12          casado(C,B).
13  GOAL
14      cunado("Juan",Z).
```

El objetivo unificó con la cabeza de la primera regla. Para hacer la meta y la cabeza de la regla iguales basta reemplazar A por "Juan" y B por Z (ver líneas 7 y 14).

Se realizan las sustituciones tanto en la cabeza como en el cuerpo de la regla.

La regla:

`cunado( A, B):- casado(A,C), hermana(C,B).`

Se transforma en: `cunado(Juan,Z):- casado(Juan,C), hermana(C,Z).`

Para la unificación en el árbol se adiciona un nodo hijo. En el nodo hijo aparece el cuerpo de la regla como quedo después de las sustituciones (ver figura abajo). En la gráfica también aparecen las sustituciones realizadas (ver figura 5.4).

En la gráfica también incluimos las sustituciones realizadas, `A = Juan, B = Z.`

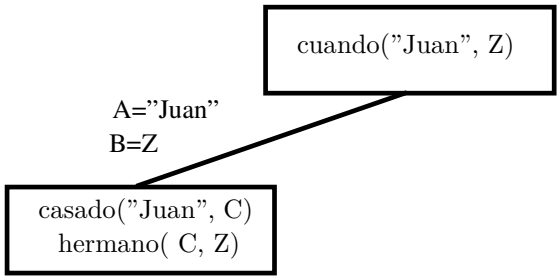


Figura 5.4

La figura 5.4 ilustra que, para encontrar cuñados de Juan, Prolog intentará:

1. Encontrar alguna persona C, tal que Juan está casado con C. Es decir, hallar la esposa de Juan.
2. Encontrar personas Z, tales que C sea hermana de Z. Es decir, hallar las hermanas de la esposa de Juan.

Unificación de la meta con la segunda regla

A continuación Prolog busca la unificación con la segunda regla:

Programa 5.9: Unificación de la meta con la primera regla

```
1 CLAUSES
2     casado("Mario","Diana").
3     casado("Juan","Julia").
4     hermano("Juan","Mario").
5     hermana("Julia","Clara").
```

```
6       hermana("Julia","Gloria").
7       cunado(A,B):-
8           casado(A,C),
9           hermana(C,B).
10      cunado(A,B):-
11          hermano(A,C),
12          casado(C,B).
13  GOAL
14      cunado("Juan",Z).
```

La meta también se puede unificar con la cabeza de la segunda regla. Basta sustituir A por `Juan` y B por Z para que la meta y la cabeza de la segunda regla queden iguales. Agreguemos un segundo hijo a la raíz del árbol en forma similar a como se agregó el primero:

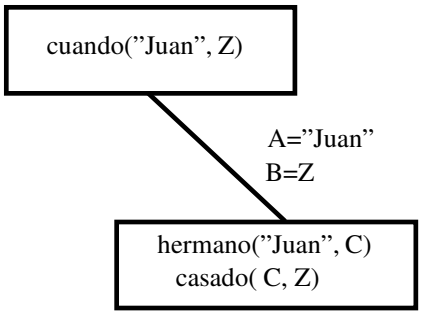


Figura 5.5

El árbol completo queda:

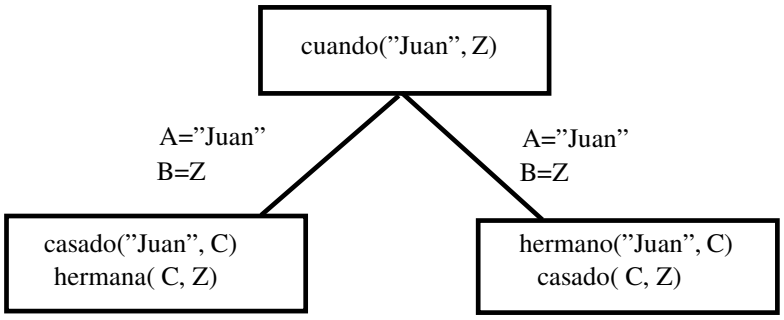


Figura 5.6

Posteriormente Prolog realiza la unificación de una submeta.

Para continuar expandiendo el árbol, se considera cada una de las sentencias de los nodos hijos como submetas. Se tienen cuatro submetas en el flujo del programa. Se aplica entonces el procedimiento de unificación a cada uno de ellas. Se presentan la sección clauses y las submetas del primer nodo hijo:

Programa 5.10: Unificación de una submeta

```
1  CLAUSES
2       casado("Mario","Diana").
3       casado("Juan","Julia").
4       hermano("Juan","Mario").
```

```
5     hermana("Julia","Clara").
6     hermana("Julia","Gloria").
7     cunado(A,B):-
8         casado(A,C),
9         hermana(C,B).
10    cunado(A,B):-
11        hermano(A,C),
12        casado(C,B).
13
14 Submetas del primer nodo hijo:
15        casado("Juan", C)
16        hermana(C,Z)
```

La primera submeta se puede unificar con el segundo de los hechos (ver código en negrilla). Basta reemplazar C por Julia para que la primera submeta y el segundo hecho queden iguales. El reemplazo también se hace también en la segunda submeta quedando esta de la siguiente manera:

```
hermana( Julia ,Z)
```

Se agrega un nuevo nodo al árbol. En el nuevo nodo va la segunda submeta como quedó después de los reemplazos (ver figura 5.7). Como la unificación no fue con una regla, no hay cuerpo de regla para escribir en el nuevo nodo. La sección de árbol queda:

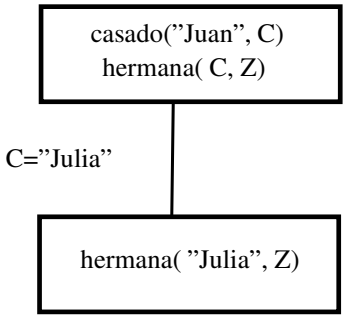


Figura 5.7

La figura 5.7 ilustra que Prolog encontró que la esposa de Juan es Julia y que lo siguiente que hará será buscar las personas Z tales que Julia sea hermana de Z. En pocas palabras, Prolog buscará las hermanas de Julia.

El árbol completo se configura de la siguiente manera:(figura 5.8)

Se presenta entonces la meta que se unificar.

Veamos de nuevo el código y la meta que se debe unificar en este momento:

Programa 5.11: Una unificación que produce un nodo vacío

```
1 CLAUSES
2     casado("Mario","Diana").
3     casado("Juan","Julia").
4     hermano("Juan","Mario").
5     hermana("Julia","Clara").
6     hermana("Julia","Gloria").
7     cunado(A,B):-
```

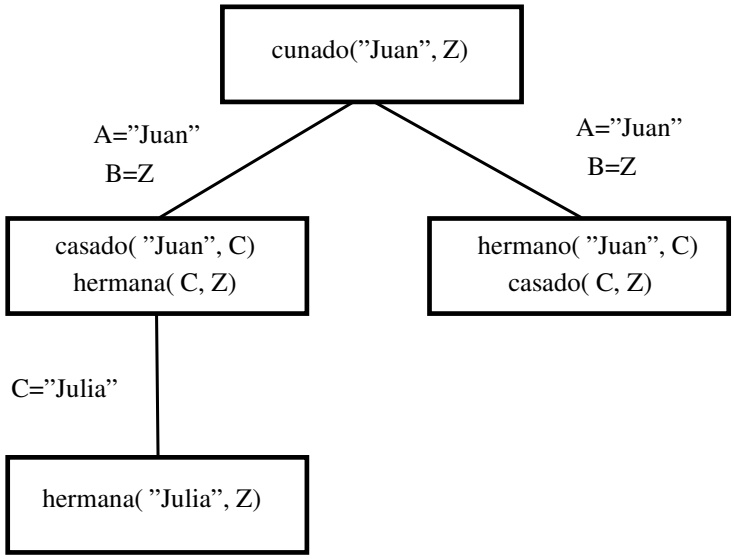


Figura 5.8

```
 8          casado(A,C),
 9          hermana(C,B).
10
11      cunado(A,B):-
12          hermano(A,C),
13          casado(C,B).
14
15  Submeta:
16          hermana ("Julia", Z)
```

La meta se puede unificar con el cuarto hecho (ver líneas 5 y 16). Basta sustituir Z por Clara . No hay cuerpo de regla ni subobjetivos adicionales para escribir en el nuevo nodo. Este queda vacío. Gráficamente: (figura 5.9)

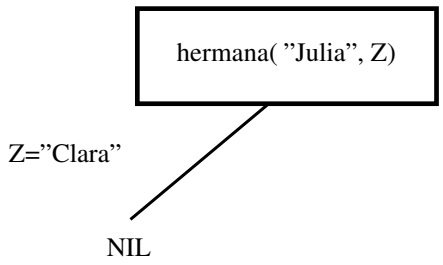


Figura 5.9

El árbol con base en la unificación que produce un nodo vacío, queda configurado de la siguiente manera: (figura 5.10)

Ahora seguimos expandiendo el árbol hasta que todas las hojas sean `NIL` o sean nodos cuyo primer objetivo no se pueda unificar con alguna cabeza de regla o hecho. En este primer ejemplo todos los subobjetivos se pueden unificar, por lo que todas las hojas quedan iguales a `NIL`. (figura 5.11)

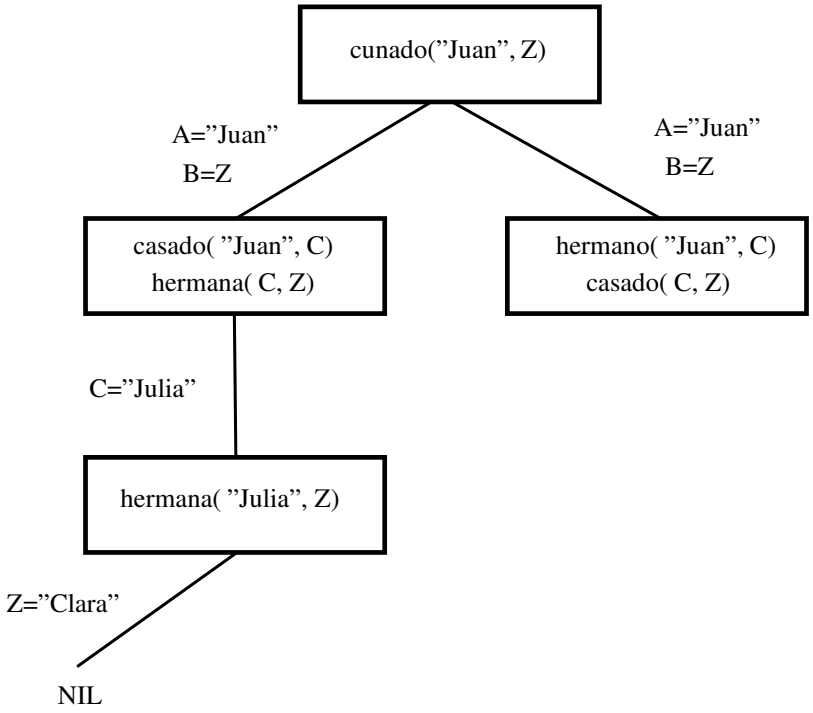


Figura 5.10

Para obtener la salida del programa anotamos los valores asignados a Z que conduzcan a un nodo hoja vacío. Los valores se presentan en orden de izquierda a derecha como aparecen en el árbol que se representó anteriormente y para que la salida quede en el mismo orden en que la produciría Prolog:

```
Z=Clara
Z=Gloria
Z=Diana
3 solutions
```

De esta manera Prolog encuentra las tres cuñadas de Juan con base en los hechos codificados en el programa.

A continuación se muestra otro ejemplo completo que utiliza razonamiento hacía atrás. Este programa halla los ancestros de una persona.

Programa 5.12: Programa que utiliza razonamiento hacía atrás

```
1  PREDICATES
2      nondeterm padre(symbol,symbol)
3      nondeterm ancestro(symbol,symbol)
4  CLAUSES
5      padre(Juan,pedro).
6      padre(pedro,tibe).
7      padre(tibe,camilo).
8      padre(rober,maria).
9      ancestro(A,B):-
10         padre(A,B).
11     ancestro(A,B):-
12         padre(C,B),
```

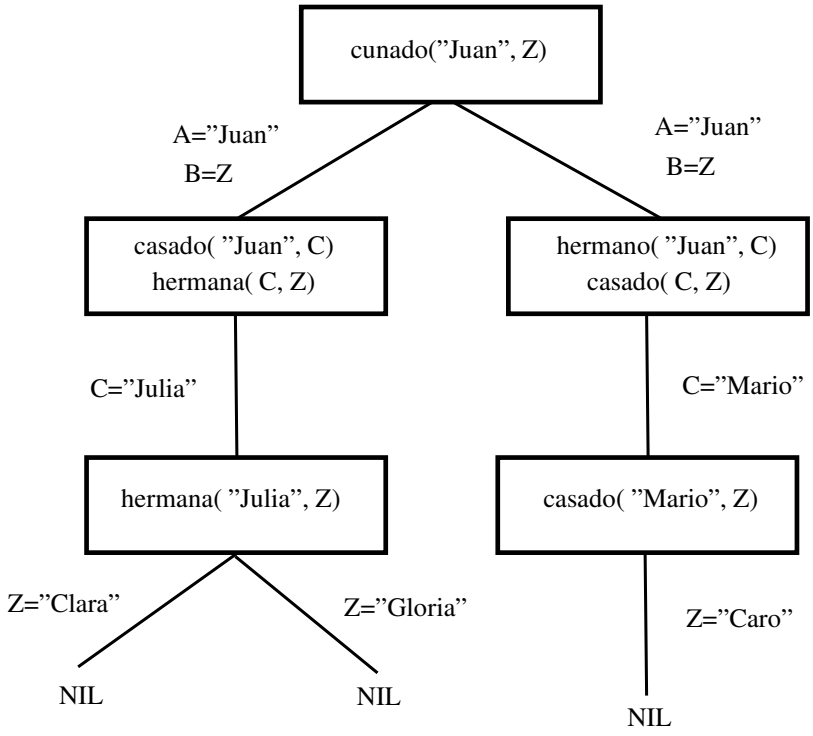


Figura 5.11

```
13          ancestro(A,C).
14 GOAL
15      ancestro(X,tibe).
```

La meta del programa 5.12, consiste en hallar todos los ancestros de `tibe`.

Se presenta la sección predicates:

```
PREDICATES
   nondeterm padre(symbol,symbol)
   nondeterm ancestro(symbol,symbol)
```

El tipo `symbol` es similar a una cadena de caracteres pero que no va encerrada entre comillas.

Si un identificador empieza por minúscula, Prolog entenderá que es una constante de tipo `symbol` y no una variable. Por ejemplo `tibe` es una constante de tipo symbol, pero `Tibe` es una variable.

Lo anterior evidencia que se tiene una rigurosidad en la sintaxis de este lenguaje de programación y debe ser tenida en cuenta al momento de construir un programa.

En la regla: `ancestro(A,B):- padre(C,B), ancestro(A,C).`

Podemos observar recursión. Prolog no dispone de bucles ni de sentencias de decisión, en su lugar utiliza reglas recursivas. En particular esta regla significa que una forma de llegar a que A es un ancestro de B es demostrando que algún C es padre de B y que A es ancestro de C.

La figura 5.12 ilustra el árbol de búsqueda correspondiente al programa que halla los ancestros. Siguen explicaciones de algunas de las unificaciones representadas en el árbol.

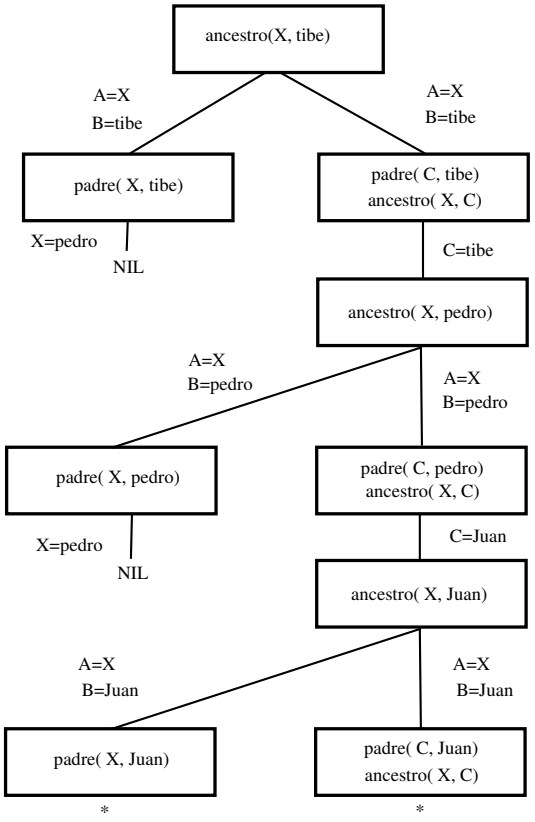


Figura 5.12

La salida del programa es:

```
X=pedro
X=Juan
2 Solutions
```

La meta se unifica con las dos reglas. Los cuerpos de las reglas, transformados, van en los nuevos nodos. Observe como se unifica la meta con las dos reglas. Los cuerpos de las reglas, transformados, van en los nuevos nodos. (figura 5.13)

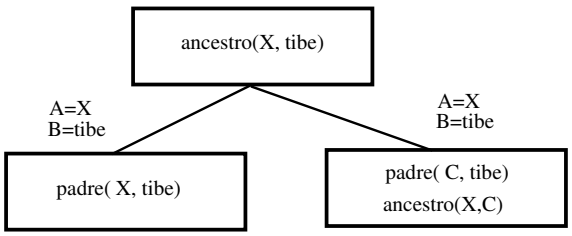


Figura 5.13

```
CLAUSES
    padre(Juan,pedro).
    padre(pedro,tibe).
    padre(tibe,camilo).
    padre(rober,maria).
    ancestro(A,B):-
        padre(A,B).
    ancestro(A,B):-
        padre(C,B),
        ancestro(A,C).
GOAL
    ancestro(X,tibe).
```

Subobjetivos del nodo que se debe expandir:
`padre(X,tibe)`

Se unifica el primer subobjetivo de un nodo con un hecho. El segundo subobjetivo, transformado, va en el nuevo nodo.

Observe como se unifica el primer subobjetivo de un nodo con un hecho. El segundo subobjetivo, transformado, va en el nuevo nodo. (figura 5.14)

```
CLAUSES
    padre(Juan,pedro).
    padre(pedro,tibe).
    padre(tibe,camilo).
    padre(rober,maria).
    ancestro(A,B):-
        padre(A,B).
    ancestro(A,B):-
        padre(C,B),
        ancestro(A,C).
Subobjetivos del nodo que se debe expandir:
     padre(C,tibe).
    ancestro (X,C)
```

Subobjetivos del nodo que se debe expandir:

```
padre(C,tibe).
ancestro (X,C)
```

Un subobjetivo que no se puede unificar

Por primera vez se tiene el caso en que el primer subobjetivo de un nodo no se puede unificar con ningún hecho ni con ninguna regla. En este caso el nodo "se muere". Un asterisco indica que nodo no se expandirá ya más. No interesa que el segundo subobjetivo si se pueda unificar. (figura 5.15)

```
CLAUSES
    padre(Juan,pedro).
    padre(pedro,tibe).
    padre(tibe,camilo).
```

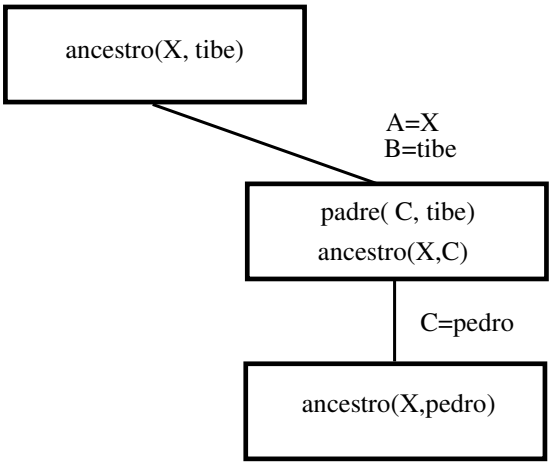
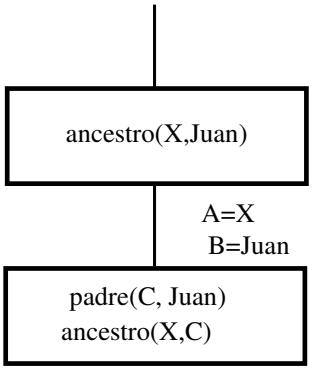


Figura 5.14

```
padre(rober,maria).
ancestro(A,B):-
    padre(A,B).
ancestro(A,B):-
    padre(C,B),
    ancestro(A,C).
```

Subobjetivos del nodo que se debe intentar expandir:

```
padre(C,Juan).
ancestro (X,C)
```



*

Figura 5.15

.:Actividad 5.2.

1. Represente mediante un árbol la búsqueda correspondiente al siguiente programa. Escriba la salida del programa en el orden exacto en que lo haría Prolog.

```
PREDICATES
    nondeterm jugador( symbol)
    nondeterm pareja ( symbol, symbol)
    nondeterm mayor  ( symbol, symbol)

CLAUSES
    jugador( pedro).
    jugador( Juan).
    jugador( Diana).
    mayor  ( pedro, Diana ).
    mayor  ( pedro, Juan).
    mayor  ( Juan, Diana ).
    pareja (X,Y):-
        jugador(X),
        jugador(Y),
        mayor  ( X, Y).
GOAL
    pareja(A,B).
```

2. Escriba y ejecute un programa Prolog para hallar los abuelos de Luis, con base en los hechos dados. No incluya en el programa ningún hecho adicional a los dados en el ejercicio. Prolog debe hacer todo el trabajo de inferencia.

   Hechos:

   Julio es el padre de Juan
   José es el padre de Julia
   Juan es el padre de Pedro
   Julia es la madre de Pedro
   Pedro es hermano de Luis

5.6 Flujo de un programa

En esta sección veremos cómo podemos desarrollar programas típicos de entrada - cálculo - salida en Prolog. Veremos también como el proceso de unificación hace innecesaria la sentencia if de los lenguajes de programación procedimentales.

5.6.1 Programa rectángulo

En esta sección veremos es el flujo de un programa que tiene la estructura: entrada - cálculo - salida en Prolog. También se presenta como el proceso de unificación hace innecesaria la sentencia `if` de los lenguajes de programación procedimentales.

Programa 5.13: Área de un rectángulo

```
1 PREDICATES
2     entrar   (Real, Real)
3     calcular (Real, Real, Real )
4     imprimir (Real)
5     ejecutar
6 CLAUSES
7     ejecutar:-
8         entrar(Base, Altura),
```

```
9           calcular(Base, Altura, Area),
10          imprimir(Area).
11      entrar(Base, Altura):-
12          write("Base: "),
13          readln(B),
14          str_real(B,Base),
15          write("Altura: "),
16          readln(A),
17          str_real(A,Altura).
18      calcular( Base, Altura, Area):-
19          Base * Altura = Area.
20      imprimir(Area):-
21          write( "Area: ", Area),nl.
22  GOAL
23      ejecutar.
```

Una meta sin argumentos

Programa 5.14: Area de un rectángulo - meta sin argumentos

```
1   CLAUSES
2       ejecutar:-
3           entrar(Base, Altura),
4           calcular(Base, Altura, Area),
5           imprimir(Area).
6       entrar(Base, Altura):-
7           write("Base: "),
8           readln(B),
9           str_real(B,Base),
10          write("Altura: "),
11          readln(A),
12          str_real(A,Altura).
13      calcular( Base, Altura, Area):-
14          Base * Altura = Area.
15      imprimir(Area):-
16                  write( "Area: ", Area),nl.
17  GOAL
18      ejecutar.
```

El predicado `ejecutar` no tiene argumentos (ver líneas 2 y 18 del programa 5.14). El programa no necesitará encontrar valores de variables que hagan verdadero el predicado. El programa unifica la meta con la cabeza de la primera regla. De hecho, la meta y la cabeza son idénticas. Como resultado de la unificación, habrá tres submetas para alcanzar: una entrada de datos, un cálculo y una impresión.

Predicados que no se unifican.

Programa 5.15: Area de un rectángulo - Predicados que no se unifican

```
1   entrar(Base, Altura):-
2           write("Base: "),
3           readln(B),
4           str_real(B,Base),
5           write("Altura: "),
6           readln(A),
7           str_real(A,Altura).
```

El predicado `entrar` en el programa 5.15, representa la meta de que el usuario ingrese por teclado la base y la altura de un rectángulo. Prolog alcanza esta meta con las siguientes acciones:

- Imprimir el letrero Base:

- Recibir por teclado el valor de la cadena de caracteres B
- Convertir la cadena de caracteres B a real. El valor real queda en la variable Base.
- Imprimir el letrero Altura:
- Recibir por teclado el valor de la cadena de caracteres A
- Convertir la cadena de caracteres A a real. El valor real queda en la variable Altura.

El programa no aplica el proceso de unificación a estas seis submetas. Simplemente ejecuta las acciones correspondientes indicadas por cada una de ellas.

Predicados calculables

```
calcular( Base, Altura, Area):-
        Base * Altura = Area.
```

Prolog tampoco aplica el proceso de unificación a la meta que está en negrilla, simplemente hará que la igualdad se cumpla. La forma de lograr esto es efectuando la multiplicación Base * Altura y almacenando el resultado en la variable Area. Esto es posible gracias a que la variable Area no tiene un valor previamente asignado.

No hay operador de asignación

```
imprimir:-
    A=3,
    write(A),  nl,
    A=4,
    write(A).
```

El operador igual `(=)` es un operador relacional y no un operador de asignación. En Prolog no existe el operador de asignación. En particular la reasignación es imposible. En la regla del ejemplo Prolog no asignará un valor de 4 a la variable A.

Prolog interpreta la meta A=4 como consiga que 3 sea igual a 4 y no como asigne a A el valor de 4. La salida del programa indicará que Prolog no puede lograr esto:

```
3
no
```

Impresión en pantalla.

```
    imprimir(Area):-
                write( "Area: ", Area),nl.
```

El predicado `write` tampoco cae en el proceso de unificación. La instrucción simplemente produce una salida. El predicado `nl`, `new line`, representa un avance de línea en la pantalla y se usa en la mayoría de las ocasiones para dar una mejor presentación a la salida de los programas que se están ejecutando.

Predicados deterministas

```
PREDICATES
    entrar   (Real, Real)
    calcular (Real, Real, Real )
    imprimir (Real)
    ejecutar
```

Todos los predicados de este programa son deterministas. Observe que no está la palabra `nondeterm` antes de cada predicado. Estos predicados solo pueden satisfacerse una vez, a diferencia de `detem`, los cuales son predicados que pueden tener varias instancias y se pueden satisfacer varias veces.

5.6.2 Raíz Cuadrada

El programa 5.16, recibe por teclado un número. Si el número es cero o positivo, calcula la raíz cuadrada del número y la imprime. Si el número es negativo el programa imprime un mensaje de error. El programa ilustra como gracias al proceso de unificación, Prolog no requiere de la sentencia de control `if`.

Programa 5.16: Raíz Cuadrada

```
1  PREDICATES
2      entrar ( Real)
3      calcular ( Real, Real )
4      imprimir (Real)
5      nondeterm ejecutar
6  CLAUSES
7      ejecutar:-
8          entrar(X),
9          X>=0,
10         calcular(X,Raiz),
11         imprimir(Raiz).
12     ejecutar:-
13         write("Numero negativo\n").
14     entrar(X):-
15         write("Numero: "),
16         readln(Xs),
17         str_real(Xs,X).
18     calcular( X, Raiz):-
19         Raiz = sqrt (X).
20     imprimir(Raiz):-
21         write("Raiz: "),
22         write( Raiz),nl.
23 GOAL
24     ejecutar.
```

Ejemplos de la ejecución del programa:

1. Número: 100

   ```
   Raíz: 10
   yes
   ```

2. Número: -1

   ```
   Número negativo
   yes
   ```

El predicado ejecutar no determinista

```
PREDICATES
        nondeterm ejecutar
    CLAUSES
        ejecutar:-
            entrar(X),
            X>=0,
            calcular(X,Raíz),
            imprimir(Raíz).

 ejecutar:-
            write("Número negativo\n").
    GOAL
        ejecutar.
```

El predicado ejecutar es no determinista, a pesar de que no tiene argumentos. Esto se debe a que puede unificarse con dos reglas distintas.

Unificación de ejecutar con la primera regla, operadores relacionales.

```
CLAUSES
        ejecutar:-
            entrar(X),
            X>=0,
            calcular(X,Raíz),
            imprimir(Raíz).
        ejecutar:-
            write("Número negativo\n").
    ...
    GOAL
        ejecutar.
```

Primero, el programa unifica el predicado `ejecutar` con la primera regla y, por lo tanto, intenta alcanzar cuatro submetas:

1. Recibir por teclado un valor de X.
2. Verificar que $X \geq 0$ sea verdadero.
3. Hallar la raíz cuadrada
4. Imprimir el resultado.

Si el usuario entra un valor negativo falla la segunda submeta y el programa no intenta alcanzar la tercera ni la cuarta. Podríamos decir que en este programa hay una sentencia `if` implícita.

Unificación de ejecutar con la segunda regla.

```
CLAUSES
        ejecutar:-
```

```
            entrar(X),
            X>=0,
            calcular(X,Raíz),
            imprimir(Raíz).
        ejecutar:-
            write("Número negativo\n").
    ...
    GOAL
        ejecutar.
```

Cuando un número negativo hace fallar la primera regla, el programa procede a unificar ejecutar con la segunda regla. Como resultado de esto imprime el mensaje Número negativo.

5.6.3 Funciones de Prolog

```
calcular( X, Raíz):-
        Raíz = sqrt (X).
```

Prolog incluye muchas funciones aritméticas como las siguientes: `sqrt`, `exp`, `sin`, `cos`, `tan`, `abs`, `log` y `ln`. En diferentes manuales se encuentran una descripción detallada de estas funciones. Para más información consultar:

`http://www.swi-prolog.org/`

.:Actividad 5.3.

Elabore programas Prolog para resolver los siguientes problemas. Pruebe en el computador los programas.

- Resolver una ecuación de primer grado: $AX + B = 0$, donde $A \neq 0$. Si el usuario entra un valor de A igual a cero, el programa debe imprimir el mensaje correspondiente.
- Hallar la nota definitiva de una asignatura, la cual es el promedio de dos notas parciales. Además, el programa imprimir el mensaje ganó o perdió según corresponda. La nota aprobatoria es de 3.0.
- Hallar el mayor de tres números enteros,
- Hallar las raíces reales de una ecuación de segundo grado. $AX^2 + BX + C = 0$, donde $A \neq 0$. El programa imprimir los mensajes correspondientes en caso de que A sea igual a 0, o en caso de que el discriminante sea negativo.

5.7 Recursión

Si un problema puede resolverse usando la solución de versiones más pequeñas de sí mismo, y esas versiones más pequeñas, se reducen fácilmente a problemas solubles, entonces se tiene un algoritmo recursivo (Baldwin & Douglas,2004). La recursión es una forma alternativa, para la resolución de problemas algorítmicos y computacionales. En toda definición recursiva de un problema se debe establecer un estado base, es decir, un estado en el cual la solución no se presente de manera recursiva sino directamente. Esto permite que las llamadas recursivas no continúen indefinidamente (Cardona, Jaramillo & Carmona, 2007)

Si consultamos un manual de Prolog nos encontraremos con una sorpresa: no hay bucles en Prolog. Sentencias como `for`, `do ... while` y `while` ..., no existen en Prolog. Solo hay un camino para lograr la repetición: La recursión.

La llamada a un algoritmo recursivo conduce a una nueva versión del método que comienza a ejecutarse, también a una organización de la gestión de los parámetros y la memoria.

En esta sección veremos diversos ejemplos que logran con recursión lo que, en otros lenguajes, se hace con bucles.

5.7.1 Multiplicación

El siguiente método retorna la multiplicación de dos números enteros de forma recursiva. El siguiente caso de prueba que permita multiplicar los (5 * 3) recursivamente:

$$\begin{aligned} 5 \times 3 &= 5 + 5 \times 2 \\ 5 \times 2 &= 5 + 5 = 10 \\ 5 \times 1 &= 5 \end{aligned}$$

Generalizando se tiene:

$$\begin{aligned} a \times b &= a, && \text{si } b = 1 \\ a \times b &= a + a \times (b-1), && \text{si } b > 1 \end{aligned}$$

A continuación, se muestra una prueba para este algoritmo recursivo.

$$\begin{aligned} \mathit{multiplicar}(5,4) &= 5 + \mathit{multiplicar}(5,3) \\ &= 5 + (5 + (\mathit{multiplicar}(5,2))) \\ &= 5 + (5 + (5 + \mathit{multiplicar}(5,1)) \\ &= 5 + (5 + (5 + 5)) \\ &= 5 + (5 + 10) \\ &= 5 + 15 \\ &= 20 \end{aligned}$$

En la siguiente figura se muestra una de las formas como se puede representar este algoritmo recursivo. (figura 5.16)

5.7.2 Factorial de un número

Inicialmente se muestra una deducción informal de cómo hallar un número factorial cualquiera. Definición de factorial para $n \geq 0$.

$$\begin{aligned} 0! &= 1 \\ 1! &= 1 \\ 2! &= 2 \times 1 = 2 \times 1! \\ 3! &= 3 \times 2 \times 1 = 3 \times 2! \\ 4! &= 4 \times 3 \times 2 \times 1 = 4 \times 3! \\ &\vdots \quad \vdots \quad \vdots \\ n! &= n \times (n-1)! \end{aligned}$$

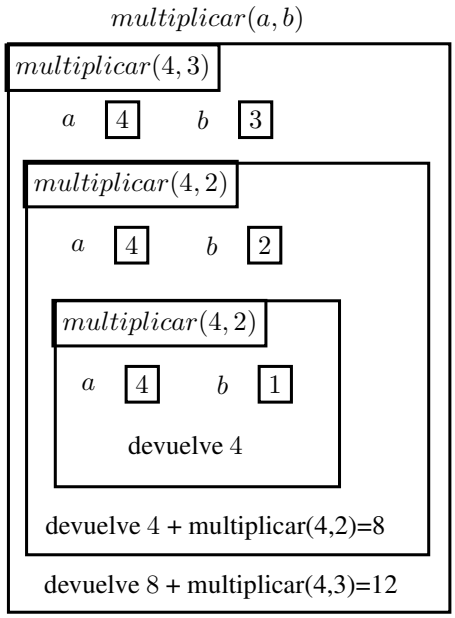


Figura 5.16: Algoritmo recursivo multiplicar

El factorial está definido en forma recursiva. Es muy natural, entonces, hacer un programa recursivo para hallar el factorial. La definición es: (figura 5.17)
A continuación se muestra una instancia del algoritmo factorial.

$$\begin{aligned} \text{factorial}(3) &= 3 \times (\text{factorial}(2)) \\ &= 3 \times (2 \times \text{factorial}(1)) \\ &= 3 \times (2 \times 1) \\ &= 3 \times 2 \\ &= 6 \end{aligned}$$

A continuación el programa, la salida y varias explicaciones:

Programa 5.17: Factorial de un numero natural

```
1  PREDICATES
2      factorial( ushort N, unsigned F)
3  CLAUSES
4      factorial(0,1):-!.
5      factorial(N,F):-
6          M=N-1,
7          factorial(M,F1),
8          F=F1*N.
9  GOAL
10     factorial(5,F).
11 Salida:
12         F=120
13         1 Solution
```

1.

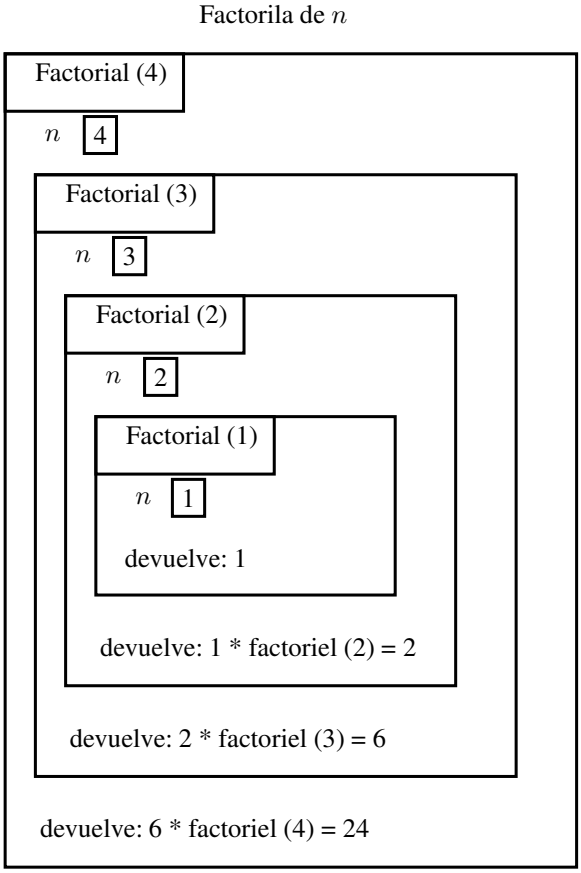


Figura 5.17

```
Predicates
      factorial( ushort N, unsigned F)
```

Observamos dos tipos de datos enteros, que no habíamos usado hasta ahora. Para este caso se debe trabajar con el valor al cual se le desea calcular su factorial.

2.

```
factorial(0,1):-!.
```

Esta regla especifica el caso trivial: el factorial de cero es uno. Más adelante se explica el significado del signo de admiración.

3.

```
factorial(N,F):-
        M=N-1,
        factorial(M,F1),
        F=F1*N.
```

La regla codifica el método recursivo. Para hallar el factorial de N al que llamamos F el programa debe:

1. Hallar M, el entero anterior a N, con una resta.
2. Hallar F1, el factorial de M, utilizando el predicado recursivamente.
3. Hallar F, el factorial de N, con una multiplicación

4. `factorial(5,F).`

   El programa se ha codificado con el número 5 fijo, por simplicidad. Es fácil adecuar el programa para que reciba por teclado un número cualquiera. En este momento el lector ya debe tener los conocimientos necesarios para hacer esto.

Supongamos que en algún momento de la ejecución, el programa se encuentra en la situación ilustrada abajo. La meta del momento es hallar el factorial de cero. Vemos que la submeta se puede unificar con ambas reglas.

```
factorial(N,F):-
        M=N-1,
        factorial(M,F1),          Submeta:
        F=F1*N.                   factorial(0,F).
```

De la unificación con la primera regla, se encuentra que $F = 1$, lo cual es lo correcto. Pero de la unificación con la segunda regla se llega a: $N = 0$, $M = -1$, $F1 = (-1)!$, lo cual no tiene sentido.

Se requiere de una manera de evitar que se realice la unificación con la segunda regla. Esta es la misión del símbolo de admiración (!). El símbolo se lee como cortar, es decir detener la búsqueda.

Observamos en el código que el predicado `factorial` es determinista, no lleva la palabra `nondeterm`. El predicado es determinista a pesar de que se pueda unificar con dos reglas. Esto también es una consecuencia del signo de admiración que evita que las dos unificaciones sean exitosas. Si la primera unificación es exitosa ya no se intenta la segunda. Una unificación exitosa es una que conduce a alcanzar una meta en algún momento de la ejecución.

.:Actividad 5.4.

1. Se desea escribir un programa recursivo que calcule la sumatoria de los primeros n números enteros positivos. Por ejemplo si se desea calcular la sumatoria de los 5 primeros números, una posible abstracción para resolver el problema puede ser la siguiente:

$$\begin{aligned}
\text{sumatoria}(5) &= 5 + \text{sumatoria}(4) \\
&= 5 + (4 + (\text{sumatoria}(3))) \\
&= 5 + (4 + (3 + \text{sumatoria}(2))) \\
&= 5 + (4 + (3 + (2 + \text{sumatoria}(1)))) \\
&= 5 + (4 + (3 + (2 + 1))) \\
&= 5 + (4 + (3 + 3)) \\
&= 5 + (4 + 6) \\
&= 5 + 10 \\
&= 15
\end{aligned}$$

   Gráficamente se puede mostrar esta funcionalidad. (figura 5.18)

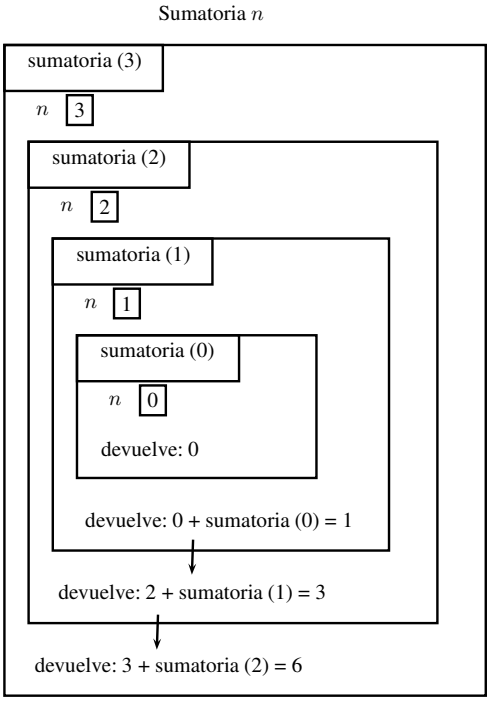


Figura 5.18

2. Deduzca que problema está resolviendo la siguiente representación, y escriba un algoritmo recursivo que resuelva este algoritmo. (figura 5.19)

.:Actividad 5.5.

1. Elabore programas para resolver los siguientes problemas. :

   - Hallar el n-ésimo término de la sucesión de Fibonacci 1, 1, 2, 3, 5, 8, ... Por ejemplo si se le pide al programa que halle el séptimo término, el programa debe responder que es 13.

     La serie de Fibonacci fue Introducida por el matemático Leonardo de Pisa, quien se llamaba a si mismo Fibonacci. Cada número de la secuencia se obtiene sumando los dos anteriores. Entonces:
     $fib(0) = 1$
     $fib(1) = 1$
     $fib(2) = fib(0) + fib(1) = 2$
     $fib(3) = fib(2) + fib(1) = 2 + 1 = 3$
     .
     .
     .
     $fib(n) = fib(n-1) + fib(n-2)$

     Por definición, los dos primeros valores son 0 y 1 respectivamente. Los otros números de la sucesión se calculan sumando los dos números que le preceden. Escriba un programa en Prolog que resuelva la serie de Fibonacci.
   - Imprimir los números desde 1 hasta un N dado

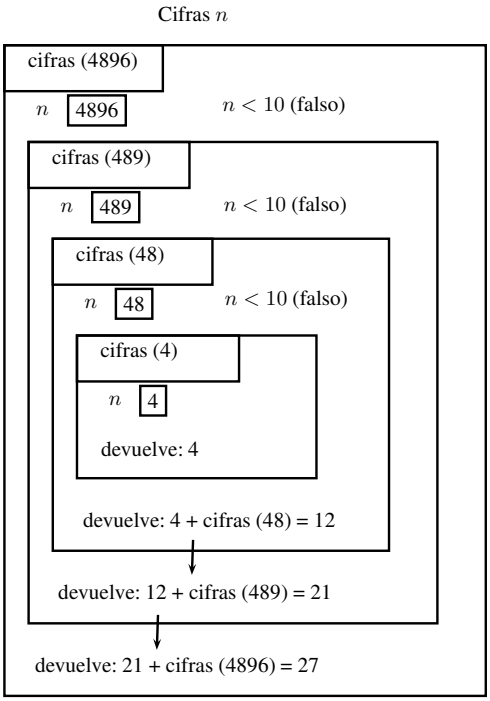

Cifras n
cifras (4896)
n 4896
$n < 10$ (falso)
cifras (489)
n 489
$n < 10$ (falso)
cifras (48)
n 48
$n < 10$ (falso)
cifras (4)
n 4
devuelve: 4
devuelve: 4 + cifras (48) = 12
devuelve: 12 + cifras (489) = 21
devuelve: 21 + cifras (4896) = 27

Figura 5.19

CAPÍTULO 6

Proyecto Final - Herramienta de soporte al Método de Resolución Proposicional

6.1 Contexto y Motivación

En el campo de la Lógica Matemática una de las áreas de investigación es la demostración automática. La demostración automática se ocupa entre otras, del desarrollo de métodos orientados a la automatización de pruebas formales para identificar propiedades de las fórmulas proposicionales, así mismo, tiene como objetivo la búsqueda y el estudio de algoritmos para resolver ciertos problemas lógicos.

Uno de los métodos de demostración automática es el Método de Resolución, el cual fue diseñando por J.A Robinson (Robinson, 1965), con el propósito de verificar si un conjunto de cláusulas es insatisfacible. El método usa una única regla de inferencia, llamada Regla de Resolución. Una de sus características, es que tiene como precondición que el conjunto de fórmulas deben ser cláusulas. Para ello es necesario transformar la formula proposicional a su equivalente en Forma Normal Conjuntiva (FNC).
Los siguientes elementos conceptuales permiten una aproximación al problema en cuestión:

- Una fórmula está en FNC si esta es una conjunción de disyunciones de literales.
- Un literal es una proposición atómica p o la negación de la proposición atómica $\neg p$.
- Una cláusula se compone de literales que se encuentran unidos por medio del operador de disyunción, esta disyunción se puede dar entre cero o más literales.
- Una fórmula está en Forma Clausal si se expresa como un conjunto de cláusulas. La transformación de una fórmula en FNC a Forma Clausal es inmediata sustituyendo las conectivas $\wedge$ por comas y englobando las disyunciones entre llaves (Labra & Fernández, 1998).

Por ejemplo la FNC: $(p \vee \neg p \wedge q) \wedge (\neg q \vee p)$, se puede simplificar en: $\{p\neg pq, \neg qp\}$.

La regla de resolución se aplica a pares de la cláusulas en las cuales existan literales complementarios entre ellas. En este caso, dadas dos cláusulas C_1 y C_2, en las cuales se tengan un literal de forma tal que $l \in C_1$ y $\neg l \in C_2$, se denomina resolvente de C_1 y C_2 a la cláusula respecto al literal l:

$$R_l(C_1, C_2) = (C_1 - \{l\}) \cup (C_2 - \{l\})$$

Por ejemplo si se tienen las cláusulas: $C_1 = pq\neg r$ y $C_2 = qr\neg t$, se tiene que el literal c, está en $C_1 = \neg r$ y en $C_2 = r$, por lo tanto se puede obtener el resolvente entre C_1 y C_2:

$$R(C_1, C_2) = (pq\neg r - \{\neg r\}) \cup (qr\neg t - \{r\}) = pq \cup q\neg t = pq\neg t$$

6.2 Proyecto

El propósito de este proyecto consiste en el diseño e implementación de una herramienta informática, que permita la comprobación automática de fórmulas proposicionales, aplicando el método de **Resolución Proposicional**.

6.3 Requerimientos del proyecto

La herramienta informática de soporte al método de Resolución, debe permitir:

- Ingresar al usuario una fórmula proposicional compuesta, mediante una colección de botones básicos o mediante teclado.
- Validar la correctitud de la fórmula proposicional ingresada por el usuario.
- Transformar la formula proposicional a su equivalente en FNC.
- Convertir la FNC a Forma Clausal.
- Mostrar las formulas en forma Clausal.
- Aplicar el Algoritmo de Resolución.
- Determinar si el conjunto de fórmulas es satisfacible o insatisfacible.

La herramienta debe cumplir con los siguientes ítems.

- La herramienta debe contener un manual de uso de la aplicación.
- La interfaz gráfica debe permitir el ingreso de los operadores: unarios $\neg$, binarios $\vee$, $\wedge$, $\rightarrow$, y de paréntesis ().
- Documentar el código con comentarios JavaDoc.

4. Fases del proyecto
Para el desarrollo del proyecto del curso se definen una serie de fases orientadas a interpretar, argumentar y resolver el problema planteado. Este proyecto está definido en tres fases, para cada una de las cuales el profesor define unas evidencias (productos) que deben ser entregadas en equipos de trabajo. Para cada una de las fases el profesor propone a los estudiantes un proceso de evaluación. A continuación se hace una descripción de la fase y de las evidencias a entregar:

6.3.1 Fase 1

En esta fase está orientada a que el estudiante interprete adecuadamente el problema a resolver en el proyecto. Las evidencias que se deben generar en esta fase son:

1. Documento con una síntesis de los principales aspectos conceptuales del método de Resolución.
2. Diagrama de flujo con el proceso para determinar la satisfacibilidad o insatisfacibilidad de una fórmula proposicional, basado en el Método de Resolución.

6.3.2 Fase 2

En esta fase está orientada a que el estudiante argumente mediante un diseño, el problema a resolver. Las evidencias que se deben generar en esta fase son:

1. Diagrama de clases con las clases de implementación y sus asociaciones.
2. Proyecto en java con código fuente que soporta las funcionalidades básicas de acuerdo a los requerimientos del proyecto.

6.3.3 Fase 3

En esta fase está orientada a que el estudiante argumente mediante un diseño, el problema a resolver. Las evidencias que se deben generar en esta fase son:

1. Proyecto en java que soporta adecuadamente el método de resolución.

6.4 Evidencias de las fases

El proceso de evaluación de la evidencia se basará en una rúbrica por indicadores. Los indicadores están asociados a la evidencia que se debe presentar en grupos de trabajo.

Cada uno de los indicadores tendrá cuatro niveles de desempeño (Excelente, Bueno, Suficiente, Deficiente), los cuales se presenta de mayor a menos nivel de complejidad. A continuación se presenta la descripción del indicador y los niveles de desempeño.

6.4.1 Fase 1

Las evidencias que se deben generar en esta fase son:

1. Documento con una síntesis de los principales aspectos conceptuales del método de Resolución.(ver tabla 6.1)

2. Diagrama de flujo con el proceso para determinar la satisfacibilidad o insatisfacibilidad de una fórmula proposicional, basado en el Método de Resolución. (ver tabla 6.2)

Indicador	Excelente	Bueno	Suficiente	Deficiente
Se presenta en un documento con una síntesis conceptual del método de resolución.	El documento presenta un análisis conceptual del método de Resolución, con el desarrollo de una idea creativa (mapa conceptual, mapa mental), con coherencia entre sí para apoyar la idea general. El análisis conceptual tiene entre cero y dos errores ortográficos, con un manejo adecuado del vocabulario, acorde con la bibliografía de referencia y con el apoyo de mínimo tres fuentes bibliográficas pertinentes según la norma APA.	El documento presenta un análisis conceptual del método de Resolución, presentando coherencia entre la introducción, el desarrollo y las conclusiones. El análisis conceptual tiene menos de 5 errores de ortografía, con un manejo adecuado del vocabulario y con el apoyo de mínimo dos fuentes bibliográficas pertinentes y completas según la norma APA.	El documento contiene un análisis conceptual general sobre el método de Resolución. El análisis conceptual tiene tienen entre 5 y 10 errores ortográficos. Se cita al menos una fuente bibliográfica.	El documento contiene un análisis conceptual general sobre el método de Resolución, con baja explicación y sin citar autores. Se tienen más de 10 errores de ortografía o de sintaxis.

Tabla 6.1

Indicador	**Excelente**	**Bueno**	**Suficiente**	**Deficiente**
Se presenta un diagrama de flujo (DFD) con el proceso para determinar la satisfacibilidad o insatisfacibilidad de una fórmula proposicional.	El DFD presentado demuestra compresión del método de resolución, mostrando con claridad los procesos y el flujo de todas las posibles secuencias, orientadas a determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado es claro, con buen uso de la ortografía, utiliza adecuadamente la notación permitida y las relaciones entre los elementos son coherentes	El DFD presentado demuestra compresión parcial del método de resolución, mostrando algunos procesos y el flujo de algunas de las posibles secuencias, orientadas a determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado es claro, con adecuada ortografía y utiliza adecuadamente la notación permitida, pero las relaciones entre los elementos no son coherentes	El DFD presentado demuestra compresión parcial del método de resolución, pero sin mostrar los procesos básicos y el flujo de las secuencias básicas, orientadas a determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado es claro, presenta errores de ortografía y no utiliza adecuadamente la notación definida.	El DFD presenta desconocimiento de los procesos y acciones a realizar para determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado no es claro, utiliza notación no permitida y presenta errores de ortografía.

Tabla 6.2

6.4.2 Fase 2

Las evidencias que se deben generar en esta fase son:

1. Diagrama de clases con las clases de implementación y sus asociaciones. (ver tabla 6.3)

2. Proyecto en java con código fuente que soporta las funcionalidades básicas de acuerdo a los requerimientos del proyecto. (ver tabla 6.4)

Indicador	**Excelente**	**Bueno**	**Suficiente**	**Deficiente**
Diagrama de clases con las clases de implementación y sus asociaciones.	Se presenta un diagrama de clases completo con la información y presentación adecuada de acuerdo a la semántica y sintaxis permitida (atributos, métodos, asociaciones, cardinalidad), obteniendo un diagrama coherente entre todas los clases y sus relaciones. El diagrama de clases presenta asociaciones de agregación, composición y herencia. La estructura del diagrama de clases se refleja totalmente en la implementación del código fuente.	Se presenta un diagrama de clases con la información básica (atributos, métodos y asociaciones), con coherencia entre las clases y sus relaciones. La mayoría de las clases presenta relaciones adecuadas dando como resultado un diagrama fácil de comprender de acuerdo a la semántica permitida. La estructura del diagrama de clases se refleja parcialmente en la implementación del código fuente.	El diagrama de clases cuenta con la información adecuada, pero no responde a los elementos semánticos permitidos. Los elementos del diagrama y sus relaciones son incongruentes dando como resultado un diagrama de difícil interpretación. La estructura del diagrama de clases no se refleja en la implementación del código fuente.	El diagrama de clases no contiene la información adecuada ni responde a los elementos semánticos permitidos. No es posible una interpretación del diagrama. La estructura del diagrama de clases no se refleja en la implementación del código fuente.

Tabla 6.3

Indicador	**Excelente**	**Bueno**	**Suficiente**	**Deficiente**
Proyecto en java con código fuente que soporta funcionalidades básicas de los requerimientos del proyecto.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario completa y la cual contiene botones de interacción. Se valida la correctitud de la formula proposicional, mediante la gestión de excepciones. La estructura de la aplicación está acorde con el diagrama de clases presentado. La documentación se realiza considerando el estándar java Doc. La documentación no contiene errores de ortografía.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario básica y la cual contiene botones de interacción. No se valida la correctitud de la formula proposicional La estructura de la aplicación está parcialmente acorde con el diagrama de clases presentado. La documentación se realiza de una forma básica. Se consideran convenciones básicas para los nombres de las variables y para los distintos tipos de comentarios. La documentación tiene entre 1 y 5 errores de ortografía.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario básica. No se valida la correctitud de la formula proposicional La estructura de la aplicación está parcialmente acorde con el diagrama de clases presentado. La documentación se realiza de una forma muy básica. No se consideran convenciones adecuadas para los nombres de las variables y para los distintos tipos de comentarios. La documentación presenta entre 6 y 10 errores de ortografía.	La aplicación permite al usuario ingresar una fórmula proposicional compuesta. No se valida la correctitud de la formula proposicional ingresada. El código fuente de los archivos no se encuentra documentada. La estructura de la aplicación no está acorde con el diagrama de clases presentado.

Tabla 6.4

6.4.3 Fase 3

Las evidencias que se deben generar en esta fase son:

1. Proyecto en java que soporta el método de resolución.(ver tabla 6.5)

Indicador	Excelente	Bueno	Suficiente	Deficiente
Proyecto en java que soporta el método de Resolución.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario completa y la cual contiene botones de interacción. La aplicación permite al usuario ingresar una fórmula proposicional compuesta y se valida la correctitud de la formula ingresada mediante la gestión de excepciones. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La documentación se realiza considerando el estándar java Doc. La documentación no contiene errores de ortografía.	La aplicación permite el ingreso de todos los operadores lógicos mediante una interfaz gráfica completa y con botones de interacción. La aplicación permite al usuario ingresar una fórmula proposicional compuesta y se valida la correctitud de la formula ingresada mediante la gestión de excepciones. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La documentación se realiza de una forma básica. Se consideran convenciones básicas para los nombres de las variables y para los distintos tipos de comentarios. La documentación tiene entre 1 y 5 errores de ortografía.	La aplicación permite el ingreso de todos los operadores lógicos mediante una interfaz gráfica básica. La aplicación permite al usuario ingresar una fórmula proposicional compuesta, pero no se valida la correctitud de la formula ingresada. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La documentación se realiza de una forma muy básica. No se consideran convenciones adecuadas para los nombres de las variables y para los distintos tipos de comentarios. La documentación presenta entre 6 y 10 errores de ortografía.	La aplicación permite el ingreso de todos los operadores lógicos mediante una interfaz gráfica básica. La aplicación permite al usuario ingresar una fórmula proposicional compuesta, pero no se valida la correctitud de la formula ingresada. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. El código fuente de los archivos no se encuentra documentado.

Tabla 6.5

CAPÍTULO 7

Proyecto Formativo - Construyendo tableros semánticos para el análisis de proposiciones

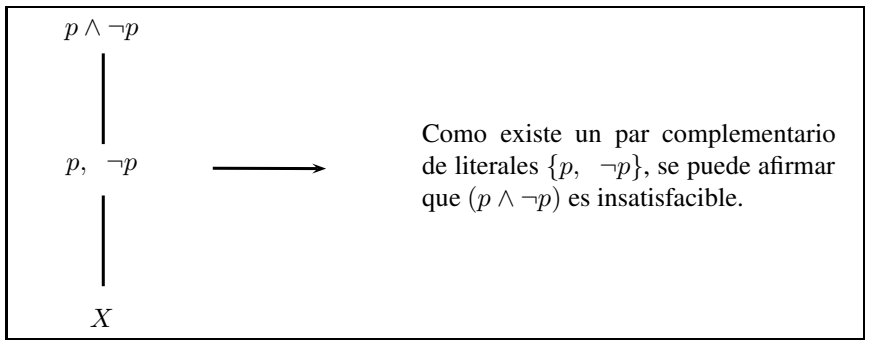
7.1 Contexto y motivación

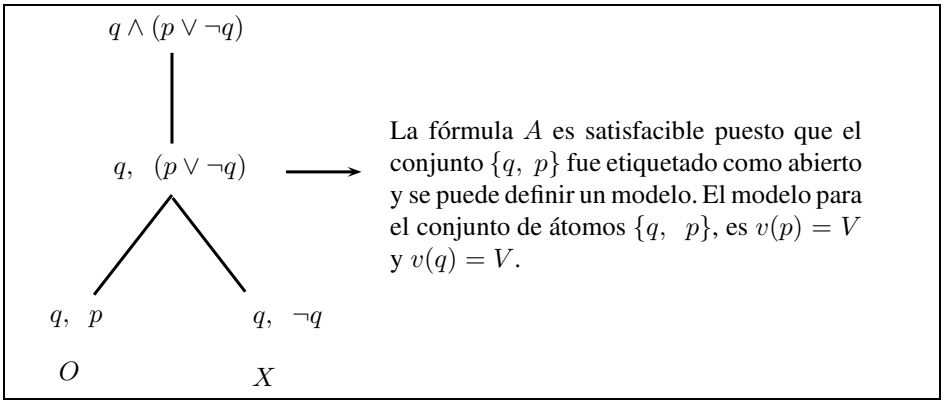
Dentro del campo de la Inteligencia Artificial una de las áreas de interés de investigación es la demostración automática. La demostración automática se ocupa entre otras, del desarrollo de métodos orientados a la automatización de pruebas formales en fórmulas proposicionales y su implementación en lenguajes de programación. Dentro de los métodos de demostración automática se encuentran los Tableros Semántico o Tableaux Semánticos, los cuales presentan los sistemas lógicos de una forma intuitiva, clara y concisa.

El método de los tableros semánticos realiza una búsqueda sistemática de modelos que verifica ciertas condiciones en una fórmula proposicional, disminuyendo la satisfacibilidad de toda la fórmula a la de algunos literales de la misma, es decir, la satisfacibilidad para una fórmula proposicional A queda asociada a la satisfacibilidad de un conjunto de literales.

El procedimiento de decisión de los tableros semánticos puede ser representado mediante un árbol de formación. La fórmula base representa la raíz del árbol y cada una de las hojas generadas como hijos de la raíz son producto de una búsqueda sistemática y serán considerados como nuevos nodos o nuevas fórmulas del árbol. El procedimiento del método finaliza cuando todas las hojas o fórmulas se encuentran debidamente marcadas aplicando α-reglas y β-reglas. Una hoja que contiene un conjunto complementario de literales será marcada con una X y una hoja que no tenga un conjunto complementario de literales será marcada con una O. Si la hoja se marca con una X, esta hoja será insatisfacible (no tiene modelo), en caso contrario esta será satisfacible, (tiene modelo). Un árbol etiquetado en todas sus hojas, es llamado un tablero semántico.

Ejemplo: Un tablero semántico (árbol que se encuentra etiquetado) para la fórmula $A: \ (p \wedge \neg p)$ se muestra en el siguiente árbol:





7.2 Fases del proyecto

Para el desarrollo del proyecto del curso se definen una serie de fases articuladas orientadas a interpretar, argumentar y resolver el problema planteado. Este proyecto está definido para cuatro fases: direccionamiento, planeación, ejecución y socialización, en las cuales intervienen en profesor y los estudiantes. A continuación se hace una descripción de cada una de estas fases.

- **Direccionamiento**: Esta fase es de responsabilidad del profesor. Se establecen los criterios necesarios a tener en cuenta en el proyecto. Se presenta a los estudiantes el proyecto a realizar, el proceso de evaluación y los recursos.
- **Planeación**: El profesor define los requerimientos del proyecto y las evidencias a generar en esta fase. Los estudiantes en equipos de trabajo planifican la realización del proyecto.
- **Ejecución**: En esta fase los estudiantes desarrollan el proyecto especificados en la fase anterior, buscando resolver el problema planteado.
- **Socialización**: Presentar a la comunidad académica el proceso y los resultados alcanzados, en la realización del proyecto.

7.2.1 Evidencias y Proceso de Evaluación

El proceso de evaluación de la evidencia se basará en una rúbrica por indicadores. Los indicadores están asociados a la evidencia que se debe presentar en grupos de trabajo. Los indicadores para evaluar las evidencias son: (tabla 7.1)

1. Se presenta en un blog un análisis conceptual de los tableros semánticos.

   Cada uno de los indicadores tendrá cuatro niveles de desempeño (Excelente, Bueno, Suficiente, Deficiente), los cuales se presenta de mayor a menos nivel de complejidad. A continuación se presenta la descripción del indicador y los niveles de desempeño.

2. Se presenta en un blog un análisis conceptual de los tableros semánticos.

Tabla 7.1

Indicador	**Excelente**	**Bueno**	**Suficiente**	**Deficiente**
Se presenta en un blog un análisis conceptual de los tableros semánticos.	El blog posee una entrada con un análisis conceptual de los tableros semánticos, con el desarrollo de una idea creativa (mapa conceptual, mapa mental), con coherencia entre sí para apoyar la idea general. El análisis conceptual tiene entre cero y dos errores ortográficos, con un manejo adecuado del vocabulario, acorde con la bibliografía de referencia y con el apoyo de mínimo tres fuentes bibliográficas pertinentes según la norma APA.	El blog contiene una entrada con un análisis conceptual de los tableros semánticos, presentando coherencia entre la introducción, el desarrollo y las conclusiones. El análisis conceptual tiene menos de 5 errores de ortografía, con un manejo adecuado del vocabulario y con el apoyo de mínimo dos fuentes bibliográficas pertinentes y completas según la norma APA.	El blog contiene una entrada con un análisis conceptual general sobre los tableros semánticos. El análisis conceptual tiene tienen entre 5 y 10 errores ortográficos. Se cita al menos una fuente bibliográfica.	El blog contiene una idea general sobre los tableros semánticos, con baja explicación y sin citar autores. Se tienen más de 10 errores de ortografía o de sintaxis.

7.2.2 Evidencias de la fase de ejecución

El proceso de evaluación de la evidencia se basará en una rúbrica por indicadores. Los indicadores están asociados a la evidencia que se debe presentar en grupos de trabajo. Los indicadores para evaluar las evidencias son:

1. Se presenta un proyecto en java con código fuente que soporta las funcionalidades básicas de acuerdo a los requerimientos del proyecto. (tabla 7.2)
2. Se presenta un diagrama de clases con las clases de implementación y sus asociaciones. (tabla 7.3)
3. Se presenta un diagrama de flujo con el proceso para determinar la satisfacibilidad o insatisfacibilidad de una fórmula proposicional. (tabla 7.4)

Tabla 7.2

Indicador	Excelente	Bueno	Suficiente	Deficiente
Proyecto en java con código fuente que soporta funcionalidades básicas de los requerimientos del proyecto.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario completa y la cual contiene botones de interacción. Se valida la correctitud de la formula proposicional, mediante la gestión de excepciones. La estructura de la aplicación está acorde con el diagrama de clases presentado. La documentación se realiza considerando el estándar java Doc. La documentación no contiene errores de ortografía.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario básica y la cual contiene botones de interacción. No se valida la correctitud de la formula proposicional La estructura de la aplicación está parcialmente acorde con el diagrama de clases presentado. La documentación se realiza de una forma básica. Se consideran convenciones básicas para los nombres de las variables y para los distintos tipos de comentarios. La documentación tiene entre 1 y 5 errores de ortografía.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario básica No se valida la correctitud de la formula proposicional La estructura de la aplicación está parcialmente acorde con el diagrama de clases presentado. La documentación se realiza de una forma muy básica. No se consideran convenciones adecuadas para los nombres de las variables y para los distintos tipos de comentarios. La documentación presenta entre 6 y 10 errores de ortografía.	La aplicación permite al usuario ingresar una fórmula proposicional compuesta. No se valida la correctitud de la formula proposicional ingresada. El código fuente de los archivos no se encuentra documentada. La estructura de la aplicación no está acorde con el diagrama de clases presentado.

Indicador	Excelente	Bueno	Suficiente	Deficiente
Diagrama de clases con las clases de implementación y sus asociaciones.	Se presenta un diagrama de clases completo con la información y presentación adecuada de acuerdo a la semántica y sintaxis permitida (atributos, métodos, asociaciones, cardinalidad), obteniendo un diagrama coherente entre todas los clases y sus relaciones. El diagrama de clases presenta asociaciones de agregación, composición y herencia. La estructura del diagrama de clases se refleja totalmente en la implementación del código fuente.	Se presenta un diagrama de clases con la información básica (atributos, métodos y asociaciones), con coherencia entre las clases y sus relaciones. La mayoría de las clases presenta relaciones adecuadas dando como resultado un diagrama fácil de comprender de acuerdo a la semántica permitida. La estructura del diagrama de clases se refleja parcialmente en la implementación del código fuente.	El diagrama de clases cuenta con la información adecuada, pero no responde a los elementos semánticos permitidos. Los elementos del diagrama y sus relaciones son incongruentes dando como resultado un diagrama de difícil interpretación. La estructura del diagrama de clases no se refleja en la implementación del código fuente.	El diagrama de clases no contiene la información adecuada ni responde a los elementos semánticos permitidos. No es posible una interpretación del diagrama. La estructura del diagrama de clases no se refleja en la implementación del código fuente.

Tabla 7.3

Tabla 7.4

Indicador	**Excelente**	**Bueno**	**Suficiente**	**Deficiente**
Se presenta un diagrama de flujo (DFD) con el proceso para determinar la satisfacibilidad o insatisfacibilidad de una fórmula proposicional.	El DFD presentado demuestra compresión de los tableros semánticos, mostrando con claridad los procesos y el flujo de todas las posibles secuencias, orientadas a determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado es claro, utiliza la notación permitida y las relaciones entre los elementos son coherentes. El DFD no presenta errores de ortografía.	El DFD presentado demuestra compresión parcial de los tableros semánticos, mostrando algunos procesos y el flujo de algunas de las posibles secuencias, orientadas a determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado es claro, utiliza la notación permitida, pero las relaciones entre los elementos no son coherentes. El DFD no presenta errores de ortografía.	El DFD presentado demuestra compresión parcial de los tableros semánticos, pero sin mostrar los procesos básicos y el flujo de las secuencias básicas, orientadas a determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado es claro, pero utiliza no utiliza adecuadamente la notación definida. El DFD presenta errores de ortografía.	El DFD presenta desconocimiento de los procesos y acciones a realizar para determinar la insatisfacibilidad y satisfacibilidad de una formula proposicional. El DFD presentado no es claro, utiliza notación no permitida. El DFD presenta errores de ortografía.

7.2.3 Evidencias de la fase de socialización

1. Se presenta un proyecto en java que soporta el método de los tableros semánticos. (tabla 7.5)

Indicador	Excelente	Bueno	Suficiente	Deficiente
Proyecto en java que soporta el método de los tableros semánticos.	La aplicación permite al usuario ingresar una formula proposicional compuesta mediante una interfaz de usuario completa y la cual contiene botones de interacción. La aplicación permite al usuario ingresar una fórmula proposicional compuesta y se valida la correctitud de la formula ingresada mediante la gestión de excepciones. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La aplicación presenta uno o varios modelos cuando la formula proposicional es satisfacible. La aplicación muestra en la interfaz el árbol de formación producto de aplicar el método de los tableros semánticos. La documentación se realiza considerando el estándar java Doc. La documentación no contiene errores de ortografía.	La aplicación permite el ingreso de todos los operadores lógicos mediante una interfaz gráfica completa y con botones de interacción. La aplicación permite al usuario ingresar una fórmula proposicional compuesta y se valida la correctitud de la formula ingresada mediante la gestión de excepciones. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La aplicación presenta uno o varios modelos cuando la formula proposicional es satisfacible. La aplicación no muestra en la interfaz el árbol de formación producto de aplicar el método de los tableros semánticos. La documentación se realiza de una forma básica. Se consideran convenciones básicas para los nombres de las variables y para los distintos tipos de comentarios. La documentación tiene entre 1 y 5 errores de ortografía.	La aplicación permite el ingreso de todos los operadores lógicos mediante una interfaz gráfica básica. La aplicación permite al usuario ingresar una fórmula proposicional compuesta, pero no se valida la correctitud de la formula ingresada. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La aplicación presenta un modelo cuando la formula proposicional es satisfacible. La aplicación no muestra en la interfaz el árbol de formación producto de aplicar el método de los tableros semánticos. La documentación se realiza de una forma muy básica. No se consideran convenciones adecuadas para los nombres de las variables y para los distintos tipos de comentarios. La documentación presenta entre 6 y 10 errores de ortografía.	La aplicación permite el ingreso de todos los operadores lógicos mediante una interfaz gráfica básica. La aplicación permite al usuario ingresar una fórmula proposicional compuesta, pero no se valida la correctitud de la formula ingresada. La aplicación determina la satisfacibilidad o insatisfacibilidad de la fórmula proposicional. La aplicación no presenta un modelo cuando la formula proposicional es satisfacible. La aplicación no muestra en la interfaz el árbol de formación producto de aplicar el método de los tableros semánticos. El código fuente de los archivos no se encuentra documentado.

Tabla 7.5

Bibliografía

[ACM & IEEE, 2004a] ACM, & IEEE. (2004a). Computer Engineering.

[ACM & IEEE, 2004b] ACM, & IEEE. (2004b). Software Engineering.

[ACM & IEEE, 2008] ACM, & IEEE. (2008). Computer Science Curriculum.

[Alonso & Borrero, 2003] Alonso, J., & Borrero, J. (2003). *Deducción automática (Construcción lógica de sistemas lógicos)*. Editorial Kronos.

[Ben-Ari, 2012] Ben-Ari, M. (2012). *Mathematical logic for computer science*. Rehovot: Springer-Verlag London.

[Baldwin & Scragg, (2004)] Baldwin, D, Scragg, G. (2004). *Algorithms and Data Structures: The Science of Computing*. Charles River Media.

[Beth, 1955] Beth, E. (1955). *Semantic entailment and formal derivability.* Mededelingen Koninklijke Nederlandse Akademie van Wetenschappen, 18(13), 309–432.

[Blanco, Smith & Damián, 2001] Blanco, J., Smith, S., & Damián, B. (2001). *Calculo de programas*. Universidad Nacional de Córdoba. Facultad de Matemática, Astronomía y Física.

[Caicedo, 1990] Caicedo, X. (1990). *Elementos de Lógica y Calculabilidad*. Colombia - Bogotá DC, Una Empresa Docente, U. de los Andes.

[Cardona, 2012] Cardona,
S. (2012). *Ambiente Virtual de Aprendizaje de Lógica Matemática*. Retrieved June 15, 2013, from Página de Ambiente Virtual de Aprendizaje de Lógica Matemática

[Cardona, 2010] Cardona, S, Hernández, L & Jaramillo, S. (2010). *Lógica matemática para Ingeniería de Sistemas y Computación.* Ediciones Elizcom. Armenia - Colombia.

[Cardona, Jaramillo, S & Carmona, P, 2010] Cardona, S, Jaramillo, S & Carmona, P. (2007). *Análisis de Algoritmos en Java*. ISBN: 978-958-44-2384-9.

[Castel de Haro, 2011] Castel de Haro, M. J. (2011). *Razonar con "Lógica", con lógica formal de primer orden*. Alicante: Universidad de Alicante. Tomado de `http://hdl.handle.net/10045/23253`

[Cubero & Berzal, 2014] Cubero. J, Berzal. F. (2014) *Tutorial de PROLOG.* Departamento de ciencias de la computación. Universidad de Granada, España. http://elvex.ugr.es/decsai/intelligent/workbook/ai/PROLOG.pdf, consultado en 11-2014.

[deLedesma, 2010] DeLedesma, L. (2010). *Lógica para la Computación.* Editorial Ra-Ma - Alfaomega. Mexico DF.

[Gentzen, 1934] Gentzen, G. (2003). *Untersuchungen Über Das Logische Schliessen (Investigaciones Sobre La Deducción Lógica).* Mathematische Zeitschrift, 34

[González, 2013] González, F. (2005). *Apuntes de Lógica Matemática - 2. Lógica de Predicados.* Universidad de Cádiz - Escuela superior de ingeniería - Departamento de Matemáticas, Cádiz - España

[Kilpatrick, 1918] Kilpatrick, W. H. (1918). *The Project Method.*. Teachers College, 19, 319–335.

[Korfhage, 1970] Korfhage, R. (1970). *Lógica y Algoritmos.* México - México Df, Limusa - Wiley S.A

[Labra, 1998] Labra, J., & Fernández, A. (1998). *Lógica Proposicional para Informática.* Oviedo.

[Muñoz, 2011] Muñoz, C. (2011). *Introducción a la Lógica y al Cálculo de Deducción Natural.* Retrieved from www.ucm.es

[Nepomuceno, 2002] Nepomuceno, A. (2002). *Enseñar lógica: lenguaje lógico y árboles semánticos.* Salamanca. Retrieved from http://logicae.usal.es

[Obeso, 2010] Obeso, V & Burgos, H. (2007). *Lógica matemática - Notas de clase.* Ediciones Uninorte. Bogotá - Colombia.

[Oostra, 2008] Oostra, A. (2008). *Una reseña de la lógica matemática.* Revista Universidad EAFIT, 44(150), 9–20.

[Ospina, 2010] Ospina, L. (2010). *Introducción sistémica al pensamiento lógico matemático* Universidad del Quindío. Armenia - Colombia.

[Páez, 2007] Páez, A. (2007). *Introducción a la lógica moderna.* Ediciones UNIANDES, Universidad de los Andes - Facultad de Ciencias Sociales - Departamento de Filosofía, Bogotá DC.

[Tobón, 2013] Tobón, S. (2013). *Formación integral y competencias (Tercera Ed.).* Bogotá: ECOE Ediciones.

[Robinson, 1965] Robinson, J. (1965). *A Machine-Oriented Logic Based on the Resolution Principle.* Journal of the ACM (JACM), 12(1), 23–41.

[Zarate, 2003] Zarate, Os. (2003). *Introducción a la Lógica.* Lima: UNMSN Fondo editorial.