Matemáticas para la computación

JOSÉ A. JIMÉNEZ MURILLO

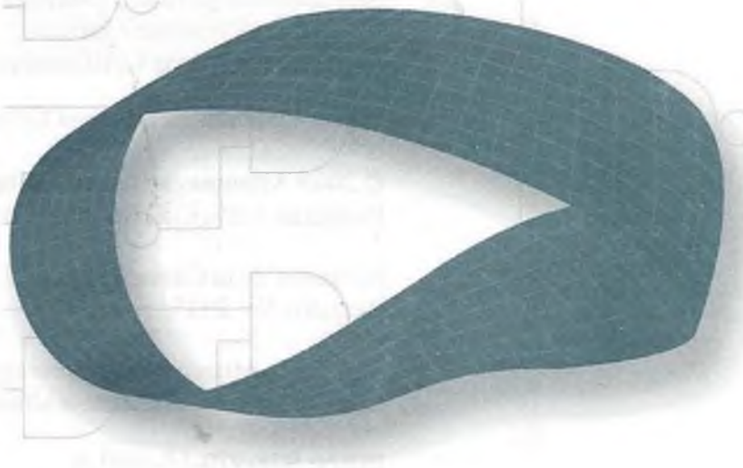
Apoyo en la web

Alfaomega

Matemáticas para la computación

Matemáticas para la computación

JOSÉ ALFREDO
JIMÉNEZ MURILLO



Alfaomega

Francisco Javier Rodríguez Cruz
jrodriguez@alfaomega.com.mx

Gerente editorial
Marcelo Grillo Giannetto
mgrillo@alfaomega.com.mx

Datos catalográficos

Jiménez, José A.
Matemáticas para la Computación
Primera Edición
Alfaomega Grupo Editor, S.A. de C.V., México
ISBN: 978-970-15-1401-6
Formato: 21 x 24 cm Páginas: 496

Matemáticas para la Computación
José Alfredo Jiménez Murillo

Primera edición: Alfaomega Grupo Editor, México, diciembre 2008

© 2009 Alfaomega Grupo Editor, S.A. de C.V.
Pitágoras 1139, Col. Del Valle, 03100, México D.F.

Miembro de la Cámara Nacional de la Industria Editorial Mexicana
Registro No. 2317

Pág. Web: **http://www.alfaomega.com.mx**
E-mail: **atencionalcliente@alfaomega.com.mx**

ISBN: 978-970-15-1401-6

Nota importante:
La información contenida en esta obra tiene un fin exclusivamente didáctico y, por lo tanto, no está previsto su aprovechamiento a nivel profesional o industrial. Las indicaciones técnicas y programas incluidos, han sido elaborados con gran cuidado por el autor y reproducidos bajo estrictas normas de control. ALFAOMEGA GRUPO EDITOR, S.A. de C.V. no será jurídicamente responsable por: errores u omisiones; daños y perjuicios que se pudieran atribuir al uso de la información comprendida en este libro, ni por la utilización indebida que pudiera dársele.

Edición autorizada para venta en todo el mundo.

Impreso en México. Printed in Mexico.

Empresas del grupo:

México: Alfaomega Grupo Editor, S.A. de C.V. – Pitágoras 1139, Col. Del Valle, México, D.F. – C.P. 03100.
Tel.: (52-55) 5089-7740 – Fax: (52-55) 5575-2420 / 2490. Sin costo: 01-800-020-4396
E-mail: atencionalcliente@alfaomega.com.mx

Colombia: Alfaomega Colombiana S.A. – Carrera 15 No. 64 A 29 – PBX (57-1) 2100122, Bogotá, Colombia,
Fax: (57-1) 6068648 – E-mail: scliente@alfaomega.com.co

Chile: Alfaomega Grupo Editor, S.A. – General del Canto 370-Providencia, Santiago, Chile
Tel.: (56-2) 235-4248 – Fax: (56-2) 235-5786 – E-mail: agechile@alfaomega.cl

Argentina: Alfaomega Grupo Editor Argentino, S.A. – Paraguay 1307 P.B. "11", Buenos Aires, Argentina,
C.P. 1057 – Tel.: (54-11) 4811-7183 / 8352, E-mail: ventas@alfaomegaeditor.com.ar

Contenido

Capítulo IV Lógica matemática 114

Capítulo V Álgebra booleana 176

Capítulo VII Grafos 284

Capítulo VIII Árboles 350

Contenido de la página Web de apoyo

El material marcado con asterisco (*) sólo está disponible para docentes.

Capítulo 1. Sistemas numéricos

Diagrama de flujo.

Simulador:

- Herramienta interactiva para hacer conversiones y operaciones entre sistemas numéricos.

Hoja de cálculo:

- Aplicación para convertir cantidades entre diferentes sistemas numéricos.

Software:

- Programa para hacer operaciones aritméticas básicas en diferentes sitemas numéricos.

Respuesta y solución de problemas seleccionados.

Autoevaluación.

Lecturas adicionales (41 págs.):

- Suma de dos cantidades en complemento a 2.
- Más ejemplos de operaciones aritméticas en diferentes sistemas numéricos.
- Sistemas numéricos y códigos binarios.
- Sistemas numéricos en el México prehispánico.

***Evaluaciones propuestas.**

***Presentaciones.**

***Respuesta y solución de ejercicios.**

Capítulo 2. Métodos de conteo

Diagrama de flujo.
Simulador:

- Aplicación que permite calcular el número de combinaciones y/o permutaciones de n elementos en arreglos de tamaño r.

Respuesta y solución de problemas seleccionados.
Autoevaluación.
***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 3. Conjuntos

Diagrama de flujo.
Simulador:

- Herramienta interactiva para hacer operaciones entre conjuntos.

Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lectura adicional (30 págs.):

- Teoría de conjuntos y filosofía.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 4. Lógica matemática

Diagrama de flujo.
Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lecturas adicionales (28 págs.):

- La obra de Gödel en lógica matemática y teoría de conjuntos.
- Sobre la inducción matemática.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 5. Álgebra booleana

Diagrama de flujo.
Simulador:

- Herramienta interactiva para hacer el diagrama lógico de una expresión booleana.

Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lecturas adicionales (72 págs.):

- Aplicación de álgebra booleana a circuitos de conmutación.
- Más aplicaciones del álgebra booleana.
- Álgebra booleana.
- Método de reducción de mapas de Karnaugh.
- Compuertas lógicas.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 6. Relaciones

Diagrama de flujo.
Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lecturas adicionales (104 págs.):

- Relaciones como listas enlazadas.
- Funciones para localizar información.
- Relaciones y funciones.

- Relaciones (1).
- Relaciones (2).
- Relaciones.
- Apuntes de matemática discreta. Relaciones.
- Entre la epistemología y la lógica. Decisión y algoritmos de ordenación.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 7. Grafos

Diagrama de flujo.
Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lecturas adicionales (56 págs.):

- Solución al problema de "locura instantánea" usando grafos.
- Circuitos de Euler y Hamilton.
- Coloración.
- Introducción a la teoría de grafos.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 8. Árboles

Diagrama de flujo.
Simulador:

- Simulador de árboles binarios.

Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lecturas adicionales (65 págs.):

- Árboles B.
- Árboles AVL.

- Evaluación de expresiones matemáticas por medio de árboles.
- Fundamentos de inteligencia artificial.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Capítulo 9. Introducción a los lenguajes formales

Diagrama de flujo.
Respuesta y solución de problemas seleccionados.
Autoevaluación.
Lecturas adicionales (313 págs.):

- Manipulación de cadenas y lenguajes.
- Lenguajes formales.
- Lenguajes formales. Sistemas informáticos.
- Lenguajes y autómatas.
- Gramáticas y lenguajes libres de contexto.
- Máquinas de Turing. Lenguajes.
- Autómatas de pila y lenguajes independientes del contexto.
- Gramáticas regulares. Expresiones regulares.
- Lenguajes regulares.

***Evaluaciones propuestas.**
***Presentaciones.**
***Respuesta y solución de ejercicios.**

Página Web de apoyo

Para tener acceso al material de la página Web de apoyo de **Matemáticas para la computación**:

1) Ir a la página

http://virtual.alfaomega.com.mx

2) Registrarse como usuario del sitio.
3) Ingresar al apartado de inscripción de libros y registrar la siguiente clave de acceso

AN87R7

4) Para navegar en la plataforma virtual de recursos del libro, usar los nombres de *Usuario* y *Contraseña* definidos en el punto número dos.

Prefacio

Las matemáticas siempre han sido una de las disciplinas que le cuesta más trabajo entender a los estudiantes. Si se observan las estadísticas de reprobación en las carreras relacionadas con la computación, junto con las matemáticas la materia de programación es una aduana muy difícil de librar. Sin embargo, las matemáticas y la programación constituyen un campo importante, apasionante y ameno. La computación y las matemáticas tienen gran relación entre sí, sólo hay que recordar que las computadoras fueron creadas inicialmente para realizar operaciones matemáticas con mayor rapidez, además de que la computación no se podría entender sin las matemáticas.

Actualmente la computación es fundamental en todas las actividades que se desarrollan diariamente en la administración, educación, medicina, ingeniería e investigación. El funcionamiento adecuado de una empresa no se podría entender sin la ayuda de la computadora, ¿qué haríamos si se tuviera que regresar al tiempo en que todo se procesaba manualmente?, ¿qué pasaría si no se contara con el correo electrónico, hojas de cálculo, procesadores de texto, lenguajes de programación e internet? Si bien es cierto que todos estos elementos son parte de las acciones que se pueden llevar a cabo en la computadora, también lo es el que las matemáticas proporcionaron el soporte necesario para desarrollar todas estas herramientas computacionales. Ramas de las matemáticas como sistemas numéricos, métodos de conteo, conjuntos, matrices, lógica matemática, álgebra booleana, relaciones y funciones son la base para el diseño de todo lo que se maneja en la computadora. Es por esto que surgen las matemáticas para la computación, mismas que permiten entender el aspecto formal de la relación matemáticas-computadora.

Este libro tiene como objetivo fundamental que los alumnos que cursan alguna carrera relacionada con la computación, aprendan con facilidad los conocimientos matemáticos básicos necesarios para entender el principio matemático usado en la creación de herramientas computacionales. Asimismo se espera que el joven que incursiona en el mundo de la computación tenga una visión más clara de los aspectos que se toman en cuenta para el desarrollo y manejo de estructuras de datos, bases de datos, circuitos electrónicos y lenguajes de programación, no para desarrollar un software al final del curso, pero si para tener una mejor visión de todo aquello que ayudó a desarrollar estas herramientas computacionales que hoy usamos y de las cuales somos muy dependientes.

Para comprender el contenido del libro sólo se requiere la formación básica elemental, ya que se tratan todos los temas con palabras y conceptos que los estudiantes pueden entender, sin descuidar el aspecto formal propio de las matemáticas.

Lo que se pretende en este libro es que el alumno vincule los conocimientos matemáticos con la computación, usando los conceptos con los que los alumnos llegan a la licenciatura. En cada uno de los capítulos se resuelven problemas representativos e ilustrativos, además de que se asignan otros con la finalidad de que el alumno consolide lo aprendido en el aula. Se recomienda al maestro asignar proyectos que tengan relación con el material del libro, para que los alumnos los desarrollen fuera del salón de clases.

Este libro está integrado por nueve capítulos. En el capítulo uno se exponen los sistemas numéricos y tiene como finalidad la representación y operación de cantidades en los sistemas binario, octal y hexadecimal, además de que en él se expone la forma general de realizar operaciones aritméticas básicas en diferentes sistemas numéricos.

En el segundo capítulo se presentan los métodos de conteo, mismos que permiten evaluar y mejorar el software, lo cual es de la mayor importancia en computación ya que se busca desarrollar programas más eficientes y compactos que permitan disminuir el número de iteraciones, comparaciones y ciclos, con el fin de optimizar los recursos. En este capítulo también se exponen los conceptos básicos de combinaciones, permutaciones y binomio de Newton.

En el tercer capítulo se presenta el tema de conjuntos, el cual es la base necesaria para comprender todo el material relacionado con la computación. En este capítulo se establece la relación de los conjuntos con la lógica matemática y el álgebra booleana.

En el cuarto capítulo se aborda la lógica matemática con el fin de que el alumno aprenda a resolver problemas usando como herramientas fundamentales la reflexión, vinculación y el sentido común, pero teniendo como herramientas las reglas de inferencia, equivalencias lógicas y tautologías. En este capítulo también se analiza la validez de proposiciones por medio de métodos deductivos como ocurre con la demostración por el método directo y contradicción, además como una alternativa para probar algoritmos se usa inducción matemática para demostrar si una proposición es cierta.

En el quinto capítulo se expone el álgebra booleana, y aquí la finalidad es que el alumno adquiera las bases necesarias para entender, representar y manejar circuitos electrónicos básicos partiendo de la consideración de que la computadora está integrada con ellos. Se simplifican funciones booleanas por medio de teoremas del álgebra booleana y mapas de Karnaugh, y se

presenta la representación de expresiones booleanas usando bloques lógicos.

En el sexto capítulo se presenta el tema de relaciones y su objetivo es que el estudiante aprenda la representación y manejo de éstas, sabiendo de antemano que las relaciones y las funciones son esenciales en bases de datos, estructuras de datos y programación.

En el séptimo capítulo se expone el tema de grafos y en él se parte del hecho de que éstos son una representación gráfica de las redes de comunicación, incluyendo por su puesto las redes de computadoras. Se abordan circuitos famosos como el circuito de Euler y Hamilton, se manejan problemas en donde las aristas tienen pesos, distancias o costos, como ocurre con el algoritmo de Dijktra, que permite eliminar aristas costosas, se tratan temas como coloración de grafos planos y se utilizan los grafos de similaridad como una forma de discriminar información con características semejantes como ocurre en el reconocimiento de patrones.

En el capítulo ocho se presenta el tema de árboles, los cuales son grafos no dirigidos conexos, sin ciclos ni lazos, que permiten la estructuración de los datos necesaria en la computación para acceder de manera más rápida y eficiente a la información, así como la evaluación de expresiones matemáticas y la compactación de información como ocurre con el código de Fuman. Además en este capítulo se aborda el recorrido de árboles y la búsqueda de información a lo ancho y en profundidad.

En el capítulo nueve se presenta una introducción a los lenguajes formales, con la finalidad de que el alumno adquiera las bases necesarias para comprender asignaturas posteriores que tienen relación directa con matemáticas para la computación. En términos más específicos, en este capítulo se exponen los conceptos fundamentales de gramáticas, lenguajes regulares, árboles de derivación, autómatas finitos determinísticos y no determinísticos, máquinas de estado finito, representación BNF y máquinas de Turing.

Además del contenido descrito, la página Web de apoyo del libro contiene

- Diagramas de flujo.
- Simuladores.
- Hojas de cálculo.
- Software.
- Videos explicativos.
- Respuesta y solución de problemas seleccionados.
- Autoevaluaciones.

- Lecturas adicionales.
- Vínculos de interés.

Como parte de los recursos exclusivos para docentes, la página Web de apoyo incluye

- Evaluaciones propuestas.
- Presentaciones.
- Respuesta y desarrollo de todos los problemas del libro.

Respectivamente los objetivos generales del material de apoyo son los siguientes:

- Ampliar la exposición teórica del libro a través de las 700 páginas de las **Lecturas adicionales**.
- Proporcionar al alumno recursos didácticos que le permitan organizar conceptualmente la teoría y autoevaluar su aprendizaje mediante los **Diagramas de flujo** y las **Autoevaluaciones**.
- Fortalecer la sección de **Respuestas de problemas seleccionados** del libro con el recurso de **Respuesta y solución de problemas seleccionados**, para alentar la confianza del alumno en el proceso de la solución de problemas.
- Mostrar al alumno formas de aplicación de la teoría mediante los **Simuladores**, **Hojas de cálculo** y **Software** incluidos.
- Proporcionar al docente recursos (**Evaluaciones propuestas**, **Presentaciones** y **Respuesta y desarrollo de todos los problemas del libro**) que le sean útiles en la exposición y evaluación del curso.

De acuerdo con estos objetivos, el material de la página Web es parte esencial del libro por lo que luego del Contenido se ha incluido el Contenido de la página Web que es una relación completa de los recursos con que se cuenta en cada capítulo.

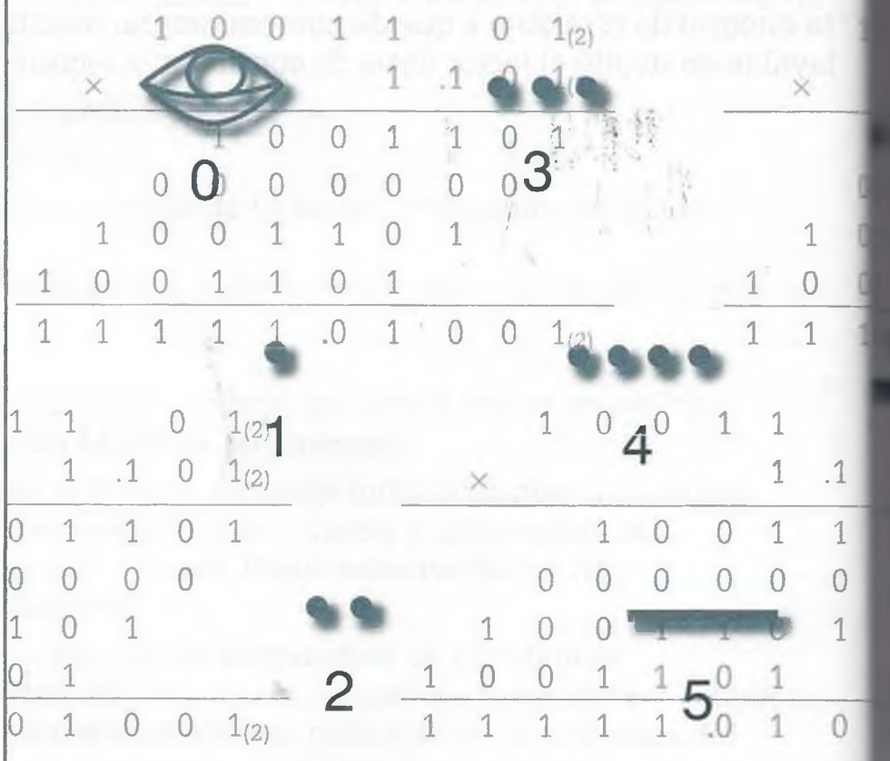
Las matemáticas discretas son uno de los pilares fundamentales de las ciencias de la computación, por esto todo el que se inicia en el estudio de este conocimiento debe de adquirir un dominio amplio de las mismas ya que desde su creación hasta su implementación cualquier elemento computacional requiere de un dominio matemático amplio.

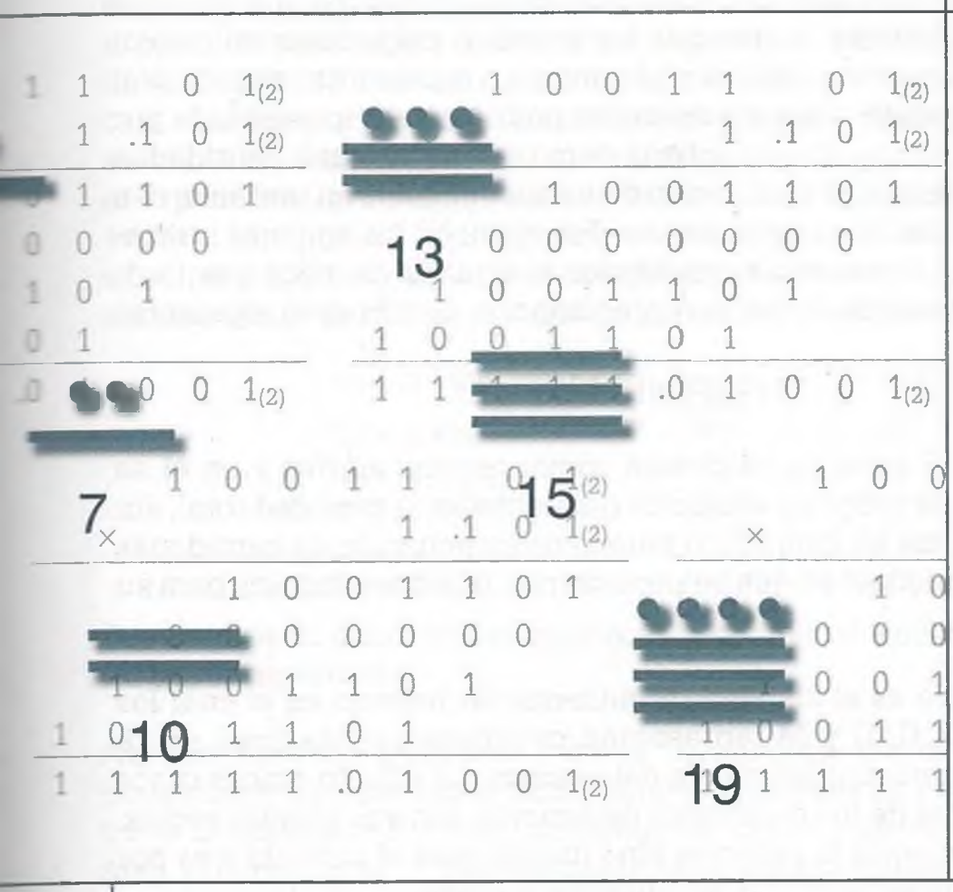
Este importante objetivo de largo plazo es el que ha servido de directriz en la creación de la página Web de este libro, desde la selección de los contenidos del material hasta el diseño relacionado con la distribución de los diferentes tipos de recursos, obteniéndose al final un sitio que es parte integral de esta obra y que de manera natural resulta ser el lugar insoslayable en donde el lector debe de continuar la lectura.

CAPÍTULO

1

Sistemas numéricos





Dios creo los números naturales,
el resto lo hizo el hombre.

Leopold Kronecker

Objetivos

- Representar cantidades en cualquier sistema numérico, incluyendo los sistemas binario, octal y hexadecimal.
- Realizar las operaciones aritméticas básicas en diferentes sistemas numéricos, incluyendo los sistemas binario, octal y hexadecimal.
- Sumar dos cantidades en complemento a 2 con la finalidad de comprender la forma en que la computadora lleva a cabo operaciones aritméticas.

1.1 Introducción

Expresión de un número en un sistema posicional

La expresión general de un número N en un sistema de numeración posicional de base b es de la forma

$$N = d_n d_{n-1} \cdots d_1 d_0, d_{-1} d_{-2} \cdots d_{-k}$$

$$= d_n b^n + d_{n-1} b^{n-1} + \cdots + d_1 b^1 + d_0 b^0 + d_{-1} b^{-1} + \cdots + d_{-k} b^{-k}$$

$$= \sum_{i=-k}^{n} d_i b^i$$

donde d_i es uno de los símbolos definidos en el sistema de numeración, b es la base del sistema de numeración, n es el número de dígitos de la parte entera del número y k es el número de dígitos de su parte fraccionaria.

Dependiendo del valor de b, entre los sistemas posicionales se encuentran el sistema decimal ($b = 10$), el sistema binario ($b = 2$), el sistema octal ($b = 8$), el sistema hexadecimal ($b = 16$).

De acuerdo con la historia se cree que los primeros pobladores utilizaban rayas, círculos, figuras de animales u objetos para representar cantidades. Por ejemplo, una manada de siete animales podría estar representada por siete rayas o siete figuras de ese animal, pero para representar cantidades cada vez mayores se usó la agrupación de varios símbolos en uno solo, con la finalidad de compactar la información. Por ejemplo, los egipcios utilizaban símbolos para representar cantidades y algunos de ellos son | = 1, ∩ = 10, ? = 100; utilizando éstos, la representación de 134 es la siguiente:

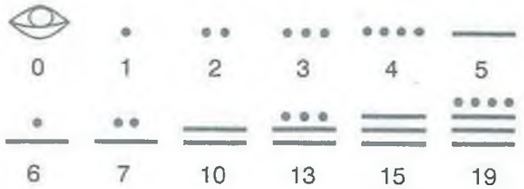
$$? \cap\cap\cap |||| = 134$$

Un sistema como el anterior se conoce como *sistema aditivo* y en él se suman los valores de todos los símbolos para obtener la cantidad total, sin embargo este sistema es impráctico para la representación de cantidades grandes o muy pequeñas, ya que se necesitarían muchos símbolos para su representación.

Otro sistema aditivo es el sistema de numeración romano en el cual los símbolos I, V, X, L, C, D y M representan cantidades y una línea sobre el símbolo implica una multiplicación del número por mil. En ambos casos se suman los valores de los caracteres de acuerdo con sus propias reglas, pero en éstas no importa la posición sino únicamente el símbolo y es por eso que se les llama *sistemas de numeración aditivos*.

Se cree que los babilonios fueron uno de los primeros pueblos en usar un sistema posicional para la representación de cantidades, ya que con base en el movimiento de los astros usaban un sistema sexagesimal (60 caracteres diferentes, en donde cada uno representa un número) para indicar cantidades. Su sistema aún se utiliza para la medición de horas, minutos y segundos, sin embargo tiene problemas con la representación del cero.

Otro sistema posicional es el sistema numérico maya, en el que se estableció un símbolo para representar el número cero, necesario para el buen funcionamiento de todo sistema posicional, con lo cual la cultura maya hizo una aportación valiosa a la ciencia. Este sistema tiene una base de 20, y los 20 símbolos distintos correspondientes se obtienen a partir de la combinación de los que se consideran los tres símbolos básicos para la representación de cantidades. Los siguientes son algunos de los símbolos de este sistema:



Parece que se trata de un sistema numérico aditivo, ya que se suman las rayas y los puntos para obtener los diferentes símbolos utilizados, sin embargo a partir del 20 se utilizan los diferentes símbolos considerando la posición que ocupan, de forma que al multiplicar el símbolo por potencias de 20 (según su posición) y sumar los resultados parciales se obtiene la cantidad a representar. Se puede notar que para representar cantidades se coloca un símbolo encima del otro, asignando respectivamente a la base el exponente 0 para el que está en la parte más baja, al que le sigue hacia arriba el exponente 1 y así sucesivamente. Como se muestra a continuación, el número que corresponde a las siguientes representaciones es el que se obtiene luego de sumar el valor de los símbolos utilizados:

Como se ve en esta representación, la posición del símbolo utilizado juega un papel importante.

Actualmente los sistemas para la representación de cantidades son posicionales, ya que éstos tienen muchas ventajas en relación con los aditivos. Ejemplos de sistemas posicionales son los sistemas numéricos decimal, binario, octal y hexadecimal. Una característica de los sistemas posicionales es que el valor del símbolo lo determina la posición que ocupa y la base del sistema, que es la cantidad de símbolos diferentes usados en un sistema numérico. En este libro se trata preferentemente la representación, conversión y operaciones aritméticas en los sistemas decimal, binario, octal y hexadecimal, pero es importante mencionar que el procedimiento de representación, conversión y operación es el mismo, independientemente del sistema numérico de que se trate.

1.2 Sistema decimal

El sistema decimal se usa en forma rutinaria para la representación de cantidades mediante los siguientes 10 caracteres diferentes:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9

Con estas cifras se pueden expresar cantidades hasta el 9. Para expresar cantidades más allá de este número es necesario introducir la representación posicional, es decir, a cada cifra se le asigna un valor posicional determinado de acuerdo con el lugar que ocupa dentro del número. Por ejemplo: el número decimal 836.74 se compone en la parte entera de la cifra 8 con el valor posicional 100, la cifra 3 con el valor posicional 10 y la cifra 6 con el valor posicional 1, y en la parte fraccionaria de la cifra 7 con

el valor posicional 0.1 y la cifra 4 con el valor posicional 0.01. Así se tiene que:

$$836.74 = 8 \times 100 + 3 \times 10 + 6 \times 1 + \frac{7}{10} + \frac{4}{100}$$

Usando exponentes esto se puede representar como:

$$836.74 = 8 \times 10^2 + 3 \times 10^1 + 6 \times 10^0 + 7 \times 10^{-1} + 4 \times 10^{-2}$$

A esta forma de representación se le llama *representación exponencial.*

La representación exponencial es especialmente importante porque por medio de ella se puede convertir una cantidad representada en cualquier sistema numérico al sistema decimal, como se estudiará más adelante. El valor de la posición lo determina el exponente en una sucesión ascendente de derecha a izquierda para los enteros a partir del punto decimal (el 6 se encuentra en la posición 0, el 3 en la posición 1 y el 8 en la posición 2), y de izquierda a derecha para la parte fraccionaria (el 7 está en la posición –1 y el 4 en la posición –2), usando como base el número 10, debido a que se está en el sistema decimal. También se dice que la base de este sistema aritmético es 10, tomando en cuenta los 10 símbolos disponibles para representar cantidades.

Sistema decimal

Desde el punto de vista matemático, el sistema decimal no posee ninguna ventaja especial sobre cualesquier otro posible sistema de numeración y su uso generalizado se debe a razones completamente ajenas a las leyes generales de las matemáticas.

De acuerdo con la antropología, el origen del sistema decimal se encuentra en el hecho de que los seres humanos tenemos diez dedos en las manos.

1.3 Sistemas binario, octal y hexadecimal

1.3.1 Sistema binario

En el sistema binario sólo hay dos cifras: 0 y 1. Como sucede en el sistema decimal, en este sistema binario también se utilizan exponentes para expresar cantidades mayores. Mientras que en el sistema decimal la base es 10, en el sistema binario la base es 2.

Como se mencionó anteriormente, la representación exponencial se utiliza para convertir una cantidad de un sistema numérico cualquiera al sistema decimal. A continuación se muestra la forma de hacer esto.

Ejemplo 1.1. Convertir el número binario 10011.01 a decimal.

Solución. Expresando el número propuesto en notación exponencial y realizando las operaciones correspondientes, se obtiene la siguiente conversión de binario a decimal:

$$10011.01_{(2)} = 1 \times 2^4 + 0 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 1 \times 2^0 + 0 \times 2^{-1} + 1 \times 2^{-2} = 16 + 0 + 0 + 2 + 1 + 0 + 0.25 = 19.25_{(10)}$$

Sistema binario

El antiguo matemático indio Pingala presentó la primera descripción que se conoce de un sistema de numeración binario en el siglo III a. de C., lo cual coincidió con su descubrimiento del concepto del número cero.

El sistema binario moderno fue documentado en su totalidad por Leibniz, en el siglo XVII, en su artículo "*Explication de l'Arithmétique Binaire*". Leibniz usó el 0 y el 1, al igual que el sistema de numeración binario actual.

Como el 0 y el 1 son caracteres válidos en el sistema decimal y en otros sistemas de mayor base, de aquí en adelante se indicará el sistema en que se encuentra un número expresando su base como un subíndice entre paréntesis, como se hizo en el ejemplo anterior en el que la cantidad binaria está indicada como $10011.01_{(2)}$.

Toda cantidad multiplicada por cero es cero, como se mostró en el caso anterior, sin embargo a partir de ahora esto será suprimido.

Si se desea convertir una cantidad que tiene una parte entera y otra fraccionaria de base diez a base dos, la parte entera se divide sucesivamente entre 2 y los restos resultantes se toman en orden contrario a como se encontraron. La parte fraccionaria se multiplica por 2 y el entero del resultado conforma la parte fraccionaria en el orden en que fueron encontrados. Este procedimiento se ilustra en el siguiente ejemplo.

Gottfried Wilhelm von Leibniz
(1646-1716)

Fue un filósofo, matemático, jurista y político alemán que nació en Leipzig y que en el área de las matemáticas descubrió el cálculo infinitesimal, independientemente de Newton, e inventó el sistema de numeración binario en que se basan casi todas las arquitecturas de computación actuales.

Ejemplo 1.2. Convertir el número $28.37_{(10)}$ a binario.

Solución. Parte entera:

	Resto	
$28 / 2 = 14$	0	↑
$14 / 2 = 7$	0	*Los restos se toman en orden inverso a como fueron encontrados.*
$7 / 2 = 3$	1	
$3 / 2 = 1$	1	
$1 / 2 = 0$	1	

Parte fraccionaria:

	Entero	
$0.37 \times 2 = 0.74$	0	↓
$0.74 \times 2 = 1.48$	1	*Los enteros se toman en el mismo orden en que fueron encontrados.*
$0.48 \times 2 = 0.96$	0	
$0.96 \times 2 = 1.92$	1	
$0.92 \times 2 = 1.84$		

Se podría seguir aproximando para determinar más dígitos en la parte fraccionaria y obtener así un resultado más exacto, sin embargo para ilustrar el procedimiento es suficiente con cuatro dígitos después del punto que separa a la parte entera de la parte fraccionaria. De esta forma, el resultado es:

$$28.37_{(10)} = 11100.0101_{(2)}$$

1.3.2 Sistema octal

Sistema octal

El sistema de numeración octal usa 8 dígitos (0, 1, 2, 3, 4, 5, 6, 7) que tienen el mismo valor que en el sistema de numeración decimal.

Este sistema es muy usado en la computación por tener una base que es potencia exacta de 2, además de que esta característica hace que la conversión a binario o viceversa sea bastante simple.

Por otro lado, este sistema es utilizado como una forma abreviada de representar números binarios que emplean caracteres de seis bits; cada tres bits (medio carácter) es convertido en un único dígito octal.

La reglas descritas para los sistemas decimal y binario, también son aplicables al sistema octal. En los siguientes ejemplos se ilustra este planteamiento.

Ejemplo 1.3. Convertir $631.532_{(8)}$ a binario.

Solución. Primero se convierte el número dado a decimal y luego de decimal a binario.

Para convertir una cantidad de cualquier sistema numérico a decimal, se plantea su representación en notación exponencial y se realizan las operaciones. En este caso particular se tiene que:

$$631.532_{(8)} = 6 \times 8^2 + 3 \times 8^1 + 1 \times 8^0 + 5 \times 8^{-1} + 3 \times 8^{-2} + 2 \times 8^{-3} = 409.6758_{(10)}$$

La conversión del número obtenido a binario es la siguiente:

Parte entera	*Resto*	Parte fraccionaria	*Entero*
409/2 = 204	1		
204/2 = 102	0	0.6758 × 2 = 1.3516	1
102/2 = 51	0	0.3516 × 2 = 0.7032	0
51/2 = 25	1	0.7032 × 2 = 1.4064	1
25/2 = 12	1	0.4064 × 2 = 0.8128	0
12/2 = 6	0		
6/2 = 3	0		
3/2 = 1	1		
1/2 = 0	1		

La conversión de octal a binario y de binario a octal es relativamente fácil si se utiliza la siguiente tabla de equivalencias*:

Octal	Binario
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

*Se puede apreciar que se utilizan tres dígitos en binario, para cada número en octal, debido a que la cantidad mayor válida en el sistema octal es el número 7, que ocupa tres bits, por lo tanto, todos deberán usar la misma cantidad de bits.

Ejemplo 1.4. Convertir $631.532_{(8)}$ a binario usando la tabla de equivalencias anterior.

Solución. En la siguiente tabla se presenta la conversión pedida:

6	3	1	.	5	3	$2_{(8)}$
110	011	001	.	101	011	$010_{(2)}$

Como se puede observar en el ejemplo 1.4, usando la tabla de equivalencias el resultado es igual al obtenido al aplicar el método general usado en el ejemplo 1.3, sin embargo se debe mencionar que cuando se usa el método general algunas veces existen diferencias en los resultados, ya que al convertir la cantidad de octal a decimal y posteriormente a binario se pierde exactitud por el redondeo y también debido a que se acordó anteriormente encontrar solamente los primeros cuatro dígitos en la parte fraccionaria. A pesar de esto, si existiera alguna diferencia ésta se presentaría en la parte fraccionaria y no en la parte entera.

Ejemplo 1.5. Convertir $1101010000011110101101 0.0001101_{(2)}$ a octal usando tablas y verificar dicho resultado usando el método general.

Solución. Cuando se usan tablas, se deben separar los bits de la cantidad binaria en bloques de tres en tres, a partir del punto decimal hacia la izquierda en la parte entera, y del punto decimal a la derecha en la parte fraccionaria. Si los bloques no se completan, se agregan ceros en los extremos como se indica a continuación:

011	010	100	000	111	101	011	010	.	000	110	$100_{(2)}$
3	2	4	0	7	5	3	2	.	0	6	$4_{(8)}$

Para verificar este resultado usando el método general, primero se convierte de binario a decimal de forma que:

$1101010000011110101101 0.0001101_{(2)} = 1 \times 2^{22} + 1 \times 2^{21} + 1 \times 2^{19} + 1 \times 2^{17} + 1 \times 2^{11} + 1 \times 2^{10} + 1 \times 2^{9} + 1 \times 2^{8} + 1 \times 2^{6} + 1 \times 2^{4} + 1 \times 2^{3} + 1 \times 2^{1} + 1 \times 2^{-4} + 1 \times 2^{-5}\ 1 \times 2^{-7} = 4194304 + 2097152 + 524288 + 131072 + 2048 + 1024 + 512 + 256 + 64 + 16 + 8 + 2 + 0.0625 + 0.0312 + 0.0078 = 6950746.1015_{(10)}$

A continuación se convierte de decimal a octal:

Parte entera	*Resto*	Parte fraccionaria	*Entero*
6950746/8 = 868843	2		
868843/8 = 108605	3	0.1015 × 8 = 0.8120	0
108605/8 = 13575	5	0.8120 × 8 = 0.4960	6
13575/8 = 1696	7	0.4960 × 8 = 0.9680	3
1696/8 = 212	0	0.9680 × 8 = 0.7440	7
212/8 = 26	4		
26/8 = 3	2		
3/8 = 0	3		

Al comparar los resultados obtenidos por el método general y mediante la tabla de equivalencias, se observa que la parte entera coincide totalmente y que solamente existe una pequeña diferencia en la parte fraccionaria, debido a errores de redondeo.

Es importante tener cuidado cuando se encuentra el resto de una división. En el caso de este ejemplo se dividió 108605/8 = 13575.625, y el resto resultó de multiplicar la parte fraccionaria del cociente por la base, esto es, 0.625 × 8 = 5.

Sistema hexadecimal

El uso del sistema hexadecimal está estrechamente relacionado con la informática y con las ciencias de la computación, ya que las computadoras suelen utilizar el byte u octeto como unidad básica de memoria.

1.3.3 Sistema hexadecimal

La base numérica del sistema hexadecimal es 16 y para representar cantidades en él se utilizan los diez dígitos del sistema decimal (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) así como las seis primeras letras del alfabeto (A, B, C, D, E, F). Con esto pueden formarse números según el principio de valor posicional como en los demás sistemas aritméticos. Los caracteres válidos en hexadecimal son del 1 al 15, con la particularidad de que a las letras se les asigna el siguiente valor: A = 10, B = 11, C = 12, D = 13, E = 14 y F = 15.

Ejemplo 1.6. Convertir a $E8A7.3D_{(16)}$ a octal.

Solución. El número dado primero se convierte a decimal:

$$E8A7.3D_{(16)} = 14 \times 16^3 + 8 \times 16^2 + 10 \times 16^1 + 7 \times 16^0 + 3 \times 16^{-1} + 13 \times 16^{-2} = 59559.2383_{(10)}$$

Ahora el número obtenido se convierte a octal:

Parte entera	*Resto*	Parte fraccionaria	*Entero*
59559/8 = 7444	7		
7444/8 = 930	4	0.2383 × 8 = 1.9064	1
930/8 = 116	2	0.9064 × 8 = 7.2512	7
116/8 = 14	4	0.2512 × 8 = 2.0096	2
14/8 = 1	6	0.0096 × 8 = 0.0768	0
1/8 = 0	1		

De igual manera que en la conversión de binario a octal, se puede obtener la siguiente tabla de equivalencias de binario a hexadecimal*:

Hexadecimal	Binario
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001
A	1010
B	1011
C	1100
D	1101
E	1110
F	1111

*Nuevamente el número mayor del sistema numérico es el que manda en relación con cuántos bits se deberán usar para representar cada uno de los caracteres. En este caso F = 15 es el símbolo mayor y ocupa cuatro bits, por lo tanto todos los símbolos deberán representarse por cuatro bits.

Ejemplo 1.7. Convertir $E8A7.3D_{(16)}$ a octal usando tablas de equivalencias.

Solución. A diferencia del método general, en el que el sistema intermedio es el sistema decimal, cuando se utilizan tablas de equivalencias el sistema intermedio es el binario, por lo que primero se pasa la cantidad al sistema binario poniendo los cuatro bits correspondientes a cada uno de los caracteres del sistema hexadecimal:

E	8	A	7	.	3	$D_{(16)}$
1110	1000	1010	0111	.	0011	$1101_{(2)}$

A continuación se pasa de binario a octal, agrupando la información en bloques de tres bits, ya que la tabla de equivalencia octal-binario utiliza solamente tres bits para cada uno de los caracteres. En caso de no completarse los bloques de tres bits, se deberán agregar los ceros necesarios en los extremos:

001	110	100	010	100	111	.	001	111	$010_{(2)}$
1	6	4	2	4	7	.	1	7	$2_{(8)}$

Como se puede observar, el resultado es semejante al obtenido en el ejemplo 1.6 y la variación, en caso de existir, es muy pequeña.

1.4 Generalización de las conversiones

De la misma manera en que fueron creados los sistemas posicionales decimal, binario, octal y hexadecimal, es posible crear nuestro propio sistema usando los dígitos necesarios del 0 al 9, y también en el caso de que se requieran las letras del alfabeto.

Las siguientes cantidades están expresadas en sistemas posicionales inexistentes, pero que podrían ser perfectamente válidos ya que respetan todas las reglas de los sistemas posicionales:

$20541.32_{(7)}$ Aquí la base es 7 y los caracteres válidos van del 0 al 6.

$7G5A90.HB_{(18)}$ En este caso, además de poder usar los dígitos del 0 al 9 es posible utilizar las letras A = 10, B = 11, C = 12, D = 13, E = 14, F = 15, G = 16, H = 17, ya que en base 18 los caracteres válidos van del 0 al 17.

Se puede notar que el número menor siempre es el 0 y que el mayor es el que corresponde a (base –1).

Esas cantidades expresadas en cualquier sistema numérico pueden ser convertidas a otro sistema existente o no, de tal forma que se puede establecer que para pasar de un sistema *X* cualquiera a decimal se representa en notación exponencial, y para pasar de decimal a un sistema *W* cualquiera se divide la parte entera entre la base a la que se desea convertir y la parte fraccionaria se multiplica por la base en cuestión, como se muestra en el diagrama de la figura 1.1.

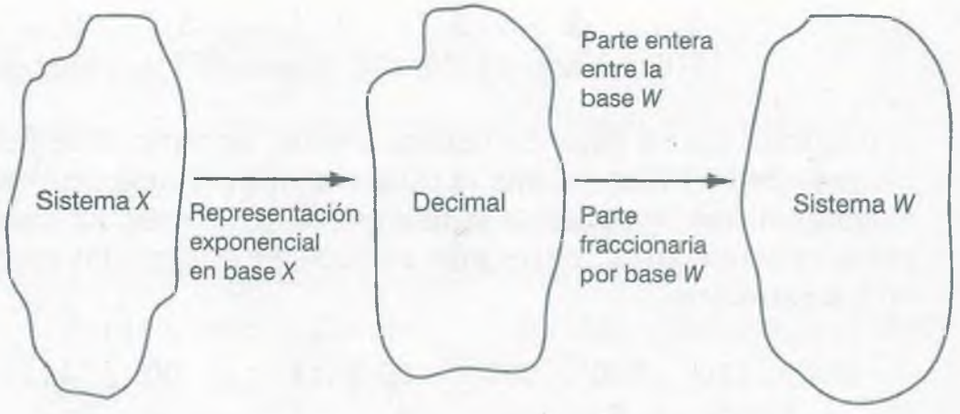


Figura 1.1 Conversión entre sistemas numéricos.

Ejemplo 1.8. Convertir $CD057.EC_{(15)}$ a base 20.

Solución. Hay que observar que en este caso no es posible usar tablas de equivalencias, por lo tanto no queda otra opción que usar el método general y convertir primero a decimal utilizando la representación exponencial y posteriormente a base 20, de acuerdo con el diagrama de la figura 1.1:

$$CD057.EC_{(15)} = 12 \times 15^4 + 13 \times 15^3 + 5 \times 15^1 + 7 \times 15^0 + 14 \times 15^{-1} + 12 \times 15^{-2}$$
$$= 651457.9866_{(10)}$$

A continuación se hace la conversión a base 20:

Parte entera	*Resto*	Parte fraccionaria	*Entero*
651457/20 = 32572	17 ↑	0.9866 × 20 = 19.732	19 ↓
32572/20 = 1628	12	0.732 × 20 = 14.64	14
1628/20 = 81	8	0.64 × 20 = 12.8	12
81/20 = 4	1	0.8 × 20 = 16.0	16
4/20 = 0	4		

Finalmente se puede concluir que:

$$CD057.EC_{(15)} = 418CH.JECG_{(20)}.$$

Aquí se debe de observar que en la base 20 pueden existir caracteres del 0 al 9, y por tanto se tiene que A = 10, B = 11, C = 12, D = 13, E = 14, F = 15, G = 16, H = 17, I = 18, J = 19.

1.5 Operaciones básicas

Las operaciones básicas de suma, resta, multiplicación y división que se realizan en el sistema decimal, también se pueden llevar a cabo en cualquier sistema numérico aplicando las mismas reglas y teniendo en cuenta la base en la que se encuentran los números con los que se efectúa la operación. Es importante observar que las cantidades que se estén operando se deben de encontrar en la misma base, y en caso de no ser así lo primero que se debe de hacer es la conversión correspondiente de cada una de ellas.

Suma, resta y multiplicación

La suma, la resta y la multiplicación de números son ejemplos de operaciones binarias, esto es, operaciones entre pares de números.

En general una operación binaria definida en un conjunto, es una regla que asocia a cada par ordenado de elementos del conjunto algún elemento del mismo conjunto.

A continuación se realizarán operaciones básicas de suma, resta, multiplicación y división en los sistemas decimal, binario, octal y hexadecimal. El sistema decimal permite ilustrar el procedimiento a seguir en cada una de las operaciones aritméticas, gracias a la familiaridad que se tiene con él, y los sistemas binario, octal y hexadecimal son de gran utilidad en el área de la computación.

1.5.1 Suma

Ejemplo 1.9. Suma en el sistema decimal:

$$\begin{array}{rrrrrrrrrr} & & & 4 & 5 & 6 & . & 7 & 8_{(10)} \\ + & 1 & 7 & 8 & 2 & 0 & . & 6 & 4 \quad 9_{(10)} \\ \hline & 1 & 8 & 2 & 7 & 7 & . & 4 & 2 \quad 9_{(10)} \end{array}$$

Explicación por columna:

$0 + 9 = 9$	El 9 es un dígito válido de base 10, por lo que se queda tal cual.
$8 + 4 = 12$	El 12 no es válido en decimal, ya que es una combinación del 1 y el 2. Cuando ocurre esto se deberá dividir entre la base (10), colocando el resto debajo de la línea y sumando el cociente a los números de la siguiente columna de la izquierda.
$1 + 7 + 6 = 14$	El 14 no es válido por lo que se divide entre la base y se procede como se hizo anteriormente.
$1 + 6 + 0 = 7$	Dígito válido en decimal.
$5 + 2 = 7$	Dígito válido.
$4 + 8 = 12$	Aquí hay que dividir entre la base.
$1 + 0 + 7 = 8$	Dígito válido.

Los espacios vacíos de los extremos, como el de arriba del 9 en el ejemplo 1.9, se consideran como 0, por esta razón se sumó 0 + 9. Esto mismo sucede en todos los sistemas numéricos, ya que el 0 es el dígito válido más pequeño. Cuando el resultado de la suma es un símbolo válido en ese sis

tema numérico normalmente no se divide entre la base, pero podría dividirse ya que el cociente es 0 y por lo tanto no afecta a la siguiente columna de la izquierda y el resto es el mismo número, por ejemplo, si el resultado es 6, el cual es válido en el sistema decimal, el cociente de dividir 6 entre 10 es 0 y el resto es 6.

Ejemplo 1.10. Suma en el sistema hexadecimal:

$$\begin{array}{rcccccccc} & A & 6 & F & C & 9 & . & 7 & B & 2_{(16)} \\ + & 4 & E & 7 & D & 0 & . & 7 & 3 & E_{(16)} \\ \hline & F & 5 & 7 & 9 & 9 & . & E & F & 0_{(16)} \end{array}$$

Explicación por columna:

$2 + 14 = 16$	Al dividir 16 entre la base se obtiene el cociente 1 y resto 0.
$1 + 11 + 3 = 15$	Dígito válido: 15 = F.
$7 + 7 = 14$	Dígito válido: 14 = E.
$9 + 0 = 9$	Dígito válido.
$12 + 13 = 25$	Al dividir 25 entre la base se obtiene el cociente 1 y resto 9.
$1 + 15 + 7 = 23$	Al dividir 23 entre la base se obtiene el cociente 1 y resto 7.
$1 + 6 + 14 = 21$	Al dividir 21 entre la base se obtiene el cociente 1 y resto 5.
$1 + 10 + 4 = 15$	Dígito válido: 15 = F

Generalización de la suma

Como se puede observar, el procedimiento para llevar a cabo la suma en los diferentes sistemas numéricos no cambia, sino que sólo hay que tener en cuenta la base en que se realiza la operación.

Por tanto, en general se puede establecer que si al sumar dos dígitos el resultado de la suma sobrepasa al dígito mayor de un sistema numérico determinado, entonces el resultado se debe dividir entre la base del sistema y el residuo de esa división se pone debajo de la línea y el cociente se suma a la columna siguiente izquierda.

Ejemplo 1.11.

$$\begin{array}{r} H\ 7\ 2\ 5\ .\ 4\ A_{(18)} \\ +\ \ G\ B\ D\ .\ 7\ H\ \ 2_{(18)} \\ \hline 1\ 0\ 5\ E\ 0\ .\ C\ 9\ \ 2_{(18)} \end{array} \qquad \begin{array}{r} 2\ 3\ 0\ 3\ 1\ 1\ .\ 2\ 1_{(5)} \\ +\ 3\ 1\ 3\ 2\ 0\ 0\ 3\ .\ 3\ 2\ \ 1_{(5)} \\ \hline 3\ 4\ 1\ 2\ 3\ 2\ 0\ .\ 0\ 3\ \ 1_{(5)} \end{array}$$

$$\begin{array}{r} 9\ 8\ 0\ B\ 3\ .\ 2\ 5_{(13)} \\ +\ \ A\ 7\ 2\ 8\ .\ 0\ 9_{(13)} \\ \hline A\ 5\ 8\ 0\ B\ .\ 3\ 1_{(13)} \end{array} \qquad \begin{array}{r} K\ 0\ J\ 7\ .\ L\ 2_{(23)} \\ +\ 2\ 7\ C\ M\ E\ .\ F\ A_{(23)} \\ \hline 3\ 4\ D\ I\ M\ .\ D\ C_{(23)} \end{array}$$

1.5.2 Resta

Ejemplo 1.12. Resta en el sistema decimal:

$$\begin{array}{r} 8\ 1\ 2\ 7\ .\ 5\ 8\ 0_{(10)} \\ -\ \ 5\ 8\ 3\ 1\ .\ 9\ 6\ 4_{(10)} \\ \hline 2\ 2\ 9\ 5\ .\ 6\ 1\ 6_{(10)} \end{array}$$

Explicación por columna:

$(0 + 10) - 4 = 6$ — Cuando el sustraendo es mayor que el minuendo, como ocurre en la primera columna, se deberá sumar la base al minuendo y después llevar a cabo la sustracción (minuendo + 10) – sustraendo = resultado.

$8 - (6 + 1) = 1$ — Cuando en la columna anterior se sumó 10 al minuendo, en la columna siguiente de la izquierda se deberá sumar 1 al sustraendo (minuendo – (sustraendo + 1)) = resultado. Si después de sumar 1 al sustraendo éste es mayor que el minuendo, entonces se deberá sumar la base al minuendo antes de llevar a cabo la resta.

$(5 + 10) - 9 = 6$ — Debido a que sustraendo > minuendo, se suma la base al minuendo y se realiza la resta.

$7 - (1 + 1) = 5$	Como en la columna anterior se le sumó la base 10 al minuendo, en esta columna se le deberá sumar 1 al sustraendo.
$(2 + 10) - 3 = 9$	Como sustraendo > minuendo, se suma la base al minuendo.
$(1 + 10) - (8 + 1) = 2$	Como se le sumó la base a la columna anterior, se deberá aumentar en 1 el sustraendo y como sustraendo > minuendo, se deberá sumar la base al minuendo antes de llevar a cabo la resta. El orden en que se llevan a cabo los incrementos en este caso es muy importante.
$8 - (5 + 1) = 2$	Sumar 1 al sustraendo, ya que en la columna anterior se le sumó la base al minuendo, y después llevar a cabo la resta.

Al efectuar la resta es necesario revisar si el sustraendo es mayor que el minuendo, ya que en caso afirmativo se debe sumar la base al minuendo antes de llevar a cabo la resta de dos dígitos de una columna cualquiera. Una vez comenzada la operación de resta cuando al minuendo se le suma la base, entonces al sustraendo de la columna izquierda próxima se le deberá sumar 1 (ya que en este caso la base es 10) antes de hacer la comparación entre el minuendo y sustraendo. En el caso de otro sistema numérico, lo que se le suma al minuendo debe de ser la base que corresponda (8 en octal, 16 hexadecimal o 2 en binario), sin embargo cuando se le suma la base al minuendo invariablemente será 1 lo que se incrementa el sustraendo de la columna izquierda próxima, independientemente del sistema numérico de que se trate (ya que ese 1 significa que se le sumó una vez la base en la columna anterior).

Ejemplo 1.13. Resta en el sistema octal:

$$\begin{array}{rcccccccc} & 4 & 1 & 0 & 7 & 2 & . & 1 & 4_{(8)} \\ - & 3 & 6 & 0 & 4 & 3 & . & 7 & 1 \quad 3_{(8)} \\ \hline & 0 & 3 & 0 & 2 & 6 & . & 2 & 2 \quad 5_{(8)} \end{array}$$

Explicación por columna:

$(0 + 8) - 3 = 5$	Como sustraendo > minuendo, hay que sumar la base al minuendo.
$4 - (1 + 1) = 2$	Sumar 1 al sustraendo debido a que se sumó la base al minuendo en la columna anterior.
$(1 + 8) - 7 = 2$	Sumar la base al minuendo, ya que el sustraendo > minuendo.
$(2 + 8) - (3 + 1) = 6$	Primero sumar 1 al sustraendo, ya que se aumentó la base en la columna anterior, después sumar la base al minuendo, ya que sustraendo > minuendo.
$7 - (4 + 1) = 2$	Sumar 1 al sustraendo debido a que se le sumó la base al minuendo en la columna anterior.
$0 - 0 = 0$	Sin cambio, ya que no se cumple que sustraendo > minuendo, ni se sumó la base en la columna anterior.
$(1 + 8) - 6 = 3$	Sumar la base al minuendo, ya que sustraendo > minuendo.
$4 - (3 + 1) = 0$	Sumar 1 al sustraendo, ya que se sumó la base al minuendo de la columna anterior.

Generalización de la resta

En forma general se puede decir que si en la primera columna se cumple la condición sustraendo > minuendo, entonces se deberá sumar la base al minuendo y después se realizará la resta; en caso de que no se cumpla la condición, solamente se hace la resta. A partir de la segunda columna se sumará 1 al sustraendo si en la columna anterior se sumó la base al minuendo (en caso contrario se deja tal cual) y después se observará si sustraendo > minuendo, si esto es verdadero se le agrega la base al minuendo para finalmente llevar a cabo la resta correspondiente en cada una de las columnas hasta terminar.

Ejemplo 1.14.

$$\begin{array}{r} 8\ 5\ A\ C\ 3\ .\ 7\ 0\ E_{(17)} \\ -\quad B\ 5\ C\ 6\ .\ F\ 2_{(17)} \\ \hline 7\ B\ 4\ G\ D\ .\ 8\ F\ E_{(17)} \end{array} \qquad \begin{array}{r} 4\ 0\ 5\ 8\ 2\ 7\ 1\ .\ 6_{(9)} \\ -\ 1\ 5\ 8\ 3\ 5\ 8\ 2\ .\ 2\quad 8\ 4_{(9)} \\ \hline 2\ 3\ 6\ 4\ 5\ 7\ 8\ .\ 3\quad 0\ 5_{(9)} \end{array}$$

$$\begin{array}{r} 2\ 3\ 1\ K\ A\ .\ 4\ 3_{(21)} \\ -\quad 1\ G\ 7\ H\ .\ C\ 2_{(21)} \\ \hline 2\ 1\ 6\ C\ D\ .\ D\ 1_{(21)} \end{array} \qquad \begin{array}{r} 7\ 4\ 1\ 0\ 0\ .\ A\ 5_{(14)} \\ -\quad 5\ C\ 5\ 3\ .\ C\ 2_{(14)} \\ \hline 6\ C\ 2\ 8\ A\ .\ C\ 3_{(14)} \end{array}$$

1.5.3 Multiplicación

La forma en que se multiplica en decimal es la misma en que se llevan a cabo las multiplicaciones en otros sistemas numéricos, la única diferencia es la base.

Ejemplo 1.15. Multiplicación en el sistema decimal:

$$\begin{array}{r} 8\ 0\ 5\ 7\ .\ 2\quad 3_{(10)} \\ \times\qquad 5\ 3\ .\ 7_{(10)} \\ \hline 5\ 6\ 4\ 0\ 0\ 6\quad 1 \\ 2\ 4\ 1\ 7\ 1\ 6\ 9 \\ 4\ 0\ 2\ 8\ 6\ 1\ 5 \\ \hline 4\ 3\ 2\ 6\ 7\ 3\ .2\ 5\quad 1_{(10)} \end{array}$$

Explicación por columna:

$7 \times 3 = 21$ Como el 21 no es un dígito válido en decimal, se divide entre la base para obtener cociente = 2 y resto = 1. El resto se coloca debajo de la línea y el cociente se suma en el producto de la siguiente columna.

$7 \times 2 + 2 = 16$ Como el 16 no es un dígito válido en el sistema decimal, se realiza lo mismo que en el caso anterior para obtener cociente = 1 y resto = 6.

El procedimiento seguido en el sistema decimal es el que se realiza en cualquier sistema numérico, tomando en cuenta que cuando la cantidad resultante no es un dígito válido en dicho sistema entonces se debe dividir entre la base; en octal se debe dividir entre 8, en hexadecimal entre 16, en vigesimal entre 20 y así sucesivamente. La forma en que se suman en el sistema decimal los resultados obtenidos en las diferentes columnas, es la misma en que se suman en otro sistema. Como se muestra en el ejemplo 1.15, en la primera columna se baja el 1 porque no tiene otro dígito con que sumarse, en la segunda columna se suma 6 + 9 = 15 y como el 15 no es un dígito válido en decimal entonces se debe dividir entre la base para obtener cociente = 1 y resto = 5, el resto se pone debajo de la línea y el cociente se suma a los dígitos de la siguiente columna de la izquierda, y así se continúa hasta terminar.

Ejemplo 1.16. Multiplicación en el sistema binario:

$$
\begin{array}{cccccccccc}
 & & 1 & 0 & 0 & 1 & 1 & . & 0 & 1_{(2)} \\
 & \times & & & & & 1 & .1 & 0 & 1_{(2)} \\
\hline
 & & & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\
 & & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \\
 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & & \\
1 & 0 & 0 & 1 & 1 & 0 & 1 & & & \\
\hline
1 & 1 & 1 & 1 & 1 & .0 & 1 & 0 & 0 & 1_{(2)}
\end{array}
$$

Entre menor sea la base del sistema numérico es más sencillo realizar operaciones aritméticas, ya que el número de dígitos válidos también se reduce en la misma proporción. Como se muestra en el ejemplo 1.16, en el sistema binario solamente hay posibilidades de 0 o 1 y esto simplifica el trabajo.

En cualquier sistema, al multiplicar una cantidad por 1 se obtiene la misma cantidad, por esa razón en el sistema binario al multiplicar 1 por el multiplicando resulta el mismo multiplicando y al multiplicar 0 por el multiplicando resulta una fila de ceros. Al sumar las columnas se pueden observar casos como el que ocurre en la tercera columna de derecha a izquierda donde 1 + 0 + 1 = 2, pero como el 2 no es un dígito válido en binario se debe dividir entre la base, obteniéndose cociente = 1 y resto = 0 por lo que el resto se coloca debajo de la línea y el cociente se suma con los dígitos de la siguiente columna, de forma que en la cuarta columna se deben sumar 1 + 1 + 0 + 0 + 1 = 3, donde el primer 1 es el cociente de la línea anterior. Como el 3 no es un dígito válido en binario, se divide entre la base y se

obtiene cociente = 1 y resto = 1. En lo que sigue se procede de la misma manera hasta terminar de sumar todas las columnas. El punto que separa a la parte entera de la fraccionaria se coloca también de manera semejante a como se realiza en el sistema decimal: se cuentan los decimales tanto del multiplicando como del multiplicador.

Como el procedimiento para multiplicar en cualquier sistema es el mismo que el que se utiliza en el sistema decimal, lo único que se debe tener presente es la base en que se está trabajando.

Ejemplo 1.17.

$$
\begin{array}{rrrrrrrrr}
 & 5 & 9 & A & C & . & 1 & 2_{(14)} \\
\times & & & & & B & .8 & 9_{(14)} \\
\hline
 & 3 & 9 & 3 & D & A & A & 4 \\
3 & 3 & 8 & 2 & C & 9 & 2 & \\
4 & 6 & 9 & 7 & 6 & C & 8 & \\
\hline
4 & A & 2 & A & D & .A & D & C & 4_{(14)}
\end{array}
$$

$$
\begin{array}{rrrrrrrrr}
 & A & 5 & 3 & 7 & 5 & . & C_{(13)} \\
\times & & & & 4 & .B & 9 & 8_{(13)} \\
\hline
 & 6 & 5 & 3 & 2 & 7 & 8 & 5 \\
7 & 2 & 8 & 6 & 2 & 1 & 4 & \\
8 & A & 6 & 0 & 4 & 0 & 2 & \\
\hline
3 & 2 & 8 & 1 & 3 & A & 9 & & \\
\hline
3 & C & 0 & 3 & 4 & B & .0 & A & C & 5_{(13)}
\end{array}
$$

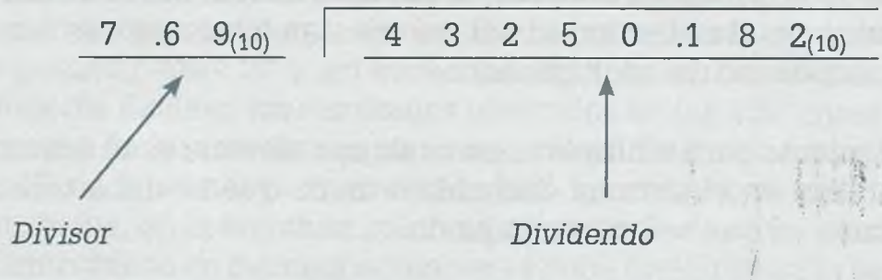
$$
\begin{array}{rrrrrrrrr}
 & C & 5 & 9 & D & A & . & 2_{(15)} \\
\times & & & & 6 & B & .4 & 5_{(15)} \\
\hline
 & 4 & 1 & D & 4 & 8 & 5 & A \\
3 & 4 & 7 & 9 & 9 & A & 8 & \\
9 & 1 & 2 & 4 & 0 & 6 & 7 & \\
\hline
4 & E & 3 & E & 7 & 0 & C & & \\
\hline
5 & 8 & 8 & A & 5 & 9 & 2 & .A & D & A_{(15)}
\end{array}
$$

$$
\begin{array}{rrrrrrrrr}
 & F & 3 & A & C & 7 & . & 3_{(17)} \\
\times & & & & 0 & .5 & D & 4_{(17)} \\
\hline
 & 3 & 9 & E & 8 & F & B & C \\
B & A & D & 3 & 8 & 8 & 5 & \\
4 & 8 & 1 & 2 & B & 1 & F & & \\
\hline
5 & 2 & F & 8 & C & .2 & 4 & G & C_{(17)}
\end{array}
$$

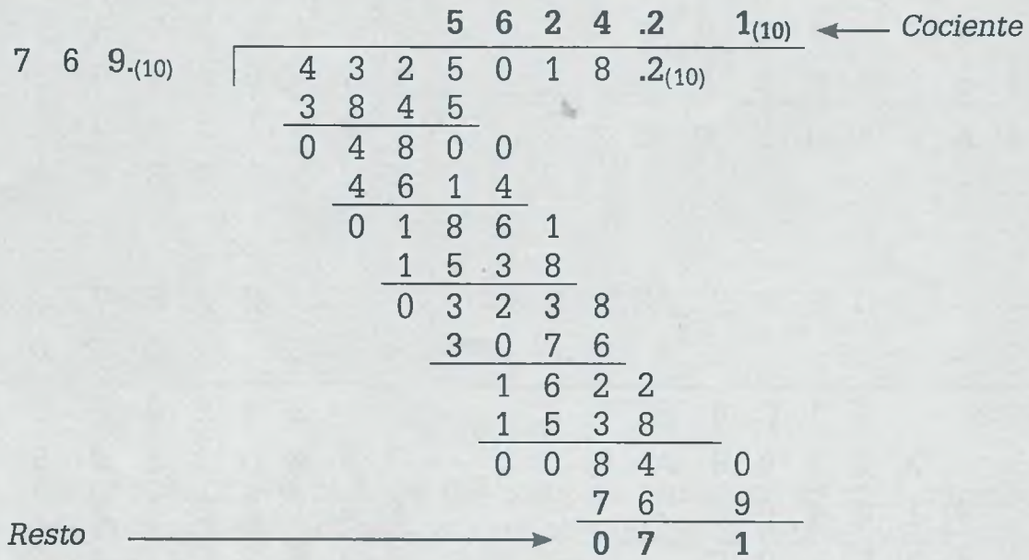
1.5.4 División

Se sabe que la división involucra operaciones de resta y multiplicación, por lo que es más complicada que las tres operaciones aritméticas anteriores. En este caso lo que se recomienda es usar lo que se conoce como división desarrollada, la cual permite realizar primero la multiplicación y después la resta, ya que de otra forma el tratar de llevar a cabo tanto la multiplicación como la resta en un sistema numérico con el que no se está familiarizado podría ser muy complicado.

Ejemplo 1.18. División en el sistema decimal:



En una división en el sistema decimal el dividendo puede tener o no punto decimal, pero el divisor no debe tenerlo o bien lo debe tener al final. En este ejemplo se debe recorrer el punto decimal dos posiciones, para mandar el punto decimal al extremo derecho del divisor. Si se recorre el punto decimal cierto número de posiciones en el divisor, también se debe recorrer esas mismas posiciones en el dividendo. En todos los casos en que sea necesario esto se debe de hacer antes de llevar a cabo la división, independientemente del sistema numérico de que se trate.



Después de recorrer el punto decimal al lado derecho del dígito menos significativo del divisor, y de recorrer ese mismo número de posiciones el punto decimal en el dividendo, se procede a llevar a cabo la división.

Como el divisor tiene tres dígitos y en este caso no cabe ninguna vez en tres dígitos del dividendo ya que $769_{(10)} > 432_{(10)}$, se toman cuatro dígitos del dividendo y se encuentra que cociente = 5, después se multiplica este cociente por el divisor, $5 \times 769_{(10)} = 3845_{(10)}$, y dicho resultado se resta de los cuatro dígitos del dividendo para obtener $4325_{(10)} - 3845_{(10)} = 0480_{(10)}$, luego se baja el siguiente dígito, se encuentra el cociente, se multiplica dicho cociente por el divisor y se resta del dividendo y así sucesivamente hasta terminar. Es importante mencionar que si después de bajar un dígito más del dividendo, el divisor no cabe ninguna vez en dicha cantidad,

entonces se pondrá 0 como cociente y se bajará otro dígito. También, una vez que se han bajado todos los dígitos del dividendo y se desean más decimales, se deberán agregar más ceros a la cantidad que resulta de la resta. Todo esto ya se sabe, porque se está familiarizado con el sistema decimal y es válido en todos los sistemas.

Ejemplo 1.19. División en el sistema hexadecimal. Dividir $1AF578.2B_{(16)}$ entre $A.7E2_{(16)}$, obtener una cifra después del punto hexadecimal y hacer la comprobación correspondiente de la división:

```
                                  2  9  1  B  D  .D(16)
A 7 E 2.(16)   | 1  A  F  5  7  8  2  B  0(16)
               | 1  4  F  C  4
               ---------------
               | 0  5  F  9  3  8
               |    5  E  6  F  2
               |    ------------
               | 0  1  2  4  6  2
               |       A  7  E  2
               |       ---------
               |       7  C  8  0  B
               |       7  3  6  B  6
               |       ------------
               |       0  9  1  5  5  0
               |          8  8  6  7  A
               |          ------------
               |             8  E  D  6  0
               |             8  8  6  7  A
               |             ------------
               |                6  6  E  6
```

En este ejemplo se puede observar que el dividendo no tiene parte fraccionaria, ya que al recorrer el punto hexadecimal incluso fue necesario agregar un 0, de forma que para obtener un dígito más después del punto hexadecimal fue necesario agregar un 0 al resto. Para comprobar la división es necesario multiplicar el divisor por el cociente y sumar el resto al resultado, como se muestra a continuación:

```
                  2  9  1  B  D  .D(16)
      ×           A  7  E  2(16)
-------------------------------------
               5  2  3  7  B  A
            2  3  F  8  6  1  6
         1  1  F  C  3  0  B
      1  9  B  1  6  A  2
-------------------------------------
   1  A  F  5  7  7  C  4  1  A(16)
               +     6  6  E  6
-------------------------------------
   1  A  F  5  7  8  2  B  0  .0
```

Notar que el punto que separa la parte entera de la parte fraccionaria se coloca después de que se suma el residuo al resultado de multiplicar el cociente por el divisor.

Ejemplo 1.20. División en otros sistemas.

El procedimiento para llevar a cabo las divisiones en cualquier sistema no cambia, ya que se trata de sistemas posicionales y lo único que debe tenerse en cuenta es la base en la que se está trabajando.

```
                          B  7  D  .6(17)
   9  D  4.(17)  | 6  A  0  C  8  7  .9(17)
                  6  5  9  A
                  ----------
                  0  4  8  2  8
                     4  0  7  B
                     ----------
                     0  7  B  E  7
                        7  8  2  1
                        ----------
                        0  3  C  6  9
                           3  7  B  7
                           ----------
                           0  4  C  2
```

```
                          4  A  9  A  .B(12)
   7  6  1  3.(12)  | 3  0  9  8  5  6  7  2(12)
                     2  6  0  5  0
                     -------------
                     0  6  9  3  5  6
                        6  3  1  0  6
                        -------------
                        0  6  2  5  0  7
                           5  7  6  B  3
                           -------------
                           0  6  A  1  4  2
                              6  3  1  0  6
                              -------------
                              0  7  0  3  8  0
                                 6  A  7  1  9
                                 -------------
                                 0  1  8  6  3
```

1.6 Suma de dos cantidades en complemento a 2

Las operaciones que la computadora realiza internamente se llevan a cabo en una forma muy particular. En principio el sistema numérico utilizado es

el binario y la operación básica es la suma. En computación las cantidades se representan por un conjunto de bits (ceros y unos), usando un bit exclusivo para distinguir las cantidades negativas de las positivas, el cual recibe el nombre de "bit de signo". La convención más común para el signo es 0 = positivo y 1 = negativo.

Existen tres formas de representar cantidades: magnitud verdadera, complemento a 1 y complemento a 2; cada una de estas formas tiene su utilidad dentro de la computación.

Magnitud verdadera

En la representación en magnitud verdadera se muestran los bits en forma real, y una característica de este tipo de representación es que se puede saber fácilmente a cuánto equivale ese conjunto de bits en el sistema decimal usando para ello la representación exponencial como se presenta en la siguiente expresión:

$$1\ 110110101.0111_{(2)} = -(1\times 2^8 + 1\times 2^7 + 1\times 2^5 + 1\times 2^4 + 1\times 2^2 + 1\times 2^0 + 1\times 2^{-2} + 1\times 2^{-3} + 1\times 2^{-4}) = -437.44_{(10)}$$

Magnitud

Bit de signo

Complemento a 1

Como en el sistema binario solamente existen como dígitos válidos el 0 y el 1, se dice que el complemento de 0 es 1 y el complemento de 1 es 0. El complemento de un número en binario se obtiene complementando cada uno de los bits, sin considerar el signo, como se muestra a continuación:

1	1	0	1	0	1	1	1	0	0	1	0	0	1	.	0	$1_{(2)}$ Magnitud verdadera
1	0	1	0	1	0	0	0	1	1	0	1	1	0	.	1	$0_{(2)}$ Complemento a 1

0	1	0	0	0	1	0	0	1	1	.	1	0	$0_{(2)}$ Magnitud verdadera
0	0	1	1	1	0	1	1	0	0	.	0	1	$1_{(2)}$ Complemento a 1

Como se puede observar, para obtener el complemento a 1 de una cantidad expresada en binario es suficiente cambiar todos los ceros por unos y los unos por ceros, pero en ningún momento se cambia el bit de signo, que en este caso es el bit de la extrema izquierda.

Complemento a 2

El complemento a 2 se obtiene sumando 1 al bit menos significativo del complemento a 1, como se muestra en el siguiente caso:

1	0	1	0	1	0	0	0	1	1	0	1	1	0	.	1	$0_{(2)}$	Complemento a 1
													+		1	
1	0	1	0	1	0	0	0	1	1	0	1	1	0	.	1	$1_{(2)}$	Complemento a 2

0	0	1	1	1	0	1	1	0	0	.	0	1	$1_{(2)}$	Complemento a 1
									+				1	
0	0	1	1	1	0	1	1	0	0	.	1	0	$0_{(2)}$	Complemento a 2

Una multiplicación es una sucesión de sumas y una división es una sucesión de restas. Como se mencionó anteriormente, la computadora no realiza restas, ni multiplicaciones, ni divisiones, sino únicamente sumas. Cuando las dos cantidades a sumar son positivas se suman tal cual, pero cuando alguna de ellas es negativa (lo que equivale a restar una cantidad de otra) entonces la cantidad negativa se complementa a 2 y después se suma a la otra cantidad, de forma que una resta se convierte en una suma.

Supóngase que se definen las variables *A*, *B* y *C* del tipo entero por lo que ocupa 1 byte de memoria cada una de ellas. Si $A = 225$, $B = 76$ y en alguna línea de un programa se tiene que $C = A + B$, entonces lo que la computadora hace es lo siguiente: primero convierte los valores de *A* y *B* a binario, y luego realiza la suma de la siguiente forma

+	2	2	$5_{(10)}$	=	0		1	1	1	0	0	0	0	$1_{(2)}$				
+		7	$6_{(10)}$	=	0		0	1	0	0	1	1	0	$0_{(2)}$				
+	3	0	$1_{(10)}$			1	0	0	1	0	1	1	0	$1_{(2)}$				

Signo *Magnitud*

El resultado obtenido es 1 $00101101_{(2)} = -45_{(10)}$ que es muy diferente al $+301_{(10)}$ esperado. Lo que ocurrió aquí es que se presentó un "desbordamiento" al querer guardar en la variable *C* definida de 8 bits, una cantidad mayor. Este error es común que ocurra durante el proceso de la programación, ya que se definen las variables de cierto tipo y con cierta capacidad y algunas veces se desea guardar en ellas una cantidad que sobrepasa esa capacidad. La finalidad de citar este caso es mostrar que se deben de considerar todos los elementos que se presentan en el momento en que la computadora realiza una operación aritmética.

Para resolver el problema de desbordamiento es conveniente definir las variables con una capacidad mayor, por ejemplo suponer que las variables *A*, *B* y *C* son enteras, pero ahora con una capacidad de 16 bits, que es lo

que realmente ocurre en un programa cuando se definen variables con capacidad menor a la requerida.

$$
\begin{array}{rcl}
+\ 225_{(10)} & = & 0\ \ 0\,0\,0\,0\,0\,0\,0\,0\,1\,1\,1\,0\,0\,0\,1_{(2)} \\
+\ \ 76_{(10)} & = & 0\ \ 0\,0\,0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,1\,1\,0\,0_{(2)} \\
\hline
+\ 301_{(10)} & & \underline{0}\ \ 0\,0\,0\,0\,0\,0\,0\,1\,0\,0\,1\,0\,1\,1\,0\,1_{(2)}
\end{array}
$$

Signo *Magnitud*

Se observa ahora que $+301_{(10)} = 0\ 0000000100101101_{(2)}$ es el resultado correcto, con lo cual se evita el desbordamiento.

Es importante mencionar que el desbordamiento solamente ocurre cuando las dos cantidades que se están sumando son del mismo signo, ya que son los únicos casos en que el resultado puede requerir mayor espacio. Cuando las cantidades a sumar son de signo contrario no se presenta el desbordamiento, pues el valor absoluto del resultado siempre será menor al valor absoluto de alguna de las cantidades que se suman.

Ejemplo 1.21. Sumar $A = -225$ con $B = +76$.

Cuando una cantidad es negativa, se deberá encontrar el complemento a 2 de esa cantidad y después realizar la suma, como se muestra a continuación.

$$
\begin{array}{rcll}
-\ 225_{(10)} & = & 1\ \ 1\,1\,1\,0\,0\,0\,0\,1_{(2)} & \text{Magnitud verdadera} \\
 & & 1\ \ 0\,0\,0\,1\,1\,1\,1\,0_{(2)} & \text{Complemento a 1} \\
 & = & \underline{\qquad\qquad\qquad\qquad\quad 1} & \\
 & & 1\ \ 0\,0\,0\,1\,1\,1\,1\,1_{(2)} & \text{Complemento a 2}
\end{array}
$$

Nótese que para obtener el complemento a 1 se cambian todos los bits por su complemento, pero el bit de signo *no* se cambia. Para encontrar el complemento a 2 se le suma 1 al bit menos significativo del complemento a 1.

Ahora sí se procede a sumar el complemento a 2 de la cantidad negativa y la otra cantidad positiva.

$$
\begin{array}{rcl}
-\ 225_{(10)} & = & 1\ \ 0\,0\,0\,1\,1\,1\,1\,1_{(2)} \\
+\ \ 76_{(10)} & = & 0\ \ 0\,1\,0\,0\,1\,1\,0\,0_{(2)} \\
\hline
-\ 149_{(10)} & & 1\ \ 0\,1\,1\,0\,1\,0\,1\,1_{(2)}
\end{array}
$$

Signo *Magnitud*

El resultado obtenido es negativo, como se esperaba, pero la magnitud obtenida no es la correcta, ya que $1\ 01101011_{(10)} = -107_{(10)}$ es diferente de $-149_{(10)}$. En forma general se puede decir que si el resultado de la suma es negativo, se deberá complementar a 2 el resultado.

$$1\ 0\ 1\ 1\ 0\ 1\ 0\ 1\ 1_{(2)} \quad \text{Resultado negativo}$$

$$1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 0_{(2)} \quad \text{Complemento a 1}$$

$$+\ 1$$

$$-\ 1\ 4\ 9_{(10)} = 1\ 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1_{(2)} \quad \text{Complemento a 2}$$

Ejemplo 1.22. Sumar $A = +225$ con $B = -76$.

Complementando a 2 la cantidad negativa se tiene:

$$-\ 7\ 6_{(10)} = 1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0_{(2)} \quad \text{Magnitud verdadera}$$

$$1\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 1_{(2)} \quad \text{Complemento a 1}$$

$$+\ 1$$

$$= 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0_{(2)} \quad \text{Complemento a 2}$$

Sumando el complemento encontrado a la cantidad positiva se obtiene:

$$+\ 2\ 2\ 5_{(10)} = 0\ \ 1\ 1\ 1\ 0\ 0\ 0\ 0\ 1_{(2)}$$

$$-\ \ 7\ 6_{(10)} = 1\ \ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0_{(2)}$$

$$+\ 1\ 4\ 9_{(10)} \quad 1\ 0 \quad 1\ 0\ 0\ 1\ 0\ 1\ 0\ 1_{(2)}$$

Acarreo *Signo* *Magnitud*

En esta suma se puede observar que el resultado es positivo, por lo tanto es el resultado correcto, ya que solamente se complementan a 2 los resultados negativos. También se obtiene un acarreo, el cual se debe despreciar en todos los casos.

Ejemplo 1.23. Sumar $A = -225$ con $B = -76$.

Cuando una cantidad es negativa, se debe determinar el complemento a 2 de esa cantidad. En este caso las dos cantidades a sumar son negativas, por lo tanto se tiene que obtener el complemento a 2 de ambas antes de realizar la suma. Pero también se debe tomar en cuenta que la suma produce un desbordamiento, de tal forma que en el complemento ya se debe trabajar con los bits correctos. Complementando a 2 ambas cantidades y considerando el desbordamiento:

$-225_{(10)} = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 1\ 1\ 0\ 0\ 0\ 1_{(2)}$ Magnitud verdadera

$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 0_{(2)}$ Complemento a 1

1

$= 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1_{(2)}$ Complemento a 2

$-76_{(10)} = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0_{(2)}$ Magnitud verdadera

$1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 0\ 1\ 1_{(2)}$ Complemento a 1

1

$= 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0_{(2)}$ Complemento a 2

Sumando:

$$\begin{array}{rcl} -225_{(10)} = & 1 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 0\ 0\ 1\ 1\ 1\ 1_{(2)} \\ -76_{(10)} = & 1 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0_{(2)} \\ \hline -301_{(10)} & 11 & 1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1_{(2)} \end{array}$$

Acarreo *Signo* *Magnitud*

Si se convierte a decimal el resultado obtenido es posible observar que no es el esperado de $-301_{(10)}$. Sin embargo, se sabe que cuando el resultado de la suma es negativo se deberá complementar a 2. En este caso también se tiene acarreo, el cual se desprecia.

$1\ 1\ 1\ 1\ 1\ 1\ 1\ 0\ 1\ 1\ 0\ 1\ 0\ 0\ 1\ 1_{(2)}$ Resultado negativo

$1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 0_{(2)}$ Complemento a 1

1

$-301 = 1\ 0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 1\ 0\ 1_{(2)}$ Complemento a 2

Conclusiones de las sumas en complemento a 2

De la misma manera en que se sumaron dos cantidades enteras en complemento a 2, también es posible sumar dos cantidades con una parte entera y otra fraccionaria, siempre y cuando se tenga en cuenta que el desbordamiento sólo puede darse en la parte entera (ya que las operaciones se llevan a cabo de derecha a izquierda) y que para evitar éste se debe trabajar con el número de bits suficiente. Para obtener el complemento a 1 se cambia cada bit por su complemento, independientemente de si se encuentra en la parte entera o en la parte fraccionaria. Para encontrar el complemento a 2 se suma un 1 en el bit menos significativo (el que esté más a la derecha) independientemente de si pertenece a la parte entera o bien a la parte fraccionaria, y sólo se complementan a 2 las cantidades negativas y los resultados negativos de las sumas.

1.7 Aplicación de los sistemas numéricos

Cuando se va a un cajero automático a retirar una cantidad de dinero, se llevan a cabo varios pasos: se inserta la tarjeta para que la computadora que tiene el cajero automático lea los datos de la cuenta, después se teclea la clave personal, que por lo general es un conjunto de números, se indica por medio de teclas la opción a realizar (retiro, saldo, depósito, etc.). Para efectuar un retiro, además de la información básica necesaria, se debe proporcionar a la computadora el monto a retirar. Finalmente, después de que la computadora hace entrega de la cantidad solicitada pregunta si se desea realizar otra operación.

Todo lo anterior es proporcionar información a la computadora, para que en función de ella realice el retiro de una cantidad de dinero de una cuenta que se tiene en una institución bancaria determinada. De esta forma se está sustituyendo a una persona que atiende la caja por un cajero automático. Sin embargo, la información se proporciona de manera entendible para uno pero no para la computadora, ya que el único lenguaje que la máquina conoce es el "binario". Por ejemplo, si el monto del retiro es de $500.00 (quinientos pesos) los dígitos 5 y 0 no los entiende, de tal manera que es necesario convertir a binario dicha cantidad para llevar a cabo la operación.

Hay otro inconveniente, la única operación que realiza la computadora es la suma de manera que es necesario restar el 500 en binario del saldo que tiene la cuenta, que también está en binario, aunque uno lo ve en decimal porque la computadora hace la conversión. Como se sabe, para efectuar

Computación y sistemas numéricos

El sistema numérico binario es el lenguaje natural de la computadora ya que con él lleva a cabo operaciones aritméticas, procesa todo tipo de información, controla los periféricos y se comunica con otras computadoras; el sistema binario es el lenguaje máquina. Sin embargo, el sistema binario es poco claro para las personas que no están en el medio de la computación, por las grandes cadenas de unos y ceros que se deben usar para representar información. Por esta razón se crearon medios que permiten una traducción del lenguaje máquina a formas que entienden las personas comunes; así surgió el código ASCII, que no es otra cosa que una tabla de equivalencias entre el sistema binario y los caracteres que se usan para representar palabras. En el código ASCII cada letra, dígito o símbolo se representa por una cadena de ocho bits, de tal manera que es relativamente fácil para la computadora traducir a sistema binario una frase que se escribe en español, inglés o cualquier otro idioma.

una resta por medio de una suma en binario, se complementa a 2 la cantidad negativa y después se realiza la suma.

De esta manera, lo aprendido en este capítulo muestra cuál es el lenguaje de la computadora, porque únicamente con la combinación de ceros y unos es posible representar información que ésta maneja y entiende.

Sin embargo, no solamente se abordó el sistema numérico binario, sino además el decimal, octal, hexadecimal y otros sistemas que ni siquiera existen como los que tienen base 13, 15 o 23 por mencionar algunos, pero que si existieran se comportarían de la misma manera que los sistemas numéricos conocidos, ya que conservan las características propias de todo sistema numérico posicional.

En el campo de la computación, los sistemas numéricos más importantes son el binario, octal y hexadecimal. El binario porque es el lenguaje natural de la computadora, y los sistemas octal y hexadecimal porque permiten compactar la información del lenguaje máquina de una forma muy sencilla, ya que la equivalencia entre sus caracteres es directa, sin llevar a cabo operación aritmética alguna. Por ejemplo, se sabe que sin hacer operaciones:

$$9C4A_{(16)} = 1001\ 1100\ 0100\ 1010_{(2)}$$

Así en lugar de tener cadenas muy grandes de caracteres en donde solamente están los dígitos 0 y 1, dicha información se puede reducir a cadenas más pequeñas que representan lo mismo, y cuando se desea traducir la información nuevamente a binario se logra con gran facilidad, cosa que no ocurre con el lenguaje decimal, ya que para realizar la conversión de decimal a binario es necesario llevar a cabo operaciones aritméticas que absorben buena parte del tiempo de la computadora, con la correspondiente lentitud de la misma.

Existen versiones de computadoras en las cuales la comunicación no es por medio de un lenguaje de alto nivel como Basic, Pascal, C o Java, sino que directamente se le dan las instrucciones en lenguajes que utilizan los sistemas numéricos octal y hexadecimal, ya que por un lado son fáciles de interpretar por la computadora y por otro también son relativamente sencillos de visualizar y entender por el ser humano, puesto que estos sistemas numéricos utilizan letras y números, y no se requieren cadenas extensas de ceros y unos, de tal manera que el sistema numérico octal y hexadecimal son importantes porque son lenguajes intermedios entre la computadora y el ser humano, con cadenas de información mucho más compactas.

Sin embargo, las tablas de equivalencias como el código ASCII no tienen el potencial de un sistema numérico y por lo tanto surgen sistemas numéricos equivalentes al sistema binario, como los sistemas octal y hexadecimal, que permiten compactar grandes cadenas de ceros y unos con la finalidad de ser más claros y ocupar menos espacio en la representación de información. Además la conversión entre los sistemas octal, binario y hexadecimal es muy sencilla no solamente para la computadora, sino también para las personas, ya que la equivalencia entre los caracteres es directa y no es necesario llevar a cabo operación matemática alguna para convertir cantidades octal-binario-hexadecimal sin per-der las propiedades de un sistema numérico posicional, en donde el valor de un carácter depende de la posición que ocupa, sin olvidar que con los sistemas numéricos es posible llevar a cabo operaciones aritméticas básicas, que al combinarse le otorgan un poder mayor a los sistemas numéricos.

1.8 Resumen

Los sistemas numéricos son métodos para la representación de cantidades. Existen sistemas numéricos *aditivos* como el sistema de numeración romano, en donde un mismo dígito vale lo mismo independientemente de la posición que ocupa. Ejemplo: en la cantidad representada en sistema Romano MDLXXIII = 1573, el valor de cada X es 10 puesto que no se toma en cuenta la posición en que están colocadas las X. Existen también sistemas *posicionales* como el decimal, binario, octal y hexadecimal, en donde el valor de cada carácter depende no sólo del propio carácter, sino además de la posición que ocupa en la cantidad representada. Por ejemplo en la cantidad $4353_{(10)}$, el valor del dígito 3 no es el mismo si se encuentra en el extremo que en el interior de la cifra representada (en este caso el tres que está en el extremo derecho vale 3 y el que está entre el cuatro y el cinco tiene un valor de 300).

Los sistemas numéricos posicionales tienen una base y el número de caracteres de un sistema posicional depende de esa base. En binario la base es 2 y los caracteres válidos en ese sistema son 0 y 1, en el octal su base es 8 y los caracteres que se utilizan para representar cantidades son 0, 1, 2, 3, 4, 5, 6 y 7. Algunos sistemas como el hexadecimal requieren de 16 símbolos para representar cantidades, diez de estos símbolos son los dígitos que se utilizan en el sistema decimal y adicionalmente utiliza las primeras seis letras del alfabeto, A, B, C, D, E y F, para completar los 16 caracteres requeridos.

Es posible convertir cantidades de un sistema numérico a otro. Para convertir una cantidad de un sistema numérico cualquiera X a otro sistema numérico cualquiera W, primeramente se convierte del sistema X al sistema decimal y posteriormente se convierte de decimal al sistema W.

Para convertir del sistema X al sistema decimal se utiliza la representación exponencial, se llevan a cabo las operaciones y el resultado ya estará expresado en decimal. Ejemplo: una cantidad que tiene cierto número de dígitos en la parte entera (e_1, e_2, e_3,..., e_n) y cierto número de dígitos en la parte fraccionaria (f_1, f_2, f_3,..., f_m) expresada en un sistema numérico cuya base es B, es posible convertirla a decimal usando la representación exponencial

$$(e_n.....e_3e_2e_1.f_1f_2f_3...f_m)_{(B)} = e_1 \times B^0 + e_2 \times B^1 + e_3 \times B^2 + ...e_n \times B^{n-1} + f_1 \times B^{-1} + f_2 \times B^{-2} + ... + f_m \times B^{-m}$$

en donde los exponentes de B, 0, 1, 2,...,(n – 1), corresponden a la posición de los dígitos de la parte entera contados de derecha a izquierda a partir del punto decimal y los exponentes –1, –2,.., m representan la posición de la parte fraccionaria contados de izquierda a derecha a partir del punto que separa la parte fraccionaria de la parte entera.

Para convertir del sistema decimal al sistema numérico W, la parte entera se divide entre la base a la que se quiere convertir, conservando el resto de la división, y la parte fraccionaria se multiplica por W, conservando la parte entera de la multiplicación.

Las operaciones aritméticas suma, resta, multiplicación y división se realizan de la misma manera en todos los sistemas numéricos. Esto implica que el procedimiento para llevar a cabo operaciones aritméticas en el sistema decimal es el mismo para todos los sistemas posicionales como el sistema binario, octal y hexadecimal y solamente se debe tener en cuenta la base en la que se está realizando la operación. A continuación se tienen sumas de los mismos dígitos que son válidos en decimal y en octal, sin embargo el resultado no es igual porque al dividir el resultado de la suma (12) entre la base de los sistemas el resto cambia.

$$\begin{array}{r} 7_{(10)} \\ +\ 5_{(10)} \\ \hline 12_{(10)} \end{array} \qquad \begin{array}{r|l} & 1 \\ \hline 10 & 12 \\ & 2 \end{array} \qquad \begin{array}{r} 7_{(8)} \\ +\ 5_{(8)} \\ \hline 14_{(8)} \end{array} \qquad \begin{array}{r|l} & 1 \\ \hline 8 & 12 \\ & 4 \end{array}$$

De igual manera en la multiplicación, la diferencia es que el resultado de multiplicar los dos dígitos $5 \times 7 = 35$ en el sistema decimal se divide entre 10 y en el sistema octal entre 8 como se muestra a continuación:

$$\begin{array}{r} 7_{(10)} \\ \times\ 5_{(10)} \\ \hline 35_{(10)} \end{array} \qquad \begin{array}{r|l} & 3 \\ \hline 10 & 35 \\ & 5 \end{array} \qquad \begin{array}{r} 7_{(8)} \\ \times\ 5_{(8)} \\ \hline 43_{(8)} \end{array} \qquad \begin{array}{r|l} & 4 \\ \hline 8 & 35 \\ & 3 \end{array}$$

La computadora no realiza operaciones en diferentes sistemas numéricos sino solamente en binario, tampoco realiza restas, multiplicaciones ni divisiones, sino solamente sumas. Considerando que una multiplicación es una sucesión de sumas, cuando se desea multiplicar $M \times N$ realmente lo que hace es sumar N veces la cantidad M o bien, cuando se desea dividir M/N, a la cantidad M se le restan N veces la cantidad N, pero como se dijo anteriormente que la computadora solamente realiza sumas, antes de sumar se debe complementar a 2 la cantidad que se desea restar y posteriormente sumar la cantidad complementada a dos. Esto implica que las cantidades negativas siempre se deberán complementar a dos antes de llevar a cabo la suma y si el resultado de la suma es negativo también deberá complementarse a dos dicho resultado, para obtener el resultado definitivo.

1.9 Problemas

1.1. Realizar las siguientes conversiones usando tablas de equivalencias binario-octal, binario-hexadecimal.

- **a)** $10010001110101001000010.0101_{(2)}$ a octal.
- **b)** $4EC7.B5_{(16)}$ a binario.
- **c)** $475320.47_{(8)}$ a hexadecimal.
- **d)** $32FE685.9C_{(16)}$ a octal.

1.2. Resolver los incisos del problema 1.1, usando el método general (del sistema X a decimal y del sistema decimal al sistema W).

1.3. Convertir usando el método general.

- **a)** $730568.23_{(9)}$ a base 14.
- **b)** $6G5A.23_{(20)}$ a binario.
- **c)** $4A7E8.52_{(18)}$ a base 15.
- **d)** $93AF5.36_{(17)}$ a base 13.
- **e)** $558C5.3G_{(18)}$ a base 24.

1.4. Realizar las siguientes conversiones usando el método general.

- **a)** $F6CD850.C5_{(17)}$ a base 18.
- **b)** $6A9346C.34_{(19)}$ a base 22.
- **c)** $1452301342001.3_{(7)}$ a base 16.
- **d)** $1000110011010101010110.011_{(2)}$ a base 12.
- **e)** $H45K731.C4_{(21)}$ a base 17.

1.5. Sumar.

a)

$$\begin{array}{rr} & 1\ 1\ 0\ 0\ 1\ 1\ 1\ 0\ .\ 0\ 1\ 1\ 1_{(2)} \\ + & 1\ 1\ 0\ 1\ 0\ 1\ 1\ .\ 1\ 1\ 0\ 1_{(2)} \\ \hline \end{array}$$

b)

$$\begin{array}{rr} & 3\ A\ 5\ 6\ 7\ B\ .\ 1\ 2_{(13)} \\ + & 9\ C\ 0\ 1\ 7\ 2\ .\ 3\ 4_{(13)} \\ \hline \end{array}$$

c)
$$\begin{array}{r} 4\,2\,0\,6\,1\,2\,3\,1\,.\,3\,2\quad 5_{(7)} \\ +\quad 5\,0\,1\,4\,2\,3\,2\,.\,0\,3_{(7)} \\ \hline \end{array}$$

d)
$$\begin{array}{r} 7\,H\,4\,G\,9\,A\,.\,E\,6_{(20)} \\ +\ C\,F\,7\,J\,7\,C\,.\,8\,D_{(20)} \\ \hline \end{array}$$

1.6. Sumar.

a)
$$\begin{array}{r} 6\,3\,4\,5\,2\,1\,7\,.\,8\,4\quad 1_{(9)} \\ +\ 4\,7\,2\,8\,4\,3\,6\,.\,2\,8_{(9)} \\ \hline \end{array}$$

b)
$$\begin{array}{r} 5\,D\,F\,0\,8\,C\,.\,A\,3_{(17)} \\ +\ 9\,D\,B\,G\,1\,E\,9\,.\,5\,C_{(17)} \\ \hline \end{array}$$

c)
$$\begin{array}{r} 8\,A\,7\,4\,2\,6\,3\,B\,.\,4\,9_{(14)} \\ +\quad C\,A\,3\,D\,C\,5\,8\,.\,9\,C\quad 7_{(14)} \\ \hline \end{array}$$

d)
$$\begin{array}{r} 5\,A\,G\,8\,C\,D\,3\,.\,2\,7_{(19)} \\ +\quad G\,7\,H\,A\,4\,F\,.\,C\,E_{(19)} \\ \hline \end{array}$$

1.7. Restar.

a)
$$\begin{array}{r} 1\,5\,D\,A\,8\,4\,3\,.\,2\,3_{(14)} \\ -\quad B\,A\,2\,5\,4\,4\,.\,2\,B_{(14)} \\ \hline \end{array}$$

b)
$$\begin{array}{r} F\,4\,J\,3\,0\,I\,9\,.\,7\,A_{(21)} \\ -\ C\,E\,H\,4\,8\,A\,K\,.\,2\,G_{(21)} \\ \hline \end{array}$$

c)
$$\begin{array}{r} 1\,0\,0\,0\,1\,1\,0\,.\,0\,0\,1_{(2)} \\ -\quad 1\,1\,0\,0\,1\,1\,.\,1\,0\,1_{(2)} \\ \hline \end{array}$$

d)
$$\begin{array}{r} 5\,3\,0\,7\,G\,4\,9\,.\,5_{(17)} \\ -\ 4\,C\,1\,F\,C\,A\,1\,.\,7\quad C\,4_{(17)} \\ \hline \end{array}$$

1.8. Restar.

a)
$$\begin{array}{r} A\ 7\ 5\ 0\ 1\ B\ 3\ .\ E\ 2_{(16)} \\ -\ 9\ 1\ D\ 3\ 5\ C\ B\ .\ 8\ C_{(16)} \\ \hline \end{array}$$

b)
$$\begin{array}{r} 5\ 3\ 4\ 7\ 0\ 1\ 2\ .\ 1\ 2_{(9)} \\ -\ 3\ 7\ 2\ 5\ 4\ 0\ 8\ .\ 6\ 7_{(9)} \\ \hline \end{array}$$

c)
$$\begin{array}{r} 3\ 4\ H\ D\ 0\ 4\ 6\ .\ E\ 2_{(19)} \\ -\ \ H\ 5\ 3\ F\ 1\ G\ .\ A\ 9\ \ 7_{(19)} \\ \hline \end{array}$$

d)
$$\begin{array}{r} D\ 8\ K\ 6\ 7\ 8\ 4\ .\ 0\ F_{(24)} \\ -\ A\ C\ M\ 8\ 2\ H\ E\ .\ N\ 5_{(24)} \\ \hline \end{array}$$

1.9. Multiplicar.

a)
$$\begin{array}{r} 7\ A\ 8\ 9\ C\ .\ 5\ B_{(14)} \\ \times\ \ \ \ \ \ D\ 9\ 7\ .\ 6_{(14)} \\ \hline \end{array}$$

b)
$$\begin{array}{r} 6\ 7\ 0\ 1\ 4\ 3\ .\ 2_{(8)} \\ \times\ \ \ \ \ \ \ \ 5\ 3\ .\ 2\ \ 4_{(8)} \\ \hline \end{array}$$

c)
$$\begin{array}{r} 4\ 8\ 3\ 2\ 1\ .\ 2_{(11)} \\ \times\ \ \ \ \ 0\ .\ 6\ 8\ 8\ \ \ 7_{(11)} \\ \hline \end{array}$$

d)
$$\begin{array}{r} \ \ \ 9\ A\ C\ 1\ .\ 2_{(17)} \\ \times\ \ \ \ \ \ \ \ \ 5\ B\ F\ \ \ 8_{(17)} \\ \hline \end{array}$$

1.10. Multiplicar.

a)
$$\begin{array}{r} E\ 3\ 9\ C\ 2\ 0\ .\ \ D_{(15)} \\ \times\ \ \ \ \ \ C\ .\ 4\ A\ \ 7_{(15)} \\ \hline \end{array}$$

b)
$$\begin{array}{r} 4\ 5\ 3\ 1\ 2\ 4\ 3\ \ 6_{(7)} \\ \times \qquad\quad 2\ 5\ .\ 6\ \ 4_{(7)} \\ \hline \end{array}$$

c)
$$\begin{array}{r} D\ 7\ C\ 3\ F\ .\ 5\ \ G_{(18)} \\ \times \qquad\quad H\ 5\ E\ .\ \ A_{(18)} \\ \hline \end{array}$$

d)
$$\begin{array}{r} 2\ F\ J\ 3\ 0\ 5\ \ C_{(20)} \\ \times \qquad\quad 2\ 3\ H\ .\ \ 8_{(20)} \\ \hline \end{array}$$

1.11. Dividir.

a) $3\ E\ 9\ 8_{(15)}\ \overline{)\ C\ A\ 4\ 5\ 2\ 7\ 6\ .\ 2_{(15)}}$

b) $0\ .\ 7\ 2\ B_{(17)}\ \overline{)\ 1\ A\ 3\ .\ 0\ 2\ 8\ \ 3_{(17)}}$

c) $5\ 3\ .\ 7\ 1_{(9)}\ \overline{)\ 2\ 3\ 8\ 1\ 0\ .\ 6\ 5\ 3_{(9)}}$

d) $7\ C\ 9\ .\ A_{(13)}\ \overline{)\ B\ 4\ 5\ A\ C\ 7\ .\ 9\ 4_{(13)}}$

1.12. Dividir.

a) $2\ A\ .\ 7\ 6_{(12)}\ \overline{)\ 5\ B\ 7\ 4\ 6\ 8\ A\ .\ 9_{(12)}}$

b) $2\ 1\ 0\ .\ 2_{(3)}\ \overline{)\ 1\ 0\ 2\ 2\ 0\ 1\ .\ 2\ 1_{(3)}}$

c) $C\ 6\ 9\ 7_{(14)}\ \overline{)\ A\ D\ 3\ C\ 5\ 8\ 4\ C\ 3_{(14)}}$

d) $6\ H\ \ 9_{(20)}\ \overline{)\ 5\ H\ 6\ C\ 4\ 0\ C\ .\ E_{(20)}}$

1.13. Realizar la suma en complemento a 2, en cada uno de los siguientes incisos. Considerar que las cantidades que se están sumando se definen como enteras y que ocupan 2 bytes de memoria. Agregar bytes en caso de ser necesario, para evitar el desbordamiento.

a) $+\ 6\ 5\ 5\ 0\ 8_{(10)}$
$+\ \ \ \ \ 1\ 0\ 3_{(10)}$

b) $-\ 6\ 5\ 5\ 0\ 8_{(10)}$
$+\ \ \ \ \ 1\ 0\ 3_{(10)}$

c) $-\ 6\ 5\ 5\ 0\ 8_{(10)}$
$-\ \ \ \ \ 1\ 0\ 3_{(10)}$

c) $+\ 6\ 5\ 5\ 0\ 8_{(10)}$
$-\ \ \ \ \ 1\ 0\ 3_{(10)}$

1.14. Considerar que las cantidades que se suman se definen como enteras y que ocupan un byte en memoria principal cada una de ellas. Realizar la suma en complemento a 2. Agregar bytes en caso de ser necesario, para evitar el desbordamiento.

a) $+\ 1\ 9\ 5_{(10)}$
$+\ \ \ 7\ 6_{(10)}$

b) $-\ 1\ 9\ 5_{(10)}$
$+\ \ \ 7\ 6_{(10)}$

c) $+\ 1\ 9\ 5_{(10)}$
$-\ \ \ 7\ 6_{(10)}$

d) $-\ 1\ 9\ 5_{(10)}$
$-\ \ \ 7\ 6_{(10)}$

1.15. Considerar que la parte entera en cada una de las cantidades que se suman ocupan 8 bits y que la parte fraccionaria ocupa 4 bits. Realizar la suma en complemento a 2. Agregar bytes en caso de ser necesario, para evitar el desbordamiento.

a) $+\ 5\ 4\ .\ 2\ 3_{(10)}$
$+\ 2\ 8\ .\ 5\ 6_{(10)}$

b) $-\ 5\ 4\ .\ 2\ 3_{(10)}$
$+\ 2\ 8\ .\ 5\ 6_{(10)}$

c) $+\ 5\ 4\ .\ 2\ 3_{(10)}$
$-\ 2\ 8\ .\ 5\ 6_{(10)}$

d) $-\ 5\ 4\ .\ 2\ 3_{(10)}$
$-\ 2\ 8\ .\ 5\ 6_{(10)}$

1.16. Contestar en cada una de las preguntas SÍ o NO, argumentando su respuesta.

a) ¿El dígito más pequeño para representar cantidades numéricas en todo sistema numérico posicional es el 0?

b) ¿El dígito más grande en sistema base 22 podría ser la letra L?

c) ¿Solamente se pueden usar letras y dígitos para representar cantidades en un sistema numérico?

d) ¿Si un sistema numérico utiliza los dígitos del 0 al 9 y además requiere más letras de las que tiene el alfabeto no podría existir dicho sistema?

e) ¿Así como se tienen tablas de equivalencia entre los sistemas binario-octal y binario-hexadecimal, se puede tener un tabla de equivalencia para llevar a cabo conversiones binario-decimal?

f) ¿Es posible obtener una tabla de conversiones de binario a base 4 y que funcione perfectamente bien?

g) ¿Es posible probar una resta, por medio de una suma en cualquier sistema numérico, como se hace en el sistema decimal?

h) ¿Es posible probar una multiplicación en cualquier sistema numérico, como se hace en el sistema decimal?

i) ¿La cantidad mayor que puede caber en n dígitos está dada por la expresión $2^n - 1$?

j) ¿Cuando se suman dos cantidades con el mismo signo en complemento a 2, siempre se presenta un desbordamiento?

k) ¿El complemento a 2 consiste en sumarle un 1 al bit menos significativo de la parte entera?

1.17. Diseñar un algoritmo que permita llevar a cabo conversiones de un sistema a otro.

1.18. Diseñar un algoritmo que permita sumar dos cantidades en cualquier sistema numérico.

1.19. Diseñar un algoritmo para restar dos cantidades en cualquier sistema.

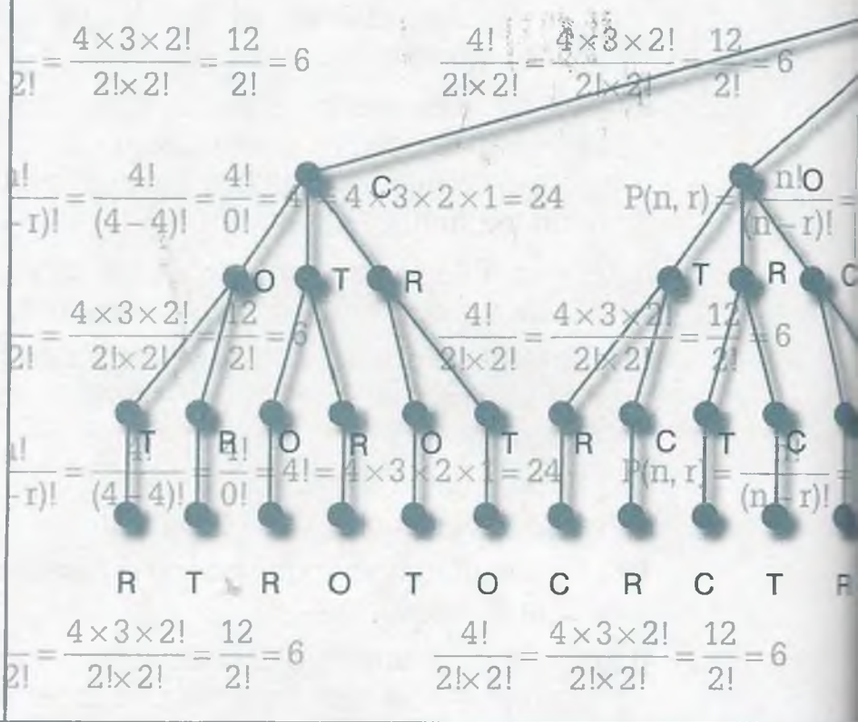
1.20. Diseñar un algoritmo para multiplicar dos cantidades en cualquier sistema.

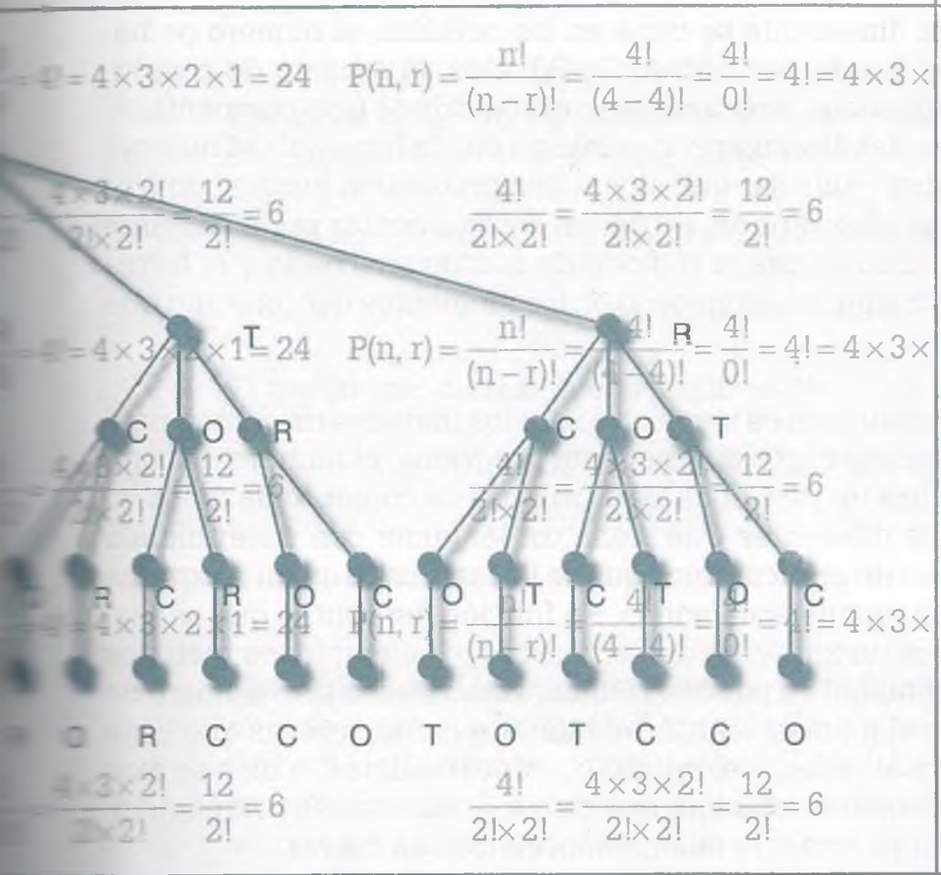
1.21. Diseñar un algoritmo que permita dividir dos cantidades en cualquier sistema.

1.22. Desarrollar un sistema que permita llevar a cabo conversiones de un sistema a otro y realizar operaciones aritméticas básicas en diferentes sistemas.

CAPÍTULO 2

Métodos de conteo





El contar es uno de los descubrimientos de la humanidad, por lo tanto no puede ser más complicado que lo que los hombres son capaces de comprender.

Richard Feynman

Objetivos

- Aprender a calcular el número de permutaciones de un conjunto de *n* elementos en arreglos de tamaño *r*, con o sin repetición.
- Aprender a calcular el número de combinaciones de un conjunto de *n* elementos, en arreglos de tamaño *r*.
- Distinguir los conceptos de permutaciones y combinaciones.
- Aplicar los métodos de conteo en la solución de problemas de computación.

2.1 Introducción

Análisis combinatorio

"Los problemas generales agrupados bajo el nombre de 'Análisis combinatorio' no parecen haber sido considerados antes de los últimos siglos de la antigüedad clásica, únicamente la fórmula

$$\binom{n}{2} = \frac{n(n-1)}{2}$$

aparece en el siglo III de nuestra era. El matemático indio Bhaskara (siglo XII) conocía la fórmula general para $\binom{n}{p}$. Un estudio más sistemático se halla en un manuscrito de Levi ben Gerson, a principios del siglo XIII: obtiene la fórmula de recurrencia que permite calcular el número V_n^p de variaciones de n objetos tomados p a p, y en particular el número de permutaciones de n objetos, enunciando también reglas equivalentes a las relaciones.

$$\binom{n}{p} = \frac{V_n^p}{p!}, \binom{n}{n-p} = \binom{n}{p}$$

Pero este manuscrito no parece haber sido conocido por sus contemporáneos, y los resultados fueron hallados poco a poco por los matemáticos de los siglos siguientes."

Nicolas Bourbaki

Es posible contar el dinero que se tiene en los bolsillos, el número de habitantes de un país que tienen entre 20 y 30 años, el número de computadoras con determinadas características que produce una compañía, el número de palabras del diccionario que inician con la letra "u", el número de placas para control vehicular que se pueden producir si inician con tres letras y terminan con dos dígitos, en fin, es posible contar prácticamente todo, siempre y cuando se use el método de conteo adecuado y la forma apropiada para distinguir sin equivocación los elementos del conjunto que se quieren contar.

En el área de la computación es necesario usar los métodos de conteo para determinar el número de ciclos que tiene un programa, el número de comparaciones que realiza un programa para ordenar un conjunto de datos, el número de palabras diferentes que tiene un lenguaje con determinada gramática, el número de intercambios que se llevan a cabo en un programa para resolver un sistema de ecuaciones. En función del conteo que se realiza en computación, un software determinado (por ejemplo los métodos para ordenar información) se puede clasificar como bueno si el número de comparaciones que ejecuta es significativamente menor que las que lleva a cabo otro software al ordenar el mismo conjunto de datos, o bien se dice que un programa es menos eficiente que otro si el número de comparaciones que realiza para procesar la misma información es mayor.

En conclusión, los métodos de conteo en computación permiten optimizar los recursos de la computadora y disminuir el tiempo de ejecución de un proceso, lo que produce una mejora en el tiempo de respuesta. Con un buen manejo de estos métodos es posible determinar cuál es el programa más eficiente, sin necesidad de ejecutarlo.

2.2 Principios fundamentales del conteo

En los métodos de conteo se encuentran implícitas dos operaciones aritméticas fundamentales, la multiplicación y la suma, y esto da origen a lo que se conoce como el *principio fundamental del producto* y el *principio fundamental de la adición*. En base a estos principios, es posible desarrollar los métodos de conteo para establecer el número de permutaciones o combinaciones que se pueden obtener entre los elementos de un conjunto de datos.

2.2.1 Principio fundamental del producto

Este principio establece que si una operación se puede hacer de n formas y cada una de éstas puede llevarse a cabo de m maneras distintas en

una segunda operación, se dice que juntas las operaciones pueden realizarse de $n \times m$ formas distintas.

Ejemplo 2.1. Un algoritmo tiene 3 procedimientos (A, B, C) y cada procedimiento tiene 4 ciclos (1, 2, 3, 4). ¿Cuántos ciclos tiene el algoritmo?

Aplicando el principio fundamental del producto se tiene que

$$\text{total de ciclos} = 3 \times 4 = 12$$

El conjunto E de resultados posibles es:

$$E = \{A1, A2, A3, A4, B1, B2, B3, B4, C1, C2, C3, C4\}$$

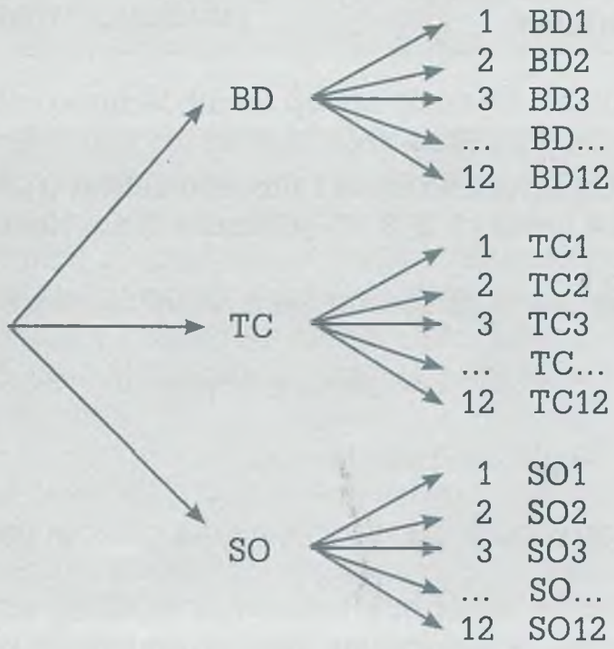
Ejemplo 2.2. En una biblioteca hay tres libros distintos de computación, uno de Bases de Datos (BD) otro de Teoría de la Computación (TC) y un tercero de Sistemas Operativos (SO), y hay un grupo de 12 alumnos que pueden hacer uso de ellos. Si se desea saber los posibles arreglos que se pueden formar entre libros y alumnos, el resultado se puede obtener multiplicando el número de libros por el número de alumnos:

$$\text{Resultados posibles} = 3 \times 12 = 36$$

Estos resultados se muestran en la siguiente tabla:

	Alumnos											
Libros	1	2	3	4	5	6	7	8	9	10	11	12
BD	BD1	BD2	BD3	BD4	BD5	BD6	BD7	BD8	BD9	BD10	BD11	BD12
TC	TC1	TC2	TC3	TC4	TC5	TC6	TC7	TC8	TC9	TC10	TC11	TC12
SO	SO1	SO2	SO3	SO4	SO5	SO6	SO7	SO8	SO9	SO10	SO11	SO12

También se puede representar esta información por medio de un árbol:



Esto significa que el libro de Bases de Datos lo pueden utilizar los alumnos 1, 2, 3,..., 12 y lo mismo sucede con el de Teoría de Computación y el de Sistemas Operativos.

Ejemplo 2.3. Se desea conocer el número de placas que se pueden formar si éstas tienen dos dígitos (D) y tres letras mayúsculas (L), como se muestra en la siguiente figura:

Lo primero que hay que considerar es que existen 10 dígitos (0, 1, 2, 3, 4, 5, 6, 7, 8, 9) y que el número de letras mayúsculas es 27 (A, B, C, ... , Z). A partir de esto se puede formar el siguiente número de placas diferentes:

$$\text{placas} = 10 \times 10 \times 27 \times 27 \times 27 = 1968300$$

si se pueden repetir letras y números, y

$$\text{placas} = 10 \times 9 \times 27 \times 26 \times 25 = 1579500$$

si no hay repetición.

2.2.2 Principio fundamental de la adición

Este principio establece que si un evento se puede llevar a cabo en n o m lugares distintos, además de no ser posible que se lleve a cabo el mismo evento en dos lugares distintos al mismo tiempo, entonces el evento se puede realizar de $m + n$ maneras diferentes.

Ejemplo 2.4. Una persona puede pagar el servicio de agua potable en cualquiera de las 7 oficinas municipales o bien en cualquiera de los 30 bancos de la ciudad. ¿En cuántos lugares diferentes se puede pagar el servicio de agua potable?

$$\text{lugares en donde se puede pagar} = n + m = 7 + 30 = 37$$

Ejemplo 2.5. El día domingo de 12:00 a 14:00, una persona puede ver uno de los 4 partidos de fútbol que pasan en diferentes canales de televisión, o bien ver alguna de las 6 películas que transmiten a esa misma hora en otros seis canales diferentes o ver alguno de los 2 conciertos que coinciden también en ese horario. ¿Cuántos eventos diferentes puede ver en la televisión esa persona de 12:00 a 14:00?

$$\text{eventos diferentes} = 4 + 6 + 2 = 12$$

Dependiendo del problema, algunas veces es necesario combinar la adición y el producto como se muestra a continuación.

Ejemplo 2.6. Supóngase que se desea etiquetar las gavetas de los alumnos de la Universidad, y que la etiqueta puede estar marcada con un solo dígito, una sola letra o la combinación de una sola letra con un solo dígito (sin importar si primero se pone la letra y después el dígito o al contrario). Bajo estas condiciones, el número de etiquetas distintas que se pueden formar son:

$$\text{etiquetas} = \text{dígitos} + \text{letras} + \text{letras} \times \text{dígitos} + \text{dígitos} \times \text{letras} =$$
$$10 + 27 + 27 \times 10 + 10 \times 27 = 577$$

2.3 Permutaciones

Las permutaciones son el número de formas distintas en que uno o varios objetos pueden colocarse, intercambiando sus lugares y siguiendo ciertas reglas específicas para guardar un orden. También se puede considerar como todo arreglo en el que es importante la posición que ocupa cada uno de los elementos que integran dicho arreglo.

Ejemplo 2.7. Supóngase que la academia de sistemas y computación está integrada únicamente por 3 maestros (Ignacio, Miriam y Jorge), y que con ellos es necesario integrar un comité que estará conformado por un presidente, un secretario y un vocal. Supóngase que primero se selecciona a la persona que ocupará el puesto de presidente, después a la que tendrá la función de secretario y finalmente a la que fungirá como vocal. ¿Cuántos tipos de arreglos se pueden formar?

$$\text{Permutaciones (P)} = 3 \times 2 \times 1 = 6$$

Si n es el número de elementos del conjunto (en este caso n = 3), entonces el número de permutaciones que se pueden formar cuando los arreglos son de tamaño n es n!

$$P = n\,(n-1)\,(n-2)\,\ldots\,1 = n!$$

Esto implica que el presidente se puede seleccionar de 3 maneras ya que es el primer puesto que se asigna, el secretario se puede seleccionar de 2 formas, ya que una persona fue seleccionada para ocupar el puesto de presidente, y el vocal solamente se podrá seleccionar de una forma considerando que las otras dos personas fueron asignadas como presidente y secretario respectivamente. La siguiente tabla muestra los diferentes comités o permutaciones que es posible formar.

	Permutaciones					
Puestos	1	2	3	4	5	6
Presidente	Ignacio	Ignacio	Miriam	Miriam	Jorge	Jorge
Secretario	Miriam	Jorge	Ignacio	Jorge	Miriam	Ignacio
Vocal	Jorge	Miriam	Jorge	Ignacio	Ignacio	Miriam

Hay que recordar que el factorial de n, denotado como n!, se define como:

$$0! = 1 \qquad 1! = 1$$

$$n! = n(n-1)(n-2)\cdots(2)1 \qquad \text{para } n > 1$$

Combinatoria

La combinatoria es una rama de la matemática que estudia colecciones finitas de objetos que satisfacen algunos criterios especificados, y que en particular se ocupa del recuento de los objetos de dichas colecciones (*combinatoria enumerativa*), del problema de determinar si cierto objeto "óptimo" existe (*combinatoria extremal*) y de establecer la estructura algebraica que estos objetos pueden tener (*combinatoria algebraica*).

Por el tipo de problemas que se plantea, la combinatoria se aplica en el álgebra, en la teoría de la probabilidad, en la teoría ergódica y en la geometría, así como en la ciencia de la computación y la física estadística.

Los matemáticos Gian Carlo Rota, Paul Erdös y George Pólya se han destacado por sus investigaciones fundamentales en esta área de la matemática.

Un ejemplo de pregunta combinatoria es la siguiente: ¿cuántas ordenaciones pueden hacerse en un mazo de 52 cartas? Ese número es 52!, esto es, el producto de todos los números naturales desde el 1 al 52, lo cual es alrededor de 8.07×10^{67}. Este número es realmente grande: es mayor que el cuadrado del número de Avogadro, $6{,}02 \times 10^{23}$ (el número de átomos, moléculas, etc., que hay en un mol), y es del mismo orden magnitud, 10^{67}, que la cantidad de átomos en la Vía Láctea.

siendo n un entero no negativo.

En el caso en que n = 6 se tiene que:

$$6! = 6 \times 5 \times 4 \times 3 \times 2 \times 1 = 720$$

Ejemplo 2.8. A diferencia del ejemplo 2.7, supóngase ahora que la academia no está formada por tres maestros sino por 8, y que de ese conjunto se desea integrar el comité que ocupará los puestos de presidente, secretario y vocal, suponiendo que primero se selecciona a quien ocupará el puesto de presidente, después el de secretario y al final el de vocal. ¿Cuántos arreglos diferentes se pueden formar?

La respuesta es

$$P = 8 \times 7 \times 6 = 336$$

Como se ve, el presidente se puede seleccionar de 8 formas distintas, el secretario de 7 y el vocal de 6.

Si n es el número de elementos del conjunto (n = 8 en este caso) y r es el número de elementos que forman el comité (en este caso r = 3), la expresión anterior se puede representar en función de n y r de la siguiente manera:

$$P = \frac{n!}{(n-r)!}$$

Sustituyendo n = 8 y r = 3, se tiene que:

$$P = \frac{8!}{(8-3)!} = \frac{8!}{5!} = \frac{8 \times 7 \times 6 \times 5!}{5!} = 8 \times 7 \times 6 = 336$$

En general, el número de permutaciones de n objetos diferentes, tomando r a la vez, se indica de la siguiente manera:

$$P(n, r) = \frac{n!}{(n-r)!}$$

Ejemplo 2.9. De un conjunto de 5 computadoras (A, B, C, D, E) se seleccionan 3 para mandarse respectivamente a los departamentos de Ventas, Compras y Mantenimiento. Si la primera que se selecciona es para

Ventas, la segunda para Compras y la tercera para Mantenimiento, ¿de cuántas formas se pueden formar los paquetes?

La respuesta es la siguiente:

$$\underset{\text{Ventas}}{5} \times \underset{\text{Compras}}{4} \times \underset{\text{Mantenimiento}}{3} = 60$$

La computadora de Ventas se puede seleccionar de 5 formas diferentes, la de Compras de 4 ya que se seleccionó una para Ventas y la de Mantenimiento de 3 porque ya se seleccionó una para Ventas y otra para Compras.

Este problema también se puede resolver de la siguiente manera:

$$P(5, 3) = \frac{5!}{(5-3)!} = \frac{5\times4\times3\times2!}{2!} = 60$$

Si se desea saber el número de formas en que se pueden ubicar las 5 computadoras, pero ahora en 5 departamentos diferentes (Dirección, Personal, Ventas, Compras y Mantenimiento), el número de permutaciones se puede encontrar de la siguiente manera:

$$\underset{\text{Dirección}}{5} \times \underset{\text{Personal}}{4} \times \underset{\text{Ventas}}{3} \times \underset{\text{Compras}}{2} \times \underset{\text{Mantenimiento}}{1} = 120$$

o bien

$$P(5, 5) = \frac{5!}{(5-5)!} = \frac{5\times4\times3\times2\times1}{0!} = 120$$

A partir del ejemplo 2.9 se ve que cuando r = n el número de permutacione es n!:

$$P(n, r) = \frac{n!}{(n-r)!} = \frac{n!}{(n-n)!} = \frac{n!}{0!} = \frac{n!}{1} = n! \qquad 0 \le r \le n$$

Si se permiten repeticiones, entonces el número de permutaciones de objetos en bloques de tamaño r está dado por:

$$P(n, r) = \underbrace{n \times n \times n \cdots \times n}_{r \text{ veces}} = n^r$$

Ejemplo 2.10. ¿Cuál es el número de permutaciones de las letras de la palabra "sal"?

a) Sin repetición y r = n.
b) Con repetición y r = n.
c) Sin repetición y r = 2.
d) Con repetición y r = 2.

La respuesta en cada caso es la siguiente:

a) $3! = 6$:

{sal, sla, asl, als, lsa, las}

b) $P(3, 3) = 3^3 = 27$:

{sss, ssa, sas, ass, saa, asa, aas, aaa, ssl, sls, lss, sll, lsl, lls, lll, lla, lal, all, laa, ala, aal, sal, sla, las, lsa, als, asl}

c) $P(3, 2) = \frac{3!}{(3-2)!} = \frac{3\times 2\times 1}{1!} = 6$:

{sa, sl, al, as, la, ls}

d) $P(3, 2) = n^r = 3^2 = 9$:

{aa, as, al, ss, sa, sl, ll, la, ls}

Algunas veces el tamaño del bloque es mayor que el número de objetos ($r > n$), y en este caso el número de permutaciones es

$$P(n, r) = n^r$$

Un ejemplo muy claro es lo que ocurre con el sistema numérico octal, en donde cada dígito en octal equivale a una cadena de ceros y unos en binario.

Octal	Binario
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

En este caso n = 2 ya que sólo son los dígitos 0 y 1 los que se deben de permutar, y la longitud de la cadena es r = 3 por lo que se obtiene

$$P(2, 3) = 2^3 = 8$$

Ejemplo 2.11. En el sistema trinario son válidos los dígitos 0, 1 y 2 de tal forma que:

a) El número de permutaciones en trinario en grupos de 2 sin que se repitan los dígitos es:

$$P(3, 2) = \frac{3!}{(3-2)!} = \frac{3!}{1!} = 6$$

{12, 13, 21, 23, 31, 32}

b) El número de permutaciones en trinario en grupos de 2 con repetición es:

$$3^2 = 9$$

{11, 12, 13, 21, 22, 23, 31, 32, 33}

Algunas veces no todos los objetos son distintos, sino que parte de ellos se repiten. En este caso el número de permutaciones de n objetos de los cuales t_1 son de un tipo, t_2 son de otro tipo distinto y t_k son del k-ésimo tipo, está dado por

$$P(n, k) = \frac{n!}{t_1!\, t_2! \cdots t_k!}$$

en donde $t_1 + t_2 + \ldots + t_k = n$.

Ejemplo 2.12. Obtener las permutaciones de la palabra BEBE.

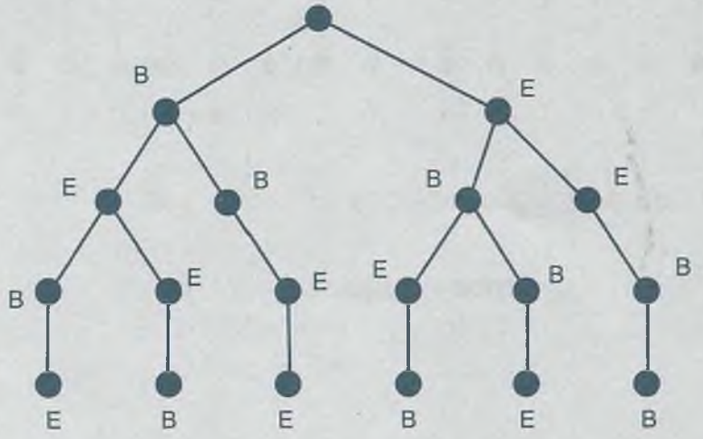
Primero hay que observar que n = 4, ya que es el número de letras de la palabra BEBE, y que los tipos involucrados son

Tipos de letras	Letra
t_1=2	B
t_2=2	E

Por tanto se tiene que

$$P(4, 2) = \frac{4!}{2! \times 2!} = \frac{4 \times 3 \times 2!}{2! \times 2!} = \frac{12}{2!} = 6$$

Por medio de un árbol se obtiene que:



La lista de las permutaciones es

permutaciones = {BEBE, BEEB, BBEE, EBEB, EBBE, EEBB}

Ejemplo 2.13. Obtener las permutaciones de las letras de la palabra COTORRO bajo las siguientes condiciones:

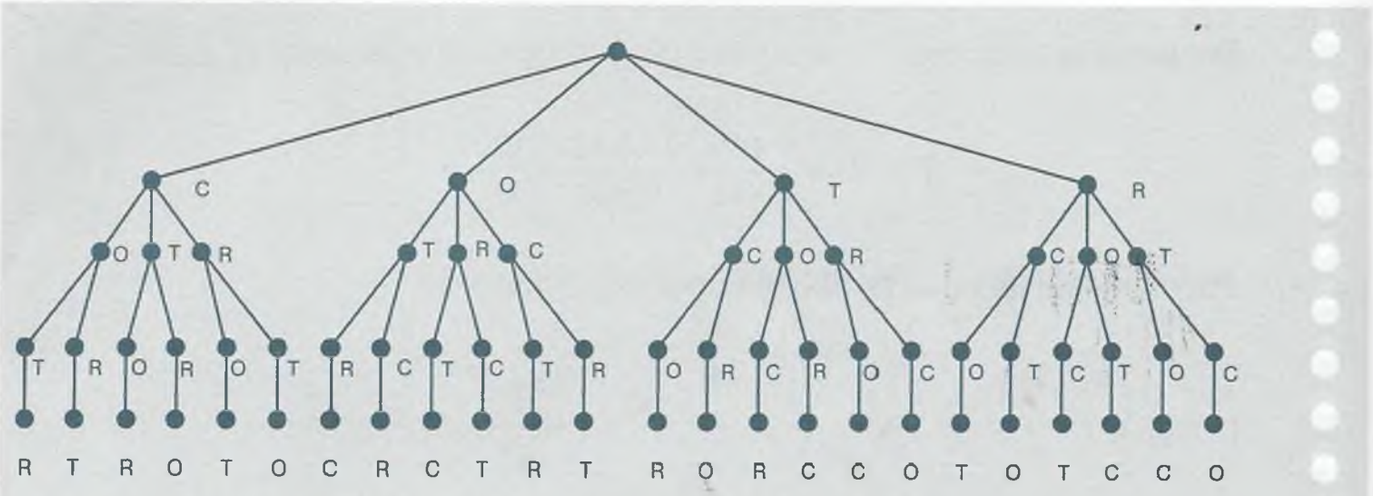
a) Eliminando tipos de repetidos.

b) Considerando que todas son diferentes, aun cuando se trate de la misma letra.

La solución es la siguiente:

a) Las letras distintas en la palabra COTORRO son COTR, por lo tanto n = 4; si r = n se tiene que

$$P(4, 4) = \frac{4!}{(4-4)!} = \frac{4!}{0!} = 4! = 4 \times 3 \times 2 \times 1 = 24$$



b) Considerando que los tipos de letras son

Tipos de letras	Letra
$t_1 = 1$	C
$t_2 = 3$	O
$t_3 = 1$	T
$t_4 = 2$	R

se tiene que

$$P = \frac{n!}{t_1!\, t_2! \cdots t_k!} = \frac{7!}{1! \times 3! \times 1! \times 2!} = \frac{7 \times 6 \times 5 \times 4 \times 3!}{3! \times 2!} = 420$$

2.4 Combinaciones

Combinación es todo arreglo de elementos que se seleccionan de un conjunto, en donde no interesa la posición que ocupa cada uno de los elementos en el arreglo, esto es, no importa si un elemento determinado es el primero, el de en medio o el que está al final del arreglo.

El número de combinaciones de n objetos distintos, tomados r a la vez, se encuentra dado por la expresión:

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

Ejemplo 2.14. Supóngase que la academia de sistemas y computación está integrada únicamente por 3 maestros (Ignacio, Miriam y Jorge), y que con ellos es necesario formar un comité que estará integrado por un presidente, un secretario y un vocal. Supóngase también que no importa cuál de los elementos ocupará cualquiera de los puestos.¿Cuántos tipos de arreglos se pueden formar?

En este caso r = n = 3 ya que se está tomando el total de los elementos para formar el arreglo, por lo tanto:

$$\binom{3}{3}=\frac{3!}{3!(3-3)!}=\frac{3!}{3!\times 0!}=\frac{3!}{3!}=1$$

lo cual indica que el número de combinaciones es 1:

(Ignacio, Jorge, Miriam) = (Ignacio, Miriam, Jorge) =
(Miriam, Jorge, Ignacio)

ya que no es importante el orden o el puesto que ocupen los maestros.

Ejemplo 2.15. A diferencia del ejemplo 2.14, ahora supóngase que la academia está integrada por 8 maestros, y que de ese conjunto se desea seleccionar a 3 de ellos que integrarán el comité que ocupará los puestos de presidente, secretario y vocal. Suponiendo que no es importante quién ocupe cualquiera de los puestos, ¿cuántos arreglos diferentes se pueden formar?

El número de arreglos es:

$$\binom{8}{3}=\frac{8!}{3!(8-3)!}=\frac{8\times 7\times 6\times 5!}{3!\times 5!}=56$$

Suponiendo que el conjunto de maestros es A={Ignacio, Miriam, Jorge, Raymundo, Esperanza, Manuel, Rogelio, Ezequiel}, las 56 combinaciones son todas las tripletas que se pueden formar con ellos, en donde el orden en que aparece el nombre de un maestro no es importante sino solamente que esté contenido en ella. Esto implica que por ejemplo las tripletas (Miriam, Ezequiel, Raymundo) y (Raymundo, Miriam, Ezequiel) realmente son iguales.

Ejemplo 2.16. Una compañía de desarrollo de software desea contratar a 8 personas de un grupo de 14 jóvenes profesionistas que acaban de egresar de la universidad como licenciados en informática. ¿De cuántas maneras se puede seleccionar a los 8 profesionistas si se aplican las siguientes condiciones?

a) No importa el orden en que se les selecciona.

b) Si se les selecciona pensando en que el primero ocupará la Dirección de desarrollo de software, el segundo la Jefatura de diseño de software, el tercero el puesto de Programador A, el cuarto el de Programador B, y así sucesivamente hasta que el octavo ocupará el puesto de Programador F, todo esto considerando que la responsabilidad del puesto y el salario van de manera decreciente.

c) Si se selecciona a los 14 profesionistas para ocupar 14 puestos de diferente responsabilidad, sin que sea importante el orden de selección.

La solución de cada caso es la siguiente:

a) Como el orden no importa, se trata de un problema de combinaciones:

$$\binom{14}{8} = \frac{14!}{8!(14-8)!} = 3003$$

b) En este caso se observa claramente que el orden de selección es importante ya que el primero será el que ocupe el puesto de mayor responsabilidad y con mejor salario, así como el octavo será el puesto de menor salario y menor responsabilidad. Por lo tanto el número de permutaciones es:

$$P(14, 8) = \frac{14!}{(14-8)!} = 121080960$$

c) Éste es un problema de combinaciones en donde r = n y por lo tanto el número de formas en que se pueden seleccionar para ocupar los diferentes puestos sin importar el orden es:

$$\binom{14}{14} = \frac{14!}{14!(14-14)!} = 1$$

Ejemplo 2.17. Se tienen 10 computadoras y 6 impresoras. Determinar el número de paquetes que es posible formar, si se desea que éstos contengan 4 computadoras y 3 impresoras.

Las formas en que se pueden seleccionar 4 computadoras de un grupo de 10 son:

$$\binom{10}{4} = \frac{10!}{4!(10-4)!} = 210$$

Las maneras en que es posible seleccionar 3 impresoras de un grupo de 6 es:

$$\binom{6}{3} = \frac{6!}{3!(6-3)!} = 20$$

Por la regla del producto se obtiene que el número de paquetes diferentes que se pueden formar, conteniendo 4 computadoras y 3 impresoras es:

$$\text{paquetes} = 210 \times 20 = 4200$$

Ejemplo 2.18. El examen de regularización de la materia de Física está integrado por 5 unidades diferentes, y cada una de éstas tiene 7 preguntas también diferentes.

a) Si se desea que se contesten solamente 4 preguntas de cada unidad, sin importar el orden, ¿de cuántas maneras distintas se puede contestar el examen?

b) Si deben de contestar 6 preguntas de la primera unidad, 4 de la segunda y tercera unidades y 3 de las unidades cuatro y cinco, sin importar el orden en que se contesten, ¿de cuántas formas distintas es posible contestar el examen?

c) En total son 35 preguntas, considerando las cinco unidades. Si solamente se deben de contestar 20 preguntas sin importar el orden ni la unidad, ¿de cuántas maneras diferentes se puede contestar el examen?

d) ¿De cuántas maneras se puede contestar el examen si se requiere que se contesten todas las unidades sin importar el orden?

La solución en cada caso es la siguiente:

a) El número de formas en que puede contestar cada unidad que contiene 7 preguntas y debe contestar solamente 4 es:

$$\binom{7}{4} = \frac{7!}{4!(7-4)!} = 35$$

Por lo tanto el número de maneras diferentes en que se puede contestar el examen considerando las 5 unidades es

$$35 \times 35 \times 35 \times 35 \times 35 = 35^5 = 52521875$$

b) El número de maneras distintas en que se pueden contestar las unidades es:

Para la unidad 1

$$\binom{7}{6} = \frac{7!}{6!(7-6)!} = 7$$

Para cada una de las unidades 2 y 3

$$\binom{7}{4} = \frac{7!}{4!(7-4)!} = 35$$

Para cada una de las unidades 4 y 5

$$\binom{7}{3} = \frac{7!}{3!(7-3)!} = 35$$

El número de formas diferentes en que se puede contestar un examen es

$$7 \times 35 \times 35 \times 35 \times 35 = 10504375$$

c) Formas diferentes en que se puede contestar el examen:

$$\binom{35}{20} = \frac{35!}{20!(35-20)!} = 3247943160$$

d) Si se requiere contestar todas las preguntas de las cinco unidades:

$$\binom{35}{35} = \frac{35!}{35!(35-35)!} = 1$$

2.5 Aplicaciones en la computación

En el campo de la computación es frecuente que se desee contar el número de veces que se ejecuta una instrucción, el número de palabras que se puede obtener con determinada gramática, el número de bits que se requieren para representar una cantidad, etcétera.

A continuación se presentan algunos ejemplos de la aplicación de los métodos de conteo en el campo de la computación.

2.5.1 Binomio elevado a la potencia n

Considérese el problema de elevar un binomio a una cierta potencia, por ejemplo $(x + y)^2$:

$$(x + y)^2 = (x + y)(x + y) = x^2 + xy + xy + y^2 = x^2 + 2xy + y^2$$

De esta manera se obtiene la conocida regla que establece que un binomio elevado al cuadrado es igual al *cuadrado del primero más el doble producto del primero por el segundo, más el cuadrado del segundo.*

Los coeficientes de este trinomio resultante se pueden obtener también por medio de la expresión matemática para calcular el número de combinaciones de n objetos, en bloques de r, como se muestra a continuación:

$$(x+y)^2 = \binom{n}{n}x^2 + \binom{n}{n-1}xy + \binom{n}{n-2}y^2 = \binom{2}{2}x^2 + \binom{2}{2-1}xy + \binom{2}{2-2}y^2$$

$$= \binom{2}{2}x^2 + \binom{2}{1}xy + \binom{2}{0}y^2 = (1)x^2 + (2)xy + (1)y^2 = x^2 + 2xy + y^2$$

A los coeficientes

$$\binom{n}{n}, \binom{n}{n-1}, \binom{n}{n-2}, \ldots, \binom{n}{n-n}$$

de cada uno de los factores en que se descompone un binomio elevado a una potencia n se les llama *coeficientes binomiales de Newton*, y para obtenerlos sólo hay que aplicar la fórmula

$$\binom{n}{r} = \frac{n!}{r!(n-r)!}$$

lo cual elimina la necesidad de hacer una larga multiplicación o de aprenderse cualquier regla nemotécnica.

Isaac Newton
(1643-1727)

Fue un físico y matemático inglés que estableció la ley de la gravitación universal y las leyes de la mecánica clásica, además de los resultados que obtuvo al estudiar la naturaleza de la luz y la óptica. En matemáticas desarrolló el cálculo diferencial e integral, además del llamado teorema del binomio.

La obra más importante de Newton es *Philosophiae Naturalis Principia Mathematica.*

Por otro lado, para obtener los exponentes de cada uno de los términos del desarrollo, primero hay que observar que el número de términos que resulta de elevar un binomio a una potencia n es n + 1, por ejemplo en el caso de n = 2 se vio que el número de factores es 3, y luego hay que escribir los n + 1 productos entre *x* y *y* teniendo presente que la suma de sus exponentes debe de ser n:

$$(x+y)^n = \binom{n}{n}x^n y^0 + \binom{n}{n-1}x^{n-1}y^1 + \binom{n}{n-2}x^{n-2}y^2 + \ldots + \binom{n}{n-n}x^{n-n}y^n$$

Obsérvese que en todos los términos están presentes los productos de *x* y *y* elevados a potencias cuya suma es n ($x^n y^0$, $x^{n-1}y^1$, $x^{n-2}y^2$,... $x^{n-n}y^n$).

Como se ve, este procedimiento evita hacer cualquier multiplicación y es además una regla muy sencilla.

En el caso de n = 3 se tiene que

$$(x+y)^3 = \binom{3}{3}x^3y^0 + \binom{n}{3-1}x^{3-1}y^1 + \binom{n}{3-2}x^{3-2}y^2 + \binom{n}{3-3}x^0y^3$$

$$= x^3 + 3x^2y + 3xy^2 + y^3$$

Como es de esperar, este resultado coincide con la regla de que un binomio elevado al cubo es *el cubo del primero, más el triple del cuadrado del primero por el segundo, más el triple del primero por el cuadrado del segundo, más el cubo del segundo.*

Para n = 4 resulta que:

$$(x+y)^4 = \binom{4}{4}x^4y^0 + \binom{4}{4-1}x^{4-1}y^1 + \binom{4}{4-2}x^{4-2}y^2 + \binom{4}{4-3}x^{4-3}y^3 + \binom{4}{4-4}x^0y^4 =$$

$$= x^4 + 4x^3y + 6x^2y^2 + 4xy^3 + y^4$$

De aquí se ve que la regla para elevar un binomio a la cuarta potencia es: *el primero elevado a la cuarta potencia, más cuatro veces el producto del cubo del primero por el segundo, más seis veces el cuadrado del primero por el cuadrado del segundo, más cuatro veces el producto del primero por el cubo del segundo, más el segundo elevado a la cuarta potencia.*

Ejemplo 2.19. Obtener los factores del binomio $(-3x + 2y^2)^2$.

a) Usando la regla del producto notable para un binomio elevado al cuadrado se tiene que:

$$(-3x + 2y^2)^2 = (-3x)^2 + 2(-3x)(2y^2) + (2y^2)^2 = 9x^2 - 12xy^2 + 4y^4$$

b) Usando el teorema binomial:

$$(-3x + 2y^2)^2 = \binom{2}{2}x^2y^0 + \binom{2}{2-1}x^{2-1}y^1 + \binom{2}{2-2}x^{2-2}y^2$$

$$= \binom{2}{2}x^2y^0 + \binom{2}{1}x^1y^1 + \binom{2}{0}x^0y^2$$

$$= (1)(-3x)^2(2y^2)^0 + (2)(-3x)^1(2y^2)^1 + (1)(-3x)^0(2y^2)^2$$

$$= 9x^2 - 12xy^2 + 4y^4$$

Para elevar un binomio al cuadrado es más sencillo usar la regla ya conocida, sin embargo a medida que la potencia del binomio aumenta es más complicado obtener el resultado haciendo la multiplicación de binomios que usando el teorema binomial. Como ejemplo de esto, considérese el siguiente caso:

$$(3x-y^2)^4 = \binom{4}{4}x^4y^0 + \binom{4}{4-1}x^{4-1}y^1 + \binom{4}{4-2}x^{4-2}y^2 + \binom{4}{4-3}x^{4-3}y^3 + \binom{4}{4-4}x^{4-4}y^4$$

$$= (1)(3x)^4(-y^2)^0 + (4)(3x)^3(-y^2)^1 + (6)(3x)^2(-y^2)^2 + (4)(3x)^1(-y^2)^3 + (1)(3x)^0(-y^2)^4$$

$$= 81x^4 - 108x^3y^2 + 54x^2y^4 - 12xy^6 + y^8$$

La utilidad del teorema binomial en computación radica en que cuando se requiere crear un algoritmo que permita elevar un binomio a cierta potencia, sin el teorema esta tarea sería muy complicada y quizás no funcionaría para una potencia n arbitraria. Sin embargo, mediante el teorema binomial es relativamente sencillo saber el resultado, sin necesidad de desarrollar un algoritmo complicado.

Blaise Pascal
(1623-1662)

Fue un matemático, físico y filósofo francés cuyas contribuciones a las ciencias naturales y aplicadas incluyen la invención y construcción de calculadoras mecánicas, estudios sobre la teoría de la probabilidad, investigaciones sobre fluidos y el planteamiento de conceptos tales como la presión y el vacío.

Luego de una profunda experiencia religiosa en 1654, Pascal abandonó las matemáticas y la física para dedicarse a la filosofía.

2.5.2 Triángulo de Pascal

Otra aplicación en computación es el desarrollo de un programa para obtener el triángulo de Pascal, el cual tiene la siguiente forma:

$$
\begin{array}{ccccccccccc}
 & & & & & 1 & & & & & \\
 & & & & 1 & & 1 & & & & \\
 & & & 1 & & 2 & & 1 & & & \\
 & & 1 & & 3 & & 3 & & 1 & & \\
 & 1 & & 4 & & 6 & & 4 & & 1 & \\
1 & & 5 & & 10 & & 10 & & 5 & & 1 \\
 & & & & \vdots & & & & & &
\end{array}
$$

Hay que observar que en el triángulo de Pascal cada número mayor que uno es igual a la suma de los números que están a la izquierda y a la derecha del mismo en la línea inmediata anterior, por ejemplo, $4 = 1 + 3 = 3 + 1$ o bien $10 = 4 + 6 = 6 + 4$.

Usando el coeficiente binomial de Newton $\binom{n}{r}$ es posible obtener el triángulo de Pascal de la siguiente forma:

$$
\begin{array}{ccccccccccc}
 & & & & & \binom{0}{0} & & & & & \\
 & & & & \binom{1}{1} & & \binom{1}{0} & & & & \\
 & & & \binom{2}{2} & & \binom{2}{1} & & \binom{2}{0} & & & \\
 & & \binom{3}{3} & & \binom{3}{2} & & \binom{3}{1} & & \binom{3}{0} & & \\
 & \binom{4}{4} & & \binom{4}{3} & & \binom{4}{2} & & \binom{4}{1} & & \binom{4}{0} & \\
\binom{5}{5} & & \binom{5}{4} & & \binom{5}{3} & & \binom{4}{2} & & \binom{4}{1} & & \binom{4}{0}
\end{array}
$$

Obsérvese que los coeficientes del triángulo de Pascal no son otra cosa que los coeficientes del teorema binomial.

2.5.3 Sort de la burbuja (bubble sort)

El siguiente algoritmo permite ordenar un conjunto de N datos por el método de la burbuja.

```
I = 1
C = N
Mientras I > 0 hacer
Inicio
    I = 0
    C = C – 1
    X = 1
    Mientras X ≤ C hacer
    Inicio
        Si A[X] > A [X + 1] entonces
        Inicio
            T = A[X]
            A[X] = A[X + 1]
            A[X + 1] = T
            I = I + 1
        Fin
        X = X + 1
    Fin
Fin
```

En este algoritmo se tiene que:

A: Conjunto de datos a ordenar.
N: Número de datos del conjunto.
X: Subíndice.
I: Intercambios.
C: Comparaciones en cada pasada.
T: Variable para guardar un dato temporalmente mientras se hace el intercambio.

El mínimo de comparaciones que realiza el sort de la burbuja es (N–1) ya que el método termina cuando detecta que el arreglo o conjunto de datos está ordenado. El número de comparaciones en el peor de los casos (ya que depende de la colocación de los datos) es $\frac{N(N-1)}{2}$. Esta expresión matemática se obtiene al considerar que en cada pasada se llevan a cabo (C –1) comparaciones. En la primera pasada se tiene que (C – 1) = (N – 1), pero en cada pasada subsiguiente se disminuye una comparación de forma que resulta que el número de comparaciones en el peor de los casos es $\frac{N(N-1)}{2}$ lo cual se puede demostrar por medio de inducción matemática.

2.6 Resumen

En los métodos de conteo con frecuencia se presenta el problema de distinguir entre permutaciones y combinaciones. La diferencia principal es que en el caso de las permutaciones el orden de los elementos de los arreglos es importante, ya que dos arreglos con los mismos elementos pero colocados en posiciones distintas son permutaciones diferentes, sin embargo esos mismos dos arreglos son una sola combinación, ya que el orden en el caso de las combinaciones no interesa, sino solamente los elementos que conforman el arreglo. Otro aspecto que se debe tomar en consideración en el caso de las permutaciones es si los elementos de los arreglos se repiten o no, mientras que el caso de las combinaciones siempre se consideran sin repetición. Por último, también hay que considerar si el tamaño de los arreglos es menor o igual a n. En el siguiente ejemplo se ilustran estas diferencias.

Ejemplo 2.20. Considérese el conjunto A = {v, w, x, y, z}. Determinar el número de:

a) Permutaciones para arreglos de tamaño r = n con repetición.
b) Permutaciones para arreglos de tamaño r = n sin repetición.
c) Combinaciones para arreglos de tamaño r = n.
d) Permutaciones para arreglos de tamaño r = 2 con repetición.
e) Permutaciones para arreglos de tamaño r = 2 sin repetición.
f) Combinaciones para arreglos de tamaño r = 2.

Las respuestas son las siguientes:

a) $5^5 = 3125$

b) $5! = 120$

c) $\binom{n}{n} = 1$

En este caso (v, w, x, y, z) va en cualquier orden.

d)

vv	vw	vx	vy	vz
wv	ww	wx	wy	wz
xv	xw	xx	xy	xz
yv	yw	yx	yy	yz
zv	zw	zx	zy	zz

$5^2 = 25$

e)

	vw	vx	vy	vz
wv		wx	wy	Wz
xv	xw		xy	Xz
yv	yw	yx		Yz
zv	zw	zx	zy	

$$\frac{5!}{(5-2)!} = 20$$

f)

	vw	vx	vy	vz
		wx	wy	wz
			xy	xz
				yz

$$\frac{5!}{2!(5-2)!} = 10$$

En la tabla siguiente se muestran las diferentes expresiones matemáticas que se utilizan, de acuerdo con las características del conteo.

Características del conteo	Expresión matemática
Permutaciones para arreglos de tamaño r donde r = n con repetición.	$P(n, r) = n^r$
Permutaciones para arreglos de tamaño r = n sin repetición.	$P(n, r) = n!$
Permutaciones para arreglos de tamaño r = n sin repetición, en forma circular.	$P(n, r) = (n-1)!$
Permutaciones para arreglos de tamaño r < n sin repetición.	$P(n, r) = \frac{n!}{(n-r)!}$
Permutaciones de n objetos de los cuales t_1 son de un tipo, t_2 son de otro tipo distinto y t_k son del k-ésimo tipo, donde $t_1+t_2+\ldots+t_k = n$.	$P(n, k) = \frac{n!}{t_1!\, t_2! \cdots t_k!}$
Combinaciones para arreglos de tamaño r = n.	$\binom{n}{r} = \frac{n!}{r!(n-r)!} = \frac{n!}{n!(n-n)!} = 1$
Combinaciones para arreglos de tamaño r < n.	$\binom{n}{r} = \frac{n!}{r!(n-r)!}$

Los métodos de conteo son útiles en todas las ramas de las ciencias, y en particular en las ciencias de la computación ya que la cantidad de información que procesa la computadora es extremadamente grande y la exigencia en la velocidad de procesamiento es fundamental. La velocidad de procesamiento depende tanto del hardware como del software, es por ello que cada día salen equipos cada vez más rápidos y también por lo que en forma paralela se busca optimizar el software, proceso en el cual los métodos de conteo tienen una participación destacada para mejorar cada vez más los algoritmos.

2.7 Problemas

2.1. La compañía Hewlett Packard (HP) produce computadoras con:

- 3 colores diferentes (negro, gris, plateado).
- 2 tipos de pantalla (plana y convencional).
- 2 tipos de procesador.
- 3 capacidades de memoria principal.
- 4 capacidades de disco duro.

a) ¿Cuántas computadoras diferentes puede producir la compañía?

b) ¿Cuántas computadoras distintas de color "gris" es posible fabricar?

c) ¿Cuántas computadoras de color negro y pantalla plana se pueden fabricar?

2.2. Una compañía produce refrescos de 4 sabores diferentes (naranja, fresa, toronja y piña). Si para cada uno de ellos se tienen 3 presentaciones distintas (lata, botella de cristal, botella de plástico), además de que las presentaciones en botella de vidrio y plástico pueden tener 3 capacidades diferentes (250 ml, 500 ml y un litro) y las de lata solamente de 250 ml, ¿cuántos tipos diferentes en total produce la compañía?

2.3. En un lenguaje de programación los identificadores pueden comenzar con una letra (L), seguida o no de hasta 5 caracteres que pueden ser letras o dígitos (D). Considerando que existen 27 letras diferentes y 10 dígitos:

a) ¿Cuántos identificadores diferentes se pueden formar?

b) ¿Cuántos identificadores distintos de longitud máxima seis se pueden formar, si los identificadores comienzan por una letra, seguida de un número o letra con repetición?

2.4. Los estudiantes de una Universidad deberán tomar en este próximo semestre 6 materias diferentes (Matemáticas, Física, Administración, Fundamentos de programación, Matemáticas para computación y Ética). Si las materias de Matemáticas y Física se ofrecen con 3 maestros distintos, las de Administración y Ética con dos maestros diferentes y las dos materias restantes solamente las imparte un maestro diferente, ¿de cuántas maneras se puede elaborar un horario para el próximo semestre?

2.5. Un examen consta de 20 preguntas, 9 de éstas son de opción múltiple y cada una tiene 4 opciones diferentes de las cuales solamente una es correcta. Las restantes 11 preguntas son de "Falso" o "Verdadero".

a) ¿De cuántas maneras distintas se puede contestar el examen?

b) ¿De cuántas maneras diferentes se puede contestar el examen, de forma que todas las respuestas sean incorrectas?

c) ¿De cuántas maneras se puede contestar el examen, de forma que todas las respuestas sean correctas?

2.6. El examen de admisión de la carrera de Licenciatura en Informática consta de 80 preguntas de opción múltiple.

a) Si cada una de las 80 preguntas tiene 5 opciones distintas, ¿de cuántas maneras puede contestar un alumno que presenta el examen de admisión?

b) Si 40 de los reactivos tienen 4 opciones distintas y las otras 40 preguntas son de "Falso" o "Verdadero", ¿de cuántas maneras diferentes puede contestar el alumno que presenta el examen de admisión?

2.7. Se aplica un examen de opción múltiple que consta de 10 preguntas. Cada pregunta tiene 5 opciones diferentes, pero solamente una de esas opciones es correcta.

a) ¿De cuántas maneras diferentes es posible contestar el examen?

b) ¿De cuántas maneras diferentes se puede contestar el examen y que todas las respuestas estén equivocadas?

c) ¿De cuántas maneras se puede contestar el examen y que todas las respuestas sean correctas?

d) ¿De cuántas maneras es posible contestar el examen y que la calificación sea de 70%?

2.8. El examen de matemáticas para computación consta de 20 preguntas, y cada pregunta consta de 5 opciones diferentes pero solamente una de ellas es correcta.

a) ¿De cuántas maneras diferentes es posible contestar el examen?

b) ¿De cuántas formas distintas se puede contestar el examen para sacar una calificación de 100%?

c) ¿De cuántas maneras diferentes se puede contestar el examen y sacar 0% de calificación?

d) ¿De cuántas maneras es posible contestar el examen y que la calificación sea mínimo de 70%?

2.9. En el sistema numérico "Trinario" solamente se admiten como dígitos válidos 0, 1 y 2 para formar cantidades.

a) ¿Cuántas cantidades de cuatro cifras se pueden formar en el sistema trinario?

b) ¿Cuántas cantidades de dos cifras se pueden formar?

c) Elaborar un árbol que muestre las 9 cantidades de dos cifras que se pueden formar en el sistema trinario.

2.10. En el sistema hexadecimal se utilizan los dígitos conocidos (0, 1, 2, ..., 9) y las primeras letras del alfabeto (A, B, C, D, E y F) para representar cantidades.

a) ¿Cuántas cantidades diferentes de cuatro cifras se pueden formar en el sistema hexadecimal?

b) ¿Cuántas cantidades distintas de siete cifras se pueden representar, si se desea que todas esas cifras comiencen con la letra F y terminen con la letra D?

2.11. Considérese el conjunto de las vocales (A, E, I, O,U).

a) ¿De cuántas maneras se pueden acomodar?

b) ¿Cuántas de esas permutaciones comienzan con la letra "E"?

c) ¿Cuántas permutaciones diferentes de 4 letras se pueden formar?

d) ¿Cuántas permutaciones diferentes de 3 letras se pueden formar?

2.12. Se tienen seis letras (A, B, C, D, E, F).

a) ¿De cuántas maneras diferentes se pueden ordenar?

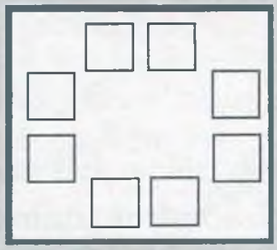
b) Si se desea que las letras B y F siempre estén juntas, ¿de cuántas maneras se pueden acomodar?

2.13. Se tiene una gran mesa circular y 10 computadoras diferentes (A, B, C, ..., J).

a) ¿De cuántas maneras distintas se pueden colocar las 10 computadoras alrededor de la mesa?

b) Si se desea que 3 de ellas siempre estén juntas ¿de cuántas maneras se pueden colocar las computadoras alrededor de la mesa circular?

2.14. ¿De cuántas formas diferentes es posible colocar 8 computadoras en una gran mesa de forma cuadrada, como se indica en la figura?



2.15. Se van a plantar en círculo 10 árboles.

a) ¿De cuántas formas diferentes se pueden plantar?

b) Si de esos 10 árboles uno es cedro, 4 eucaliptos y 5 pinos, y se desea que queden seguidos de acuerdo a la clase de árbol, ¿de cuántas maneras diferentes se pueden acomodar formando el círculo?

2.16. Se conectarán 18 computadoras, para formar una red circular.

a) ¿De cuántas formas distintas se pueden conectar?

b) Si 8 de ellas son HP, ¿de cuántas maneras distintas se pueden conectar las 18 computadoras, si se desea que las computadoras HP estén una seguida de otra?

2.17. Se invitó a los ganadores de medalla de "Oro", "Plata" y "Bronce" de "*Salto de garrocha*", "*Lanzamiento de jabalina*", "*Lanzamiento de bala*" y "*Salto de altura*" a un evento para entregarles su medalla respectiva.

a) ¿De cuántas maneras se pueden sentar en una fila de 12 butacas si no se ponen restricciones?

b) ¿De cuántas maneras se pueden sentar si se acomodan por prueba (juntos los de salto de altura, juntos los de salto con garrocha, etc., sin importar qué prueba se coloca primero)?

c) ¿De cuántas maneras se pueden sentar si se acomodan por metal (juntos los de medalla de oro, juntos los de medalla de plata, etc., sin importar si los bloques están al principio, en medio o al final)?

d) ¿De cuántas maneras se pueden sentar si se acomodan primero los de medalla de oro, después los de plata y finalmente los de bronce sin importar la prueba?

e) ¿De cuántas maneras se pueden sentar si primero se colocan los de salto de garrocha, después los de lanzamiento de jabalina, en la siguiente posición los de lanzamiento de bala y finalmente los de salto de altura.

2.18. Un entrenador de fútbol soccer tiene una plantilla de 22 jugadores.

a) ¿De cuántas maneras diferentes puede conformar su equipo de 11 titulares, considerando que todos pueden jugar en cualquier posición?

b) ¿De cuántas maneras diferentes puede estructurar el equipo de 11 titulares, si la plantilla tiene 3 porteros, 6 defensas, 8 medios y 5 delanteros? ¿Considerando que el equipo de titulares debe tener un portero, 4 defensas, 3 medios y 3 delanteros?

2.19. Encontrar las permutaciones de la palabra TENDERETE.

a) Con repetición.

b) Sin repetición.

2.20. ¿De cuántas maneras se pueden ordenar las letras de la palabra MININOS?

a) Sin repetición.

b) Con repetición

2.21. El Departamento de Sistemas y Computación del Instituto Tecnológico de Morelia tiene 4 catedráticos con grado de Doctor, 19 con grado de Maestría y 5 con Licenciatura. Se desea formar comités para que participen en los exámenes de titulación. Los comités deben estar integrados por 4 elementos (Presidente, Secretario, Vocal y Vocal suplente).

a) ¿Cuántos comités se pueden formar si no se pone restricción alguna?

b) ¿Cuántos comités se pueden formar si se desea que dichos comités estén integrados por un Doctor, dos Maestros en Ciencias y un profesionista con grado de Licenciatura?

c) ¿Si todos deben de tener por lo menos grado de maestría?

d) ¿Si uno de los Doctores es Juan Manuel García y debe estar en todos los comités y los tres elementos restantes se seleccionan sin restricción?

2.22. Una agencia automotriz tiene 5 automóviles rojos, 4 grises y 3 azules.

a) ¿Cuántas formas diferentes de seleccionar 6 automóviles se puede obtener si se desea que haya 2 automóviles de cada color?

b) ¿De cuántas maneras se puede seleccionar un lote de 6 automóviles si se desea que en ese lote estén incluidos 3 automóviles rojos, 2 grises y un azul?

2.23. Policía y tránsito del estado de Michoacán desea comprar 40 patrullas para actualizar su parque vehicular. Comprará esos 40 automóviles a una compañía automotriz que tiene en existencia 58 automóviles, de los cuales 12 tienen algún defecto de fabricación.

a) ¿De cuántas maneras se pueden seleccionar los 40 automóviles de forma que en dicha selección estén incluidos 9 automóviles defectuosos?

b) ¿De cuántas maneras se pueden seleccionar los 40 automóviles de forma que en dicha selección estén incluidos por lo menos 6 defectuosos?

2.24. En una clase de matemáticas para computación se formarán equipos de 5 personas de un grupo de 32 alumnos, de los cuales 18 son hombres y 14 mujeres. De cuántas maneras se pueden formar los comités considerando que:

a) No existe restricción alguna.
b) Deben tener más hombres que mujeres.
c) Deben tener 2 mujeres por lo menos.
d) Deben tener más mujeres que hombres.
e) Deben estar integrados solamente por mujeres.

2.25. Se desea distribuir un grupo de 40 alumnos de Ingeniería en Sistemas Computacionales en cinco talleres diferentes: Redes, Páginas web, Sistemas operativos, Bases de datos y Programación ensamblador, de tal manera que cada uno de los talleres tenga 8 alumnos. ¿De cuántas maneras distintas podrán distribuir a los alumnos en los diferentes talleres?

2.26. Se plantarán en línea 12 árboles, de los cuales 4 son pinos, 3 robles y 5 fresnos.

a) ¿De cuántas formas diferentes se pueden plantar si no se pone ninguna restricción?
b) ¿De cuántas maneras se pueden plantar, si se requiere que los árboles de una misma clase queden juntos?

2.27. Desarrollar los binomios de cada uno de los siguientes incisos, usando para ello el teorema binomial.

a) $(2x^2 - y)^5$

b) $\left(\frac{1}{2}a + \frac{3}{4}b\right)^3$

2.28. Desarrollar los binomios de cada uno de los siguientes incisos, usando para ello el teorema binomial.

a) $(-4x^3 - 2y)^3$
b) $(x^2 + 3y^2)^4$
c) $\left(\frac{1}{3}a + \frac{2}{5}b^2\right)^2$
d) $\left(-\frac{4}{5}a^2 + \frac{1}{3}b^3\right)^4$
e) ¿Cuál es la regla en palabras para elevar al cubo un binomio?
f) ¿Cuál es la regla en palabras para elevar a la cuarta potencia un binomio?

2.29. Considérese el siguiente programa:

```
a=0
x=4
Mientras a<5 hacer
   Inicio
      a = a+3
         b=13
         y=2
         Mientras b >= 4 hacer
               Inicio
                  x=3*x–y
                  b=b–2
                  y=y-4
            Fin
   Fin
   Imprimir a,b,x,y
```

a) ¿Cuántas veces se ejecuta la instrucción x=3*x–y.

b) ¿Cuáles son los resultados de a, b, x, y que se imprimen al final.

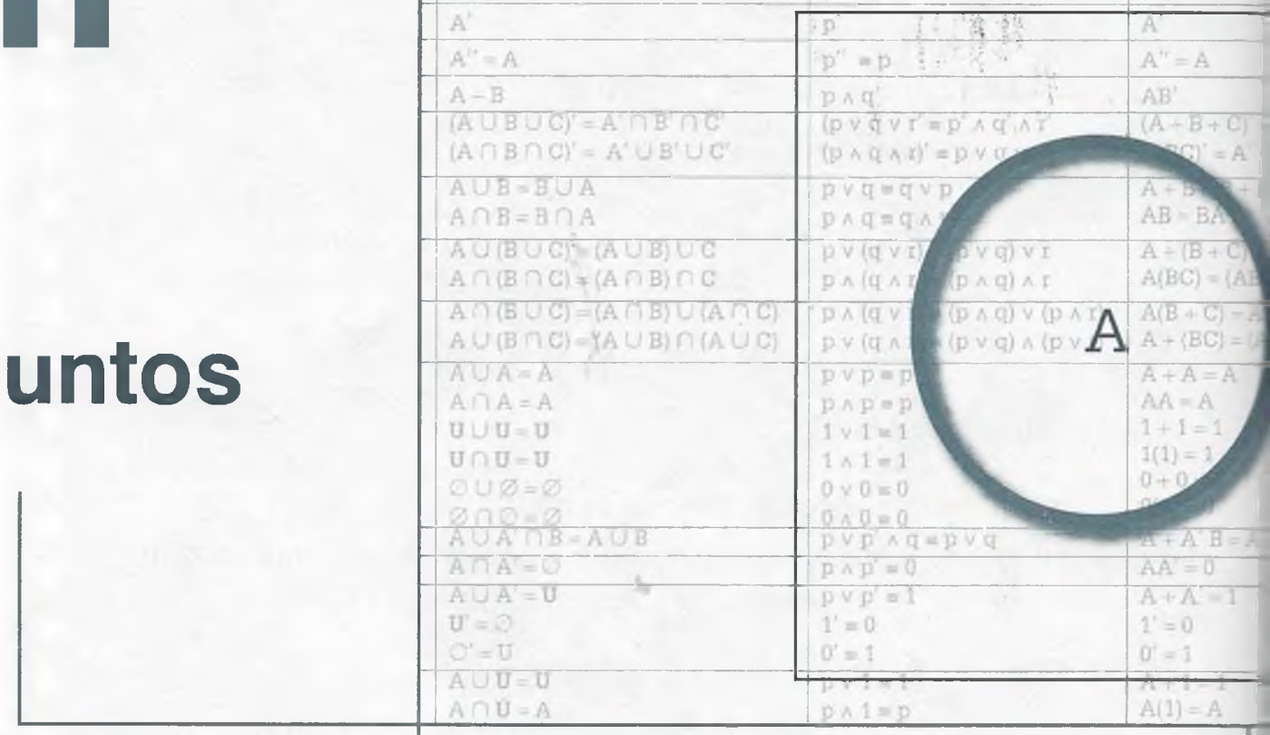
2.30. Considérese el siguiente programa.

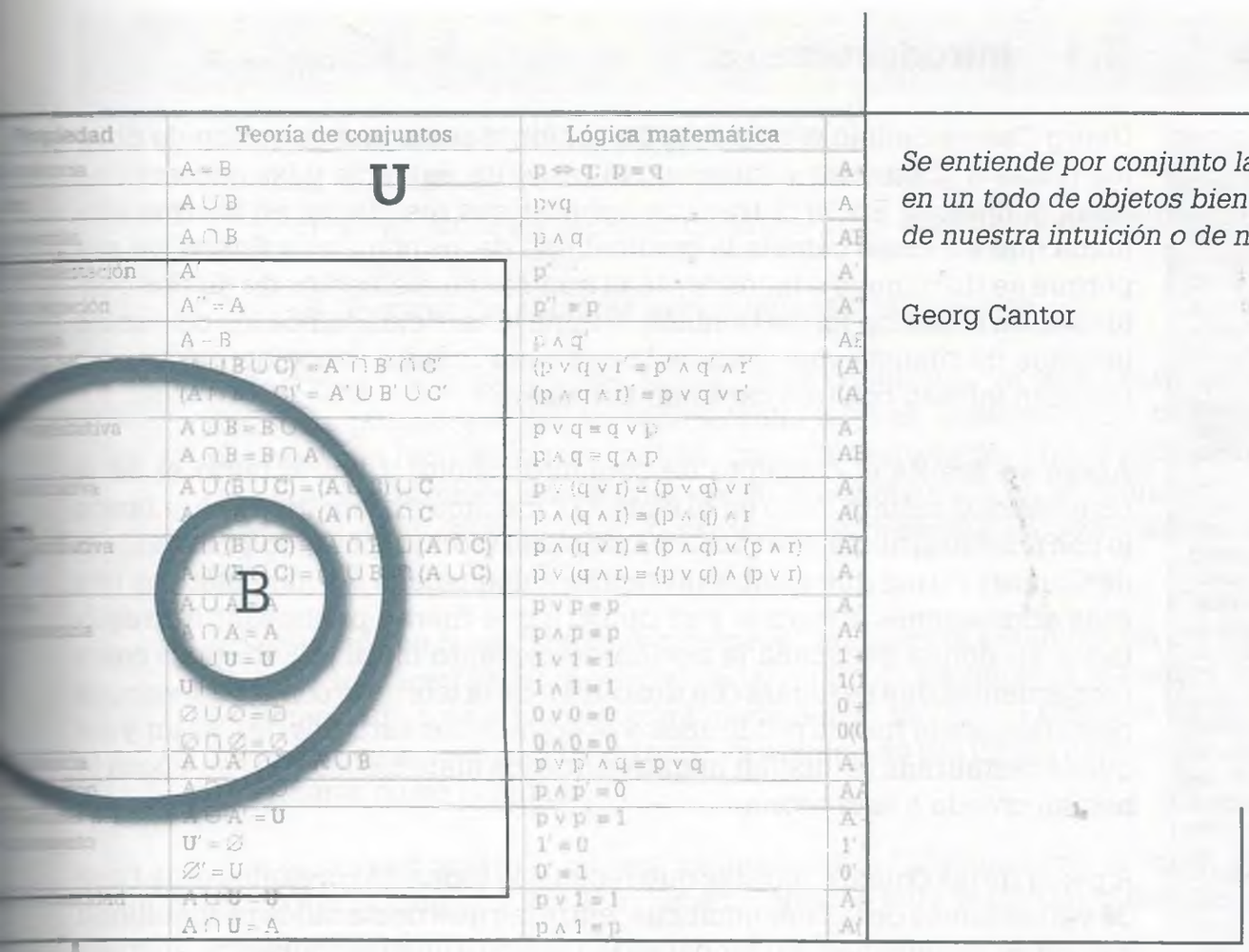
```
a=1
x=4
Mientras x >= a hacer
   Inicio
      b=5
      y=8
      Mientras y > b hacer
         Inicio
            c=0
            w=7
            Mientras c < w hacer
               Inicio
                  Imprimir ('Hola', a*b*w)
                  w=w–2
               Fin
         b=b+1
      Fin
   x=x–1
Fin
```

¿Cuántas veces se ejecuta la instrucción: **Imprimir ('Hola ', a*b*w)**?

CAPÍTULO 3

Conjuntos





Se entiende por conjunto la agrupación en un todo de objetos bien diferenciados de nuestra intuición o de nuestra mente.

Georg Cantor

Objetivos

- Proporcionar las bases de teoría de conjuntos para una mejor comprensión de los capítulos: lógica matemática, álgebra booleana, relaciones, grafos, árboles e introducción a los lenguajes formales.
- Aprender a representar conjuntos finitos e infinitos, subconjuntos y operaciones entre conjuntos por medio de expresiones matemáticas o bien por medio de diagramas de Venn.
- Identificar la similitud entre teoría de conjuntos, lógica matemática y álgebra booleana con la finalidad de aprovechar los conocimientos de conjuntos en esas áreas afines.

3.1 Introducción

Georg Cantor
(1846-1918)

Fue un matemático alemán creador de la teoría de conjuntos que demostró, entre otras cosas, que los números reales son "más numerosos" que los números naturales. Aún más, la teoría de Cantor supone la existencia de una "infinidad de infinitos".

Al principio la teoría de los números transfinitos de Cantor fue considerada como no intuitiva y encontró resistencia de parte de los matemáticos contemporáneos como Leopold Kronecker y Henri Poincaré, y más tarde de Hermann Weyl y L.E.J. Brouwer, mientras que Ludwig Wittgenstein planteó objeciones filosóficas. Por su parte algunos teólogos cristianos vieron el trabajo de Cantor como un desafío a la singularidad de la infinitud absoluta de la naturaleza de Dios.

Las objeciones fueron feroces: Poincaré se refiere a las ideas de Cantor como una "grave enfermedad" que infecta a la disciplina de las matemáticas, y la oposición pública de Kronecker y los ataques personales incluyen la descripción de Cantor como un "charlatán científico", un "renegado" y "corruptor de la juventud". En alguna época los episodios recurrentes de depresión de Cantor desde 1884 hasta el final de su vida se achacaron a la actitud hostil de muchos de sus contemporáneos, sin embargo estos episodios pueden ser vistos ahora como probables manifestaciones de un trastorno bipolar.

La dura crítica ha ido acompañada de elogios. David Hilbert defendió a Cantor con su ya famosa declaración: "Nadie podrá expulsarnos del Paraíso que Cantor ha creado."

Georg Cantor definió el concepto de conjunto como una colección de objetos reales o abstractos e introdujo el conjunto potencia y las operaciones entre conjuntos. En 1872 trató de publicar sus resultados en los que afirmaba que así como cambia la cardinalidad de los conjuntos finitos, ya sea porque se disminuye o incrementa el número de elementos de dichos conjuntos, de la misma forma también cambia la cardinalidad de los conjuntos infinitos de manera que para cada conjunto infinito conocido existe otro también infinito con una cardinalidad mayor.

Ahora se acepta el concepto de conjunto infinito y por lo tanto el de la cardinalidad infinita, pero en el siglo XIX muchos matemáticos de la época lo consideraron absurdo e incluso Kronecker (contemporáneo y compañero de Cantor) afirmó que Cantor pretendía enloquecer a las matemáticas con esas afirmaciones absurdas y se opuso a que fueran publicados los resultados en donde explicaba la noción de conjunto infinito. Esto trajo como consecuencia que se mirara con desconfianza la teoría de conjuntos, aunque posteriormente fueron publicados y aceptados los estudios de Cantor y así quedó restaurada la imagen negativa que los matemáticos de esa época le habían creado a esta teoría.

A pesar de las críticas iniciales que recibió, la teoría de conjuntos es la base de varias ramas de las matemáticas, entre las que destacan la probabilidad y la lógica matemática. En probabilidad permite ilustrar conceptos abstractos que sería imposible explicar sin el apoyo de conjuntos, y en lógica matemática la teoría de conjuntos proporciona las herramientas necesarias como axiomas, postulados, leyes y reglas de inferencia para probar relaciones y teoremas complejos por medio del método deductivo. Pero aún más, la teoría de conjuntos es la base de las ciencias de la computación ya que sirve de fundamento del álgebra booleana, de los lenguajes, de los autómatas, de las relaciones, de las bases de datos, de los grafos, de las redes y de los árboles, entre otros temas.

3.2 Concepto de conjunto

Un conjunto es una colección *bien definida* de objetos llamados elementos o miembros del conjunto.

En esta definición la frase *bien definida* es esencial para determinar si un grupo de personas o una colección de objetos es o no un conjunto, ya que para que una colección de objetos se considere como un conjunto no debe haber ambigüedad ni subjetividad.

Ejemplo 3.1. Considérense los siguientes enunciados:

a) La colección de pizarrones azules.
b) El grupo de alemanes entre 20 y 30 años.
c) El grupo de los *mejores maestros* de la especialidad de sistemas computacionales.
d) El grupo de *alumnas más guapas* de informática.

Los incisos a y b se pueden tomar como conjuntos, ya que están bien definidos puesto que por un lado el color azul es universal para todos y por tanto es fácil determinar si un pizarrón pertenece o no al conjunto, y por el otro también es sencillo ubicar a una persona en el conjunto de alemanes entre 20 y 30 años, conociendo obviamente su nacionalidad y edad.

En el inciso c la frase "mejores maestros" no permite establecer si un determinado maestro pertenece o no al conjunto de los mejores maestros de la especialidad de sistemas computacionales, ya que el término "mejor maestro" es subjetivo. Por lo tanto, el enunciado del inciso c no se puede considerar como conjunto.

Con el inciso d ocurre lo mismo, ya que la frase "alumnas más guapas" es ambigua puesto que existen distintos gustos para catalogar a una chica como guapa o no.

Los conjuntos se indican por medio de una letra mayúscula y los elementos de un conjunto por medio de letras minúsculas, números o combinación de ambos. Los elementos se colocan entre llaves, { }, separados por comas.

Ejemplo 3.2. El conjunto B tiene como elementos a las letras de la palabra "mandarina" y éste se representa como:

$$\begin{aligned} B &= \{m, a, n, d, a, r, i, n, a\} \\ &= \{m, a, n, d, r, i\} \\ &= \{n, r, a, i, m, d\} \end{aligned}$$

Como se ve, en un conjunto se pueden eliminar los elementos repetidos además de que el orden en que se listen dichos elementos no es importante.

Se dice que un elemento x pertenece a un conjunto C si se verifica que el elemento se encuentra dentro del conjunto, y para expresar la pertenencia se tiene la siguiente notación:

$x \in C$ significa que x es elemento del conjunto C.
$x \notin C$ significa que x no es elemento del conjunto C.

Ejemplo 3.3. Sea el conjunto

$$A = \{1, 3, 5, 7, 9\}$$

Por lo tanto se tiene que $3 \in A$ pero $6 \notin A$.

Algunas veces es imposible o inconveniente listar los elementos de un conjunto entre llaves, entonces en lugar de esto se utiliza lo que se conoce como *notación abstracta*:

$$A = \{x \mid P(x)\}$$

que se lee como A es el conjunto de las x, tal que cumple con la condición (o condiciones) P(x).

Ejemplo 3.4. Sea el conjunto B que tiene como elementos a todas las palabras del idioma español que comienzan con la letra "e". En este caso no es imposible hacer el listado de todos los elementos del conjunto, sin embargo sí es inconveniente ya que el número de elementos es considerable.

En lugar del listado, el conjunto se puede expresar de la siguiente manera:

$B = \{x \mid x$ es una palabra del idioma español que comienza con "e"$\}$

Por otro lado, sea el conjunto C que tiene como elementos a todos los números reales comprendidos entre 2 y 3. En este caso es imposible listar los elementos del conjunto ya que hay una cantidad infinita de ellos; en lugar de esto el conjunto se puede indicar de la siguiente manera:

$C = \{x \mid x$ es un número real entre 2 y 3$\}$

Aunque es válido especificar las características de los elementos de un conjunto con palabras, como se hizo anteriormente, existen conjuntos importantes que se pueden usar para compactar la información. Algunos de los conjuntos que más se utilizan en matemáticas son los siguientes:

$\mathbf{N}$ = *Conjunto de los números naturales*
$= \{1, 2, 3, \ldots\}$

$\mathbf{Z}^+$ = *Conjunto de los números enteros no negativos*
$= \{0, 1, 2, 3, \ldots\}$

$\mathbf{Z}$ = *Conjunto de los números enteros*
$= \{\ldots -2, -1, 0, 1, 2, 3, \ldots\}$

$\mathbf{Q}$ = *Conjunto de los números racionales*
$= \{\frac{a}{b} \mid a, b \in \mathbf{Z};\ b \neq 0\}$

$\mathbf{R}$ = *Conjunto de los números reales*

$\mathbf{C}$ = *Conjunto de los números complejos*
$= \{x + yi \mid x, y \in \mathbf{R};\ i^2 = -1\}$

$\mathbf{U}$ = *Conjunto universo*

$\varnothing$ = *Conjunto vacío*

Usando esta información, el conjunto

$\mathbf{C} = \{x \mid x$ es un número real entre 2 y $3\}$

también se puede expresar como:

$\mathbf{C} = \{x \mid x \in \mathbf{R};\ 2 < x < 3\}$

En este caso hay que observar que son dos las condiciones que tiene que cumplir x para pertenecer al conjunto C:

1) $x \in \mathbf{R}$ Que x sea un número real.

2) $2 < x < 3$ Que x esté entre 2 y 3.

Además de esto, se acostumbra separar con un punto y coma (;) cada una de las condiciones que se deben de satisfacer para que un elemento x pertenezca a un conjunto dado.

3.3 Subconjuntos

Si todos los elementos de A también son elementos de B, se dice que A es subconjunto de B o que A está contenido en B, y esto se denota como

$$A \subseteq B$$

Si A no es subconjunto de B se escribe:

$$A \not\subset B$$

Por otro lado, se dice que dos conjuntos A y B son iguales si tienen los mismos elementos, es decir, si se cumple que

$$A \subseteq B \quad \text{y} \quad B \subseteq A$$

Sean

$$A = \{\text{Rojo, Amarillo, Azul}\}$$
$$B = \{\text{Azul, Rojo, Amarillo}\}$$

entonces

$$A = B$$

Ejemplo 3.5. Considérense los siguientes conjuntos:

$A = \{x \mid x \in \mathbf{Z};\ 10 \leq x \leq 100\}$
$B = \{2, 3, 5, 11, 12, 15, 21, 30, 45, 82\}$
$C = \{12, 15, 45\}$

Entonces se tiene que:

$C \subseteq B$	$A \not\subset B$
$C \subseteq A$	$A \not\subset C$
$B \not\subset A$	$B \not\subset C$

Aplicando la definición de subconjunto se obtiene que

1) Todo conjunto A es un subconjunto de sí mismo:

$$A \subseteq A$$

2) **El conjunto vacío (∅) es subconjunto de todos los conjuntos y en particular de él mismo:**

$$\emptyset \subseteq A$$
$$\emptyset \subseteq \mathbf{U}$$
$$\emptyset \subseteq \emptyset$$

3) **Todos los conjuntos son subconjuntos del conjunto universo (U):**

$$A \subseteq \mathbf{U}$$
$$\emptyset \subseteq \mathbf{U}$$
$$\mathbf{U} \subseteq \mathbf{U}$$

Por otro lado, si A es un conjunto entonces al conjunto de todos los subconjuntos de A se le llama conjunto potencia de A y se indica como P(A).

Ejemplo 3.6. Sea el conjunto

$$A = \{a, b, c\}$$

Entonces el conjunto potencia de A es:

$$P(A) = \{\emptyset, \{a\}, \{b\}, \{c\}, \{a,b\}, \{a,c\}, \{b,c\}, \{a,b,c\}\}$$

El número de subconjuntos del conjunto A está dado por

$$|P(A)| = 2^n$$

donde n es el número de elementos del conjunto A.

En el caso del ejemplo 3.6 se tiene que

$$|P(A)| = 2^3 = 8$$

3.4 Diagramas de Venn

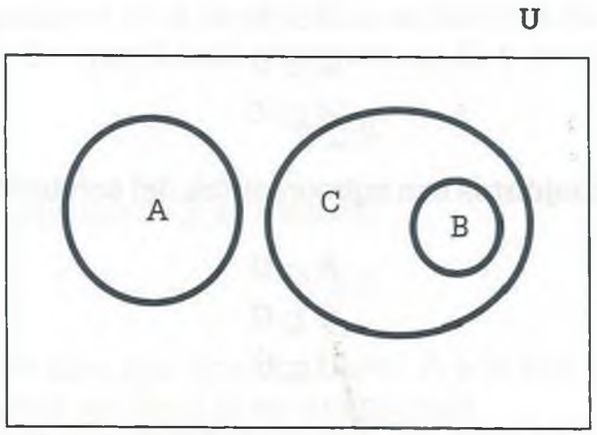
Los diagramas de Venn son representaciones gráficas para mostrar la relación entre los elementos de los conjuntos. Por lo general cada conjunto se representa por medio de un círculo, óvalo o rectángulo, y la forma en que se entrelazan las figuras que representan a los conjuntos muestra la relación que existe entre los elementos de los respectivos conjuntos.

John Venn
(1834-1923)

Fue un matemático y filósofo británico que creó los diagramas que llevan su nombre y que son usados en teoría de conjuntos, probabilidad, lógica, estadística y ciencias de la computación.

Estudiante y más tarde profesor en la Universidad de Cambridge, Venn dio a conocer sus diagramas en 1880 con la publicación de su trabajo *De la representación mecánica y diagramática de proposiciones y razonamientos* en el *Philosophical Magazine and Journal of Science*. Aunque la primera forma de representación geométrica de silogismos lógicos se debe a Gottfried Leibniz y luego fue ampliada por George Boole y Augustus De Morgan, el método de Venn superó en claridad y sencillez a los sistemas de representación anteriores, hasta el punto de convertirse con el tiempo en un nuevo estándar.

El siguiente esquema es un ejemplo de diagrama de Venn.



Algunas afirmaciones de este diagrama de Venn son:

$A \subseteq \mathbf{U}$ $\quad C \subseteq \mathbf{U}$ $\quad \mathbf{U} \not\subset A$

$B \subseteq C$ $\quad B \subseteq \mathbf{U}$ $\quad \mathbf{U} \not\subset C$

$A \not\subset C$ $\quad B \not\subset A$ $\quad \mathbf{U} \not\subset B$

$C \not\subset B$ $\quad C \not\subset A$

3.5 Operaciones y leyes de conjuntos

Así como es posible llevar a cabo operaciones entre números, también se pueden realizar operaciones con conjuntos y éstas se aplican en prácticamente todos los temas de las ciencias de la computación.

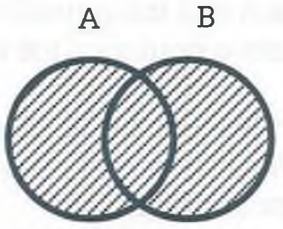
Por otro lado, las operaciones con conjuntos se pueden ilustrar por medio de un diagrama de Venn con el fin de observar más claramente la relación entre los conjuntos.

3.5.1 Unión ($A \cup B$)

La unión del conjunto A y el conjunto B es el conjunto que contiene a todos los elementos del conjunto A y del conjunto B:

$$A \cup B = \{x \mid x \in A \text{ ó } x \in B\}$$

El siguiente diagrama ilustra la definición:



$A \cup B$

Ejemplo 3.7. Sean los conjuntos:

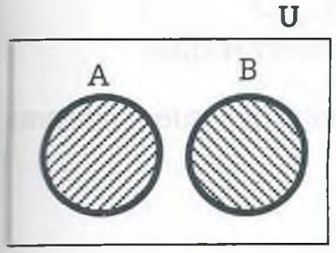
$$A = \{1, 2, 3, 6, 7, 8\}$$

$$B = \{x \mid x \in \mathbf{Z}^+;\ x \leq 12;\ x \text{ es par}\}$$

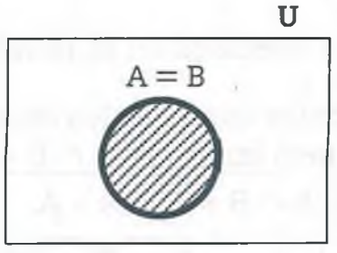
Aplicando la definición de unión de conjuntos se tiene que:

$$A \cup B = \{1, 2, 3, 4, 6, 7, 8, 10, 12\}$$

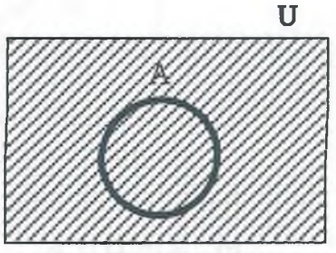
Considérense los siguientes diagramas de Venn:



$A \cup B$



$A \cup B$



$A \cup \mathbf{U}$

A partir de éstos se puede determinar que la ley conmutativa y la ley de idempotencia se cumplen para el caso de la unión, y que la unión del conjunto universo con otro conjunto es el conjunto universo:

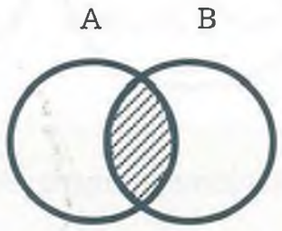
$A \cup B = B \cup A$	Ley conmutativa
$A \cup A = A$	Ley de idempotencia ($A = B$)
$A \cup \mathbf{U} = \mathbf{U}$	

3.5.2 Intersección ($A \cap B$)

La intersección del conjunto A y el conjunto B es el conjunto que contiene a todos los elementos que son comunes a los conjuntos A y B:

$$A \cap B = \{x \mid x \in A;\ x \in B\}$$

El siguiente diagrama ilustra la definición:



$A \cap B$

Ejemplo 3.8. Sean los conjuntos:

$A = \{1, 2, 3, 6, 7, 8\}$
$B = \{x \mid x \in \mathbf{Z}^+;\ x \leq 12;\ x \text{ es par}\}$

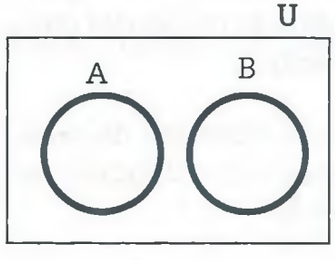
Aplicando la definición de intersección de conjuntos se tiene que:

$$A \cap B = \{2, 6, 8\}$$

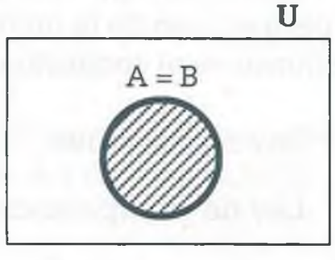
A partir de la definición de intersección es posible observar que:

a) Si A y B son conjuntos disjuntos (es decir, conjuntos que no tienen elementos comunes) entonces $A \cap B = \varnothing$.
b) Si $A = B$ entonces $A \cap B = A \cap A = A$.
c) $A \cap \mathbf{U} = A$
d) $A \cap \varnothing = \varnothing$.

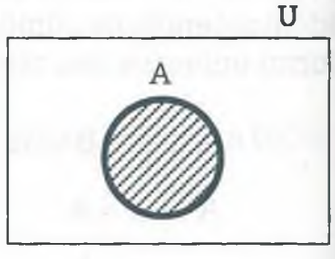
Esto se muestra en los siguientes diagramas:



$A \cap B = \varnothing$



$A \cap A = A$



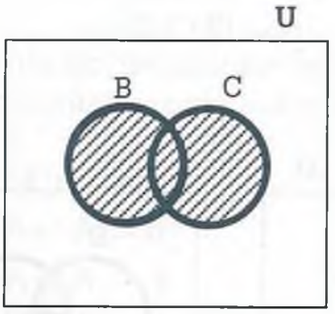
$A \cap \mathbf{U} = A$

3.5.3 Ley distributiva

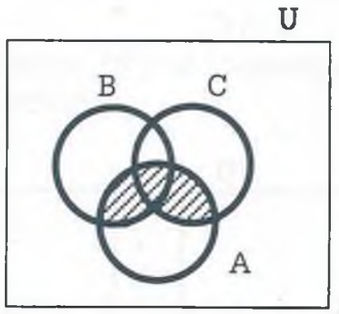
Dados tres conjuntos arbitrarios A, B y C, se puede ver que se cumple la siguiente ley distributiva en la que intervienen la unión y la intersección de conjuntos:

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

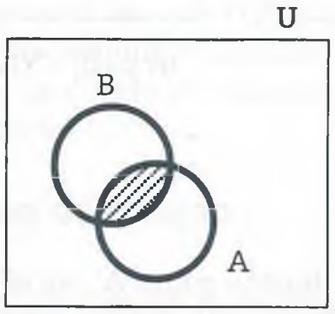
Los siguientes diagramas de Venn ilustran la validez de esta ley:



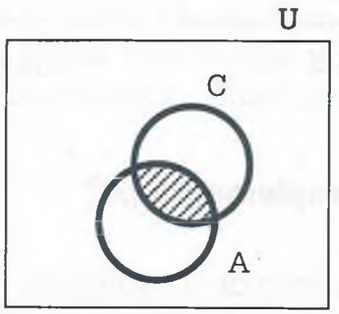
$B \cup C$



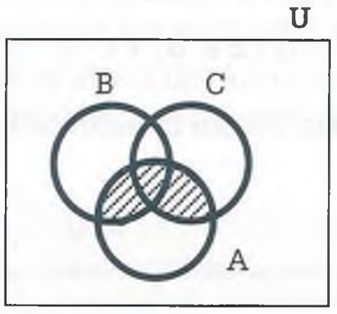
$A \cap (B \cup C)$



$A \cap B$



$A \cap C$

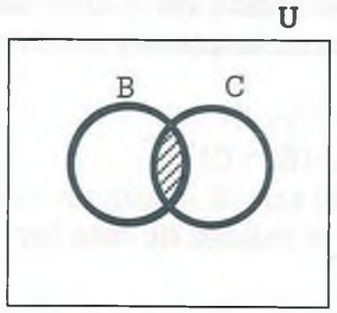


$(A \cap B) \cup (A \cap C)$

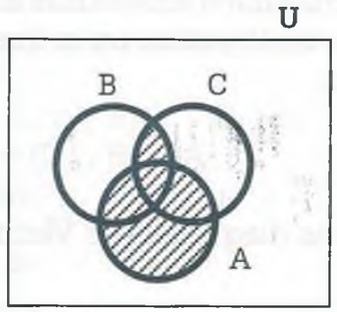
Por otro lado, la validez de la expresión

$$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$$

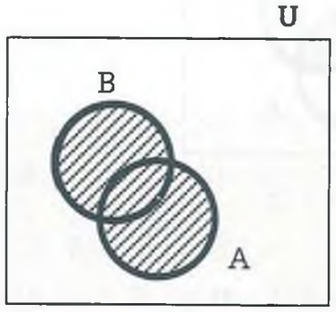
se muestra mediante los siguientes diagramas:



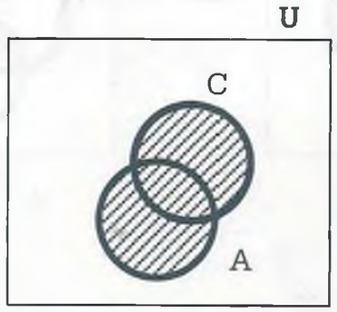
$B \cap C$



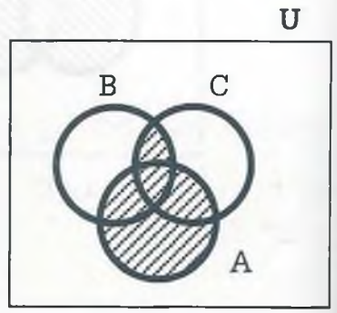
$A \cup (B \cap C)$



$A \cup B$



$A \cup C$



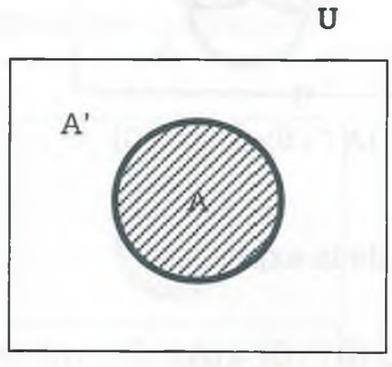
$(A \cup B) \cap (A \cup C)$

3.5.4 Complemento (A′)

El complemento de un conjunto A, que se denota como A′, es el conjunto que contiene a todos los elementos del conjunto universo que no pertenecen al conjunto A:

$$A' = \{x \mid x \in \mathbf{U};\ x \notin A\}$$

El siguiente diagrama de Venn ilustra la definición de A′:



Ejemplo 3.9. Sean los conjuntos

$$\mathbf{U} = \{x \mid x \in \mathbf{Z}\} \qquad A = \{1, 3, 5, 8\}$$

Entonces aplicando la definición de A′ se tiene que:

$$A' = \{x \mid x \in \mathbf{Z};\ x \notin \{1, 3, 5, 8\}\}$$
$$= \{x \mid x \in \mathbf{Z};\ x \neq 1;\ x \neq 3;\ x \neq 5;\ x \neq 8\}$$

Partiendo de las definiciones correspondientes, se puede mostrar la validez de las siguientes propiedades del complemento:

a) $(A')' = A$
b) $A \cup A' = \mathbf{U}$
c) $A \cap A' = \varnothing$
d) $\mathbf{U}' = \varnothing$
e) $\varnothing' = \mathbf{U}$

Posteriormente se verá cómo todas estas propiedades que son válidas en la teoría de conjuntos también lo son en lógica matemática y en álgebra booleana, en donde se considera el conjunto universo como 1 y el conjunto vacío como 0.

3.5.5 Ley de Morgan

El matemático inglés Augustus De Morgan demostró que:

1) La negación de la intersección de dos o más conjuntos es equivalente a la unión de los conjuntos negados separadamente.
2) La negación de la unión de dos o más conjuntos es igual a la intersección de los conjuntos negados por separado.

Ejemplo 3.10. Sean los conjuntos:

$\mathbf{U} = \{1, 2, 3, 4, 5, 6, 7, 8, 9, 10\}$
$A = \{1, 3, 6, 7, 9, 10\}$
$B = \{1, 2, 3, 7, 9, 10\}$

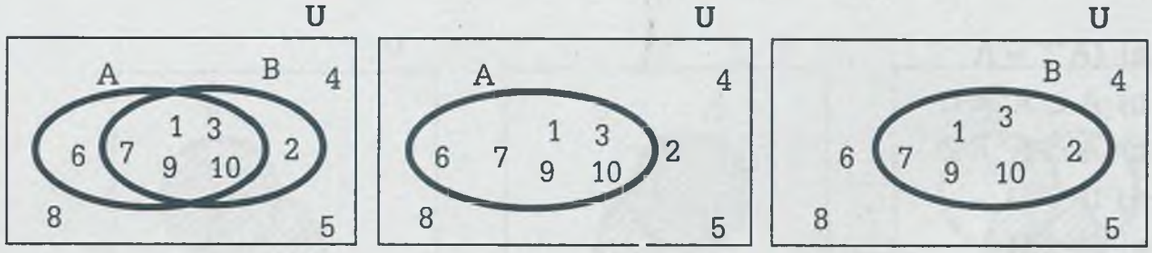
Aplicando las definiciones correspondientes se tiene que:

$A \cup B = \{1, 2, 3, 6, 7, 9, 10\}$
$(A \cup B)' = \{4, 5, 8\}$

Por otro lado también se tiene que:

$A' = \{2, 4, 5, 8\}$
$B' = \{4, 5, 6, 8\}$
$A' \cap B' = \{4, 5, 8\}$

Usando diagramas de Venn se tiene que:



$(A \cup B)' = \{4, 5, 8\}$ $A' = \{2, 4, 5, 8\}$ $B' = \{4, 5, 6, 8\}$

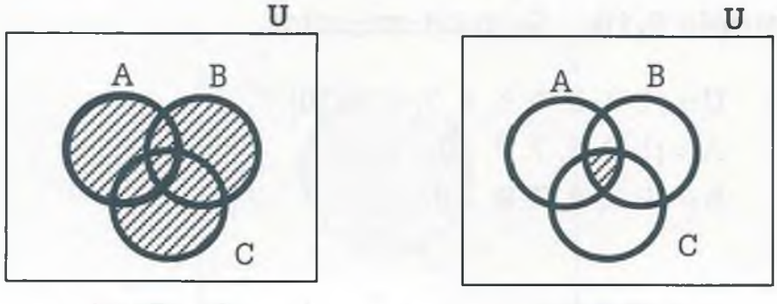
Por lo tanto se puede afirmar que

$$(A \cup B)' = (A' \cap B')$$

Hay que tomar en cuenta que la ley de Morgan también se puede aplicar a la intersección de forma que

$$(A \cap B)' = (A' \cup B')$$

Finalmente es importante mencionar que las operaciones de unión e intersección de conjuntos así como la ley de Morgan, se pueden extender a más de dos conjuntos. Por ejemplo, para los conjuntos arbitrarios A, B y C se tiene que:



$A \cup B \cup C$ $A \cap B \cap C$

También se cumple que:

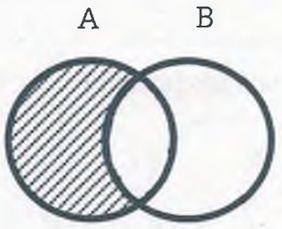
$$(A \cup B \cup C)' = A' \cap B' \cap C'$$
$$(A \cap B \cap C)' = A' \cup B' \cup C'$$

3.5.6 Diferencia (A-B)

La diferencia entre dos conjuntos arbitrarios A y B es el conjunto que contiene a todos los elementos del conjunto A que no se encuentran en B:

$$A - B = \{x \mid x \in A;\ x \notin B\}$$

El conjunto diferencia también se conoce como complemento de B con respecto a A. La siguiente figura muestra el diagrama de Venn de la definición:



A – B

Ejemplo 3.11. Sean los conjuntos:

$A = \{1, 2, 3, 4, 7, 9, 10\}$
$B = \{3, 4, 5, 6, 7, 8\}$

Entonces se tiene que:

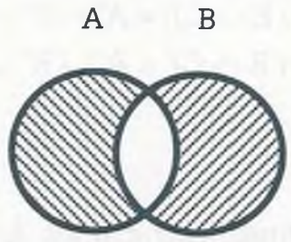
$A - B = \{1, 2, 9, 10\}$
$B - A = \{5, 6, 8\}$

3.5.7 Diferencia simétrica (A ⊕ B)

La diferencia simétrica entre los conjuntos A y B, que se denota como $A \oplus B$, es el conjunto que contiene a todos los elementos que se encuentran en el conjunto A pero no están en el conjunto B y también a los elementos del conjunto B que no están en A. Dicho de otra manera, el conjunto $A \oplus B$ contiene a todos los elementos que se encuentran en $A \cup B$ pero que no están en $A \cap B$:

$$A \oplus B = \{x \mid (x \in A \text{ y } x \notin B) \text{ o } (x \in B \text{ y } x \notin A)\}$$

El diagrama de Venn de la definición de diferencia simétrica es el siguiente:



$A \oplus B$

Ejemplo 3.12. Sean los conjuntos:

$A = \{1, 2, 3, 4, 7, 9, 10\}$
$B = \{3, 4, 5, 6, 7, 8\}$

Entonces aplicando las definiciones correspondientes se obtiene que:

$B - A = \{5, 6, 8\}$
$A - B = \{1, 2, 9, 10\}$
$A \oplus B = \{1, 2, 5, 6, 8, 9, 10\}$

Como se muestra en los siguientes ejemplos, para determinar de manera más clara los elementos de un conjunto es importante utilizar el diagrama de Venn cuando sea necesario.

Ejemplo 3.13. Sean los conjuntos:

$\mathbf{U} = \{x \mid x \in \mathbf{Z}\}$
$A = \{1, 2, 5, 7, 10, 12\}$
$B = \{x \mid x \in \mathbf{Z};\ 3 < x < 15;\ x \text{ es primo}\}$
$C = \{3, 5, 9, 10, 12, 13, 14\}$
$D = \{2, 4, 8, 10, 11\}$

Obtener:

a) $(A \cap B)'$
b) $(C \cup D') \oplus B'$
c) $C' - (D \oplus A)$
d) $[(A \cup B') - C] \oplus D'$

Solución

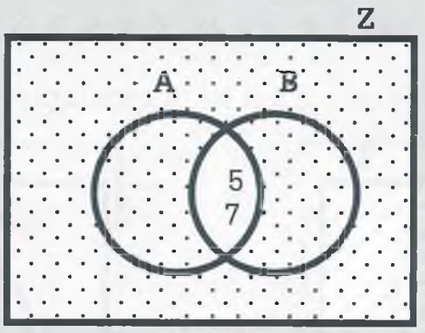
a) Tomando en cuenta la definición de número primo se tiene que $B = \{5, 7, 11, 13\}$ de forma que

$$A \cap B = \{5, 7\}$$

Aplicando la definición de complemento de un conjunto resulta que $(A \cap B)'$ son todos los números enteros a excepción de los que pertenecen a $A \cap B$:

$$(A \cap B)' = \{x \mid x \in \mathbf{Z};\ x \notin \{5, 7\}\}$$

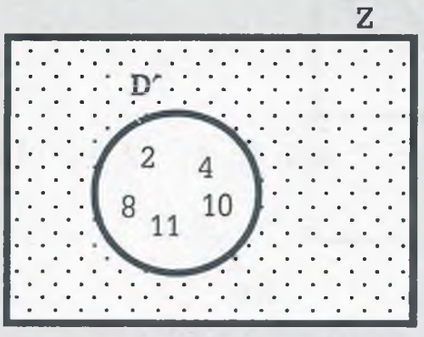
Esto se ilustra en el siguiente diagrama de Venn:



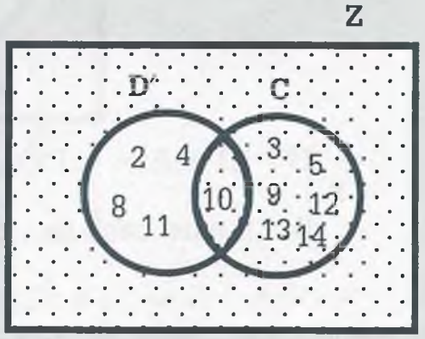
b) Aplicando las definiciones correspondientes se tiene que:

$D' = \{x \mid x \in \mathbf{Z};\ x \notin \{2, 4, 8, 10, 11\}\}$
$C \cup D' = \{x \mid x \in \mathbf{Z};\ x \notin \{2, 4, 8, 11\}\}$
$B' = \{x \mid x \in \mathbf{Z};\ x \notin \{5, 7, 11, 13\}\}$
$(C \cup D') \oplus B' = \{2, 4, 5, 7, 8, 13\}$

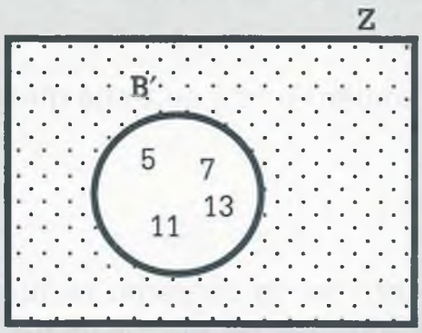
El diagrama de Venn de cada operación es el siguiente:



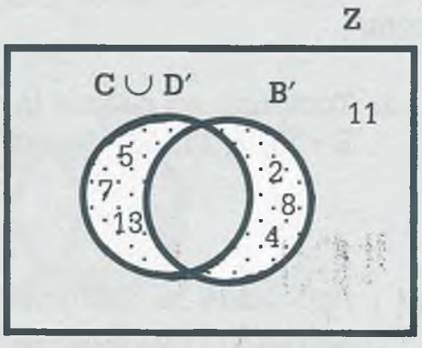
$D' = \{x \mid x \in \mathbf{Z};\ x \notin \{2, 4, 8, 10, 11\}\}$



$C \cup D' = \{x \mid x \in \mathbf{Z};\ x \notin \{2, 4, 8, 11\}\}$



$B' = \{ x \mid x \in \mathbf{Z};\ x \notin \{5, 7, 11, 13\}\}$



$(C \cup D') \oplus B' = \{2, 4, 5, 7, 8, 13\}$

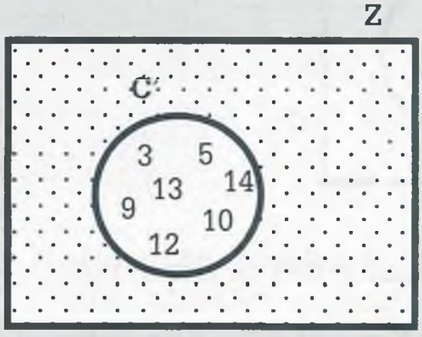
c) A partir de las definiciones correspondientes resulta que:

$\mathbf{C'} = \{x \mid x \in \mathbf{Z};\ x \notin \{3, 5, 9, 10, 12, 13, 14\}\}$

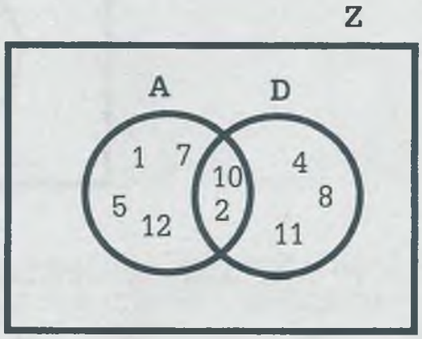
$D \oplus A = \{1, 4, 5, 7, 8, 11, 12\}$

$C' - (D \oplus A) = \{x \mid x \in \mathbf{Z};\ x \notin \{1, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14\}\}$

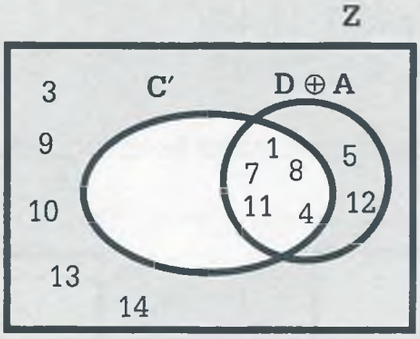
El diagrama de Venn de cada paso es el siguiente:



$C' = \{x \mid x \in \mathbf{Z};\ x \notin \{3, 5, 9, 10, 12, 13, 14\}\}$



$D \oplus A = \{1, 4, 5, 7, 8, 11, 12\}$



$C' - (D \oplus A) = \{x \mid x \in \mathbf{Z};\ x \notin \{1, 3, 4, 5, 7, 8, 9, 10, 11, 12, 13, 14\}\}$

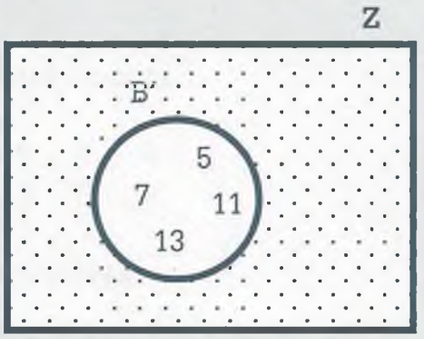
d) En este caso se tiene que:

$A \cup B' = \{x \mid x \in Z;\ x \notin \{11, 13\}\}$

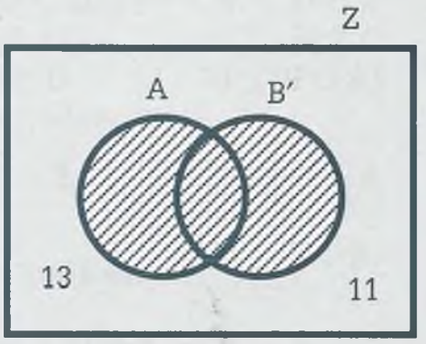
$(A \cup B') - C = \{x \mid x \in Z;\ x \notin \{3, 5, 9, 10, 11, 12, 13, 14\}\}$

$[(A \cup B') - C] \oplus D' = \{2, 3, 4, 5, 8, 9, 12, 13, 14\}$

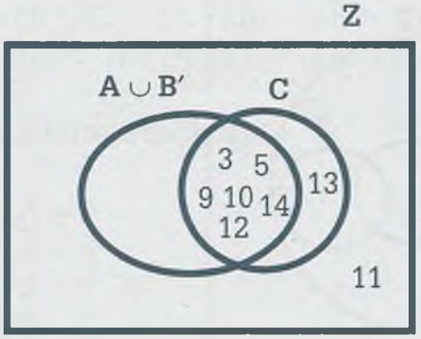
Los diagramas de Venn correspondientes son los siguientes:



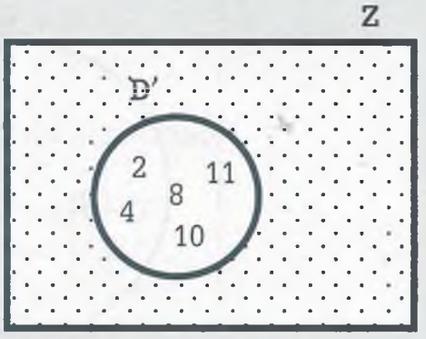
$B' = \{x \mid z \in \mathbf{Z};\ x \notin \{5, 7, 11, 13\}\}$



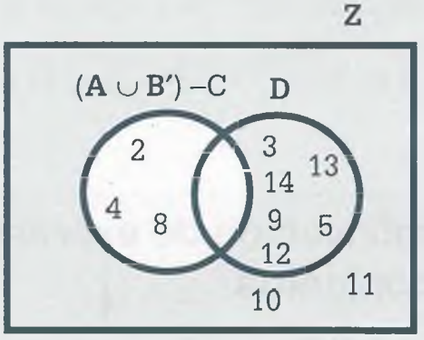
$A \cup B' = \{x \mid x \in \mathbf{Z};\ x \notin \{11, 13\}\}$



$(A \cup B') - C = \{x \mid x \in \mathbf{Z};\ x \notin \{3, 5, 9, 10, 11, 12, 13, 14\}\}$



$D' = \{x \mid x \in \mathbf{Z};\ x \notin \{2, 4, 8, 10, 11\}\}$



$[(A \cup B') - C] \oplus D' = \{2, 3, 4, 5, 8, 9, 12, 13, 14\}$

Ejemplo 3.14. El diagrama de Venn que cumple con las condiciones:

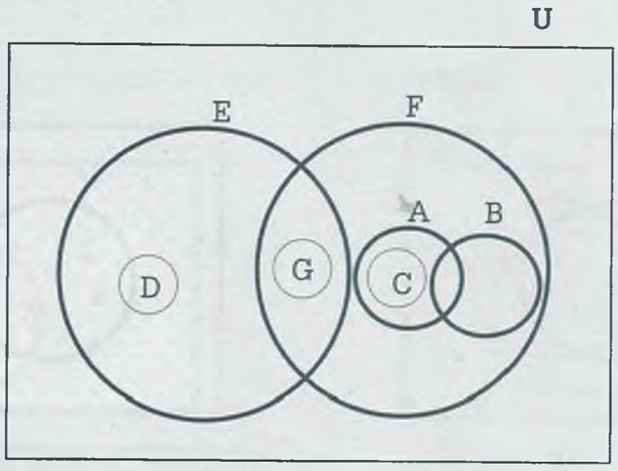
$C \subseteq (A - B)$ $\quad D \subseteq (E - F)$

$(A \cup B) \subseteq (F - E)$ $\quad G \subseteq (E \cap F)$

$A \cap B \neq \varnothing$ $\quad E \cap F \neq \varnothing$

$(E \cup F) \subseteq \mathbf{U}$

es el siguiente:



3.6 Simplificación de expresiones usando leyes de conjuntos

A partir de las definiciones planteadas es posible establecer varias leyes de conjuntos que son útiles para simplificar u obtener expresiones equivalentes en donde intervienen operaciones propias de conjuntos. En la tabla 3.1 se presentan las leyes de conjuntos más importantes.

Tabla 3.1 Leyes de conjuntos

1.- Doble negación **a)** $A'' = A$	**6.- Ley de Morgan** **a)** $(A \cup B \cup C)' = A' \cap B' \cap C'$ **b)** $(A \cap B \cap C)' = A' \cup B' \cup C'$
2.- Ley conmutativa **a)** $A \cup B = B \cup A$ **b)** $A \cap B = B \cap A$	**7.- Equivalencia** **a)** $A \cup A' \cap B = A \cup B$
3.- Ley asociativa **a)** $A \cup (B \cup C) = (A \cup B) \cup C$ **b)** $A \cap (B \cap C) = (A \cap B) \cap C$	**8.- Contradicción** **a)** $A \cap A' = \varnothing$
4.- Ley distributiva **a)** $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ **b)** $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	**9.- Propiedades del complemento** **a)** $A \cup A' = \mathbf{U}$ **b)** $\mathbf{U}' = \varnothing$ **c)** $\varnothing' = \mathbf{U}$
5.- Ley de idempotencia **a)** $A \cup A = A$ **b)** $A \cap A = A$ **c)** $\mathbf{U} \cup \mathbf{U} = \mathbf{U}$ **d)** $\mathbf{U} \cap \mathbf{U} = \mathbf{U}$ **e)** $\varnothing \cup \varnothing = \varnothing$ **f)** $\varnothing \cap \varnothing = \varnothing$	**10.- Ley de identidad** **a)** $A \cup \mathbf{U} = \mathbf{U}$ **b)** $A \cap \mathbf{U} = A$ **c)** $A \cup \varnothing = A$ **d)** $A \cap \varnothing = \varnothing$ **e)** $A \cup A \cap B = A \cap (\mathbf{U} \cup B) = A$

Ejemplo 3.15. Usando las leyes de conjuntos, demostrar que

$$A' \cap B' \cap C \cup A' \cap B \cap C \cup A \cap B' \cap C \cup A \cap B \cap C \cup A \cap B \cap C' = A \cup B \cap C$$

Solución

$$\mathbf{A'} \cap B' \cap \mathbf{C} \cup \mathbf{A'} \cap B \cap \mathbf{C} \cup \mathbf{A} \cap B' \cap \mathbf{C} \cup \mathbf{A} \cap B \cap \mathbf{C} \cup A \cap B \cap C' = C \cup A \cap B$$

$(A' \cap C) \cap (\mathbf{B'} \cup \mathbf{B}) \cup (A \cap C) \cap (\mathbf{B'} \cup \mathbf{B}) \cup A \cap B \cap C' = \mathbf{C} \cup A \cap B$
Ley distributiva 4a.

$(A' \cap C) \cap \mathbf{U} \cup (A \cap C) \cap \mathbf{U} \cup A \cap B \cap C' = C \cup A \cap B$
Propiedades del complemento 9a.

$A' \cap \mathbf{C} \cup A \cap \mathbf{C} \cup A \cap B \cap C' = C \cup A \cap B$
Ley de identidad 10b.

$C \cap (\mathbf{A'} \cup \mathbf{A}) \cup A \cap B \cap C' = C \cup A \cap B$
Ley distributiva 4a.

$C \cap \mathbf{U} \cup A \cap B \cap C' = C \cup A \cap B$
Propiedades del complemento 9a.

$\mathbf{C} \cup A \cap B \cap C' = C \cup A \cap B$
Ley de identidad 10b.

$\mathbf{C} \cup \mathbf{C'} \cap \mathbf{A} \cap \mathbf{B} = C \cup A \cap B$
Ley conmutativa 2b.

$C \cup A \cap B = C \cup A \cap B$
Equivalencia 7a.

Como se ve, en la demostración del ejemplo anterior se dejó fijo el lado derecho de la expresión y se simplificó el lado izquierdo hasta el punto en que se obtuvo la expresión de la derecha. Asimismo, debajo de cada planteamiento está la regla que se aplicó.

Por ejemplo: en las primeras tres líneas de la simplificación se aplica la "ley distributiva 4a" que consiste en factorizar algo que es común y dejar dentro del paréntesis lo no común, con la finalidad de que la información que quede dentro del paréntesis se pueda simplificar aplicando una nueva regla de conjuntos. Posteriormente se aplica la "propiedad del complemento 9a" que establece que $A \cup A' = \mathbf{U}$ y después la "ley de identidad 10b" $A \cap \mathbf{U} = A$. A continuación se muestra este procedimiento.

$$\underline{A'} \cap B' \cap \underline{C} \cup \underline{A'} \cap B \cap \underline{C} = (A' \cap C) \cap (B' \cup B)$$

Lo común es lo que está subrayado y aplicando la ley distributiva 4a que establece que:

$$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$$

Se obtiene que:

$$(A' \cap C) \cap (B' \cup B) = (A' \cap C) \cap \mathbf{U}$$

Considerando que la propiedad del complemento 9a establece que $A \cup A' = \mathbf{U}$, entonces se puede decir que $B' \cup B = \mathbf{U}$ de forma que:

$$(A' \cap C) \cap \mathbf{U} = A' \cap C$$

Ya que la "ley de identidad 10b" establece que $A \cap \mathbf{U} = A$, de aquí se desprende que $(A' \cap C) \cap \mathbf{U} = A' \cap C$ y por tanto:

$$\underline{A} \cap B' \cap \underline{C} \cup \underline{A} \cap B \cap \underline{C} = (A \cap C) \cap (B' \cup B)$$

Lo común está subrayado y aplicando la ley distributiva 4a resulta que:

$$(A \cap C) \cap (B' \cup B) = (A \cap C) \cap \mathbf{U}$$

Aplicando la propiedad del complemento 9a se obtiene:

$$(A \cap C) \cap \mathbf{U} = A \cap C$$

Aplicando la ley de identidad 10b.

$$A \cap C = A \cap C$$

En las líneas cuatro, cinco y seis se aplican nuevamente la ley distributiva 4a, la propiedad del complemento 9a y la ley de identidad 10b respectivamente, como se muestra a continuación:

$$A' \cap \underline{C} \cup A \cap \underline{C} = C \cap (A' \cup A)$$

Según ley distributiva 4a.

$$C \cap (A' \cup A) = C \cap \mathbf{U}$$

Después de aplicar la propiedad del complemento 9a.

$$C \cap U = C$$

Después de aplicar la ley de identidad 10b.

En las líneas séptima y octava se aplicaron la ley conmutativa 2b y finalmente la ley de equivalencia 7a, como se muestra a continuación:

$$C \cup A \cap B \cap \underline{C'} = C \cup C' \cap A \cap B$$

Después de aplicar la ley conmutativa 2b. (El conjunto que cambia de posición está subrayado.)

$$C \cup C' \cap A \cap B = C \cup A \cap B$$

Después de aplicar la ley de equivalencia que establece que $A \cup A' \cap B = A \cup B$. (Para poder aplicar la regla se considera que $A = C$; $A' = C'$ y $B = A \cap B$.)

El procedimiento generalizado es factorizar información común para aplicarle una nueva regla a la información no común que se encuentra dentro del paréntesis y de esa manera ir eliminando algunos conjuntos.

Ejemplo 3.16. Demostrar que

$$A' \cap B \cup (A \cap B \cap C)' \cup C \cap (B' \cup A) = \mathbf{U}$$

Solución

$A' \cap B \cup (\mathbf{A \cap B \cap C})' \cup C \cap (B' \cup A) = \mathbf{U}$
$A' \cap B \cup \mathbf{A' \cup B' \cup C'} \cup C \cap (B' \cup A) = \mathbf{U}$
Ley de Morgan 6b.

$\mathbf{A' \cap B \cup A' \cup B' \cup C' \cup C \cap B' \cup C \cap A = U}$
Ley distributiva 4a.

$A' \cap (B \cup \mathbf{U}) \cup B' \cap (\mathbf{U} \cup C) \cup C' \cup C \cap A = \mathbf{U}$
Ley distributiva 4a.

$A' \cap \mathbf{U} \cup B' \cap \mathbf{U} \cup C' \cup C \cap A = \mathbf{U}$
Ley de identidad 10a.

$A' \cup B' \cup \mathbf{C' \cup C} \cap A = \mathbf{U}$
Ley de identidad 10b.

$A' \cup B' \cup C' \cup A = \mathbf{U}$
Equivalencia 7a.

$\mathbf{A' \cup A} \cup B' \cup C' = \mathbf{U}$
Ley conmutativa 2a.

$\mathbf{U} \cup B' \cup C' = \mathbf{U}$
Propiedad del complemento 9a.

$\mathbf{U} \cup C' = \mathbf{U}$
Ley de identidad 10a.

$\mathbf{U} = \mathbf{U}$
Ley de identidad 10a.

3.7 Relación entre teoría de conjuntos, lógica matemática y álgebra booleana

La lógica matemática y el álgebra booleana son herramientas fundamentales de la computación que se apoyan en las leyes de la teoría de conjuntos para explicar teoremas matemáticos o bien para simplificar expresiones booleanas. En la tabla 3.2 se presenta una comparación entre las leyes de la teoría de conjuntos, algunas equivalencias lógicas usadas en lógica matemática para la demostración de teoremas y algunas leyes del álgebra booleana que se utilizan en la simplificación de funciones booleanas.

Tabla 3.2 Equivalencias entre teoría de conjuntos, lógica matemática y álgebra booleana

Propiedad	Teoría de conjuntos	Lógica matemática	Álgebra booleana
Equivalencia	$A = B$	$p \Leftrightarrow q;\ p \equiv q$	$A = B$
Unión	$A \cup B$	$p \vee q$	$A + B$
Intersección	$A \cap B$	$p \wedge q$	AB
Complementación	A'	p'	A'
Doble negación	$A'' = A$	$p'' \equiv p$	$A'' = A$
Diferencia	$A - B$	$p \wedge q'$	AB'
Leyes de Morgan	$(A \cup B \cup C)' = A' \cap B' \cap C'$ $(A \cap B \cap C)' = A' \cup B' \cup C'$	$(p \vee q \vee r)' \equiv p' \wedge q' \wedge r'$ $(p \wedge q \wedge r)' \equiv p \vee q \vee r'$	$(A + B + C)' = A'\,B'\,C'$ $(ABC)' = A' + B' + C'$
Ley conmutativa	$A \cup B = B \cup A$ $A \cap B = B \cap A$	$p \vee q \equiv q \vee p$ $p \wedge q \equiv q \wedge p$	$A + B = B + A$ $AB = BA$
Ley asociativa	$A \cup (B \cup C) = (A \cup B) \cup C$ $A \cap (B \cap C) = (A \cap B) \cap C$	$p \vee (q \vee r) \equiv (p \vee q) \vee r$ $p \wedge (q \wedge r) \equiv (p \wedge q) \wedge r$	$A + (B + C) = (A + B) + C$ $A(BC) = (AB)C$
Ley distributiva	$A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$ $A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$	$p \wedge (q \vee r) \equiv (p \wedge q) \vee (p \wedge r)$ $p \vee (q \wedge r) \equiv (p \vee q) \wedge (p \vee r)$	$A(B + C) = AB + AC$ $A + (BC) = (A + B)(A + C)$
Ley de idempotencia	$A \cup A = A$ $A \cap A = A$ $\mathbf{U} \cup \mathbf{U} = \mathbf{U}$ $\mathbf{U} \cap \mathbf{U} = \mathbf{U}$ $\varnothing \cup \varnothing = \varnothing$ $\varnothing \cap \varnothing = \varnothing$	$p \vee p \equiv p$ $p \wedge p \equiv p$ $1 \vee 1 \equiv 1$ $1 \wedge 1 \equiv 1$ $0 \vee 0 \equiv 0$ $0 \wedge 0 \equiv 0$	$A + A = A$ $AA = A$ $1 + 1 = 1$ $1(1) = 1$ $0 + 0 = 0$ $0(0) = 0$
Equivalencia	$A \cup A' \cap B = A \cup B$	$p \vee p' \wedge q \equiv p \vee q$	$A + A'\,B = A + B$
Contradicción	$A \cap A' = \varnothing$	$p \wedge p' \equiv 0$	$AA' = 0$
Propiedades del complemento	$A \cup A' = \mathbf{U}$ $\mathbf{U}' = \varnothing$ $\varnothing' = U$	$p \vee p' \equiv 1$ $1' \equiv 0$ $0' \equiv 1$	$A + A' = 1$ $1' = 0$ $0' = 1$
Ley de identidad	$A \cup \mathbf{U} = \mathbf{U}$ $A \cap \mathbf{U} = A$ $A \cup \varnothing = A$ $A \cap \varnothing = \varnothing$ $A \cup A \cap B = A \cap (\mathbf{U} \cup B) = A$	$p \vee 1 \equiv 1$ $p \wedge 1 \equiv p$ $p \vee 0 \equiv p$ $p \wedge 0 \equiv 0$ $p \vee p \wedge q \equiv p \wedge (1 \vee q) \equiv p$	$A + 1 = 1$ $A(1) = A$ $A + 0 = A$ $A(0) = 0$ $A + AB = A(1 + B) = A$

En la tabla 3.2 hay que observar dos cosas: la primera es que las leyes de la lógica matemática y el álgebra booleana son formalmente las mismas que las de la teoría de conjuntos, y la segunda es que las operaciones equivalentes se denotan de manera diferente en cada una.

En el caso de la unión, en la teoría de conjuntos se usa el símbolo $\cup$ mientras que la operación equivalente se denota como $\vee$ en lógica matemática y como + en álgebra booleana. En relación con la intersección, en la teoría de conjuntos se usa el símbolo $\cap$ mientras que para la operación equivalente en lógica matemática se usa el símbolo $\wedge$ y en álgebra booleana simplemente se indica como una multiplicación.

Otra diferencia de notación que hay que tener presente es que en teoría de conjuntos el conjunto universo se denota como **U** y el conjunto vacío como $\varnothing$, mientras que en lógica matemática y álgebra booleana los conceptos equivalentes se denotan como 1 y 0 respectivamente.

Por último hay que agregar que el contenido de la tabla 3.2 se usará ampliamente en la exposición de la lógica matemática y el álgebra booleana.

3.8 Conjuntos finitos

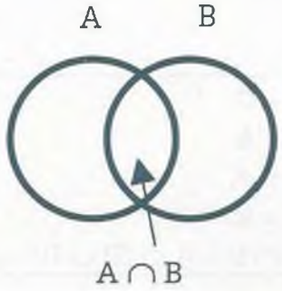
En algunos de los ejemplos anteriores se usaron conjuntos infinitos, como el conjunto de los enteros no negativos ($\mathbf{Z}^+$) y el conjunto de los números reales (**R**), o bien conjuntos que resultaron infinitos porque no es posible saber el número exacto de sus elementos, como $A = \{x \mid x \in \mathbf{Z};\ x > 9\}$. En este tipo de conjuntos se conocen las características de los elementos, pero no se sabe cuántos de ellos pertenecen al conjunto. Sin embargo algunas veces se desea saber cuántos elementos pertenecen a un conjunto, y no necesariamente cómo son éstos. En este caso se utilizan conjuntos finitos o bien conjuntos en donde se sabe con exactitud el número de elementos contenidos.

Sean A y B dos conjuntos finitos, entonces:

$$|A \cup B| = |A| + |B| - |A \cap B|$$

donde $|A|$ es la cardinalidad de A y $|B|$ es la cardinalidad de B.

Utilizando diagramas de Venn la expresión de $|A \cup B|$ corresponde a la suma del "área" del conjunto A más el "área" del conjunto B, pero como el "área" de $A \cap B$ se sumó dos veces es necesario restarla una vez.

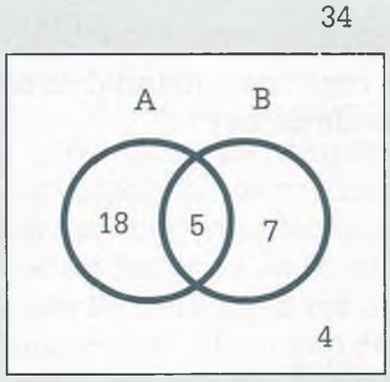


Ejemplo 3.17. De 34 programas revisados en programación "C++", 23 marcaron error en la compilación, 12 tuvieron fallas en lógica y 5 en lógica y compilación. ¿Cuántos programas tuvieron al menos un tipo de error?

Aquí se tiene que

$$|A \cup B| = |A| + |B| - |A \cap B| = 23 + 12 - 5 = 30$$

En este caso es más claro si el problema se resuelve usando un diagrama de Venn, poniendo la información de dentro hacia afuera. Esto significa poner primero $|A \cap B| = 5$ en el diagrama, después $|A| = 18 + 5$ y $|B| = 7 + 5$ y así sucesivamente hasta $|U| = 18 + 5 + 7 + 4 = 34$, como se muestra en la siguiente figura:

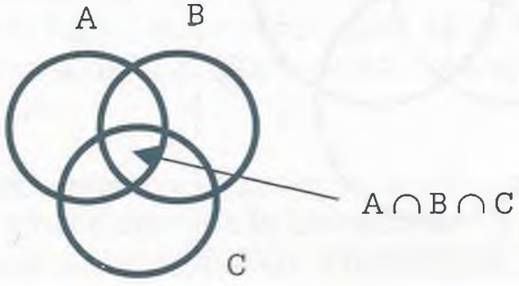


A partir del diagrama se puede observar que 4 de los programas no tuvieron error.

En el caso de tres conjuntos finitos A, B, y C, la expresión

$$|A \cup B \cup C| = |A| + |B| + |C| - |A \cap B| - |B \cap C| - |A \cap C| + |A \cap B \cap C|$$

también se puede determinar sumando "áreas" como se indica a continuación:



Ejemplo 3.18. En la biblioteca existen 103 libros de ciencias de la computación que tratan en cierta medida los siguientes temas:

- Compiladores.
- Estructura de datos.
- Redes.

Del total, 50 libros tienen información sobre compiladores, 54 sobre estructuras de datos, 51 sobre redes, 30 sobre compiladores y estructuras de datos, 32 sobre compiladores y redes, 35 sobre estructuras de datos y redes, 19 sobre los tres temas.

a) ¿Cuántos libros contienen material exactamente sobre uno de los tres temas?

b) ¿Cuántos no tienen material de redes?

c) ¿Cuántos no tienen material sobre ninguno de los temas?

d) ¿Cuántos libros contienen material de compiladores y redes pero no de estructura de datos?

Solución

Considérese que:

A = Compiladores.

B = Estructuras de datos.

C = Redes.

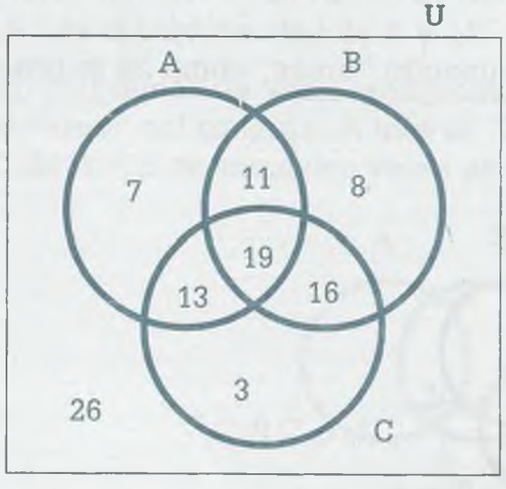
Por tanto:

$|A| = 50$ $\quad |A \cap B| = 30$ $\quad |A \cap B \cap C| = 19$

$|B| = 54$ $\quad |A \cap C| = 32$

$|C| = 51$ $\quad |B \cap C| = 35$

Colocando la información en el diagrama de Venn se tiene que



Del diagrama se puede leer que:

a) El número de libros que contiene material exclusivamente de uno de los temas es 7 + 8 + 3 = 18.

b) Los libros que no tienen material de redes son 26 + 7 + 11 + 8 = 52.

c) Los libros que no tienen material de ninguno de los tres temas son 26.

d) Los libros que tienen información de redes y compiladores pero no de estructura de datos son 13.

Generalización de conjuntos finitos

Cuando se opera con más de tres conjuntos es complicado determinar la fórmula que representa la unión entre todos los conjuntos, sumando y restando las diferentes secciones involucradas de un diagrama de Venn, porque incluso es difícil determinar las diferentes secciones en el diagrama. En este caso se usa el principio de *inclusión exclusión* que establece que se deben sumar las áreas que involucran un número *non* de conjuntos y se restan las que relacionan un número *par*. El número de elementos que se suman o restan en la fórmula está dado por $(2^n - 1)$, donde n es el número de conjuntos que participan. Por ejemplo, para cuatro conjuntos se tiene que

$$|A \cup B \cup C \cup D| = |A| + |B| + |C| + |D| - |A \cap B| - |A \cap C| - |A \cap D| - |B \cap C| - |B \cap D| - |C \cap D| + |A \cap B \cap C| + |A \cap B \cap D| + |A \cap C \cap D| + |B \cap C \cap D| - |A \cap B \cap C \cap D|$$

Hay que observar que las cantidades que se suman o restan son $(2^4 - 1) = 15$, y que las partes que relacionan números nones de conjuntos se suman y las que involucran números pares se restan.

3.9 Aplicación de la teoría de conjuntos

Ya se vio la estrecha relación que existe entre la teoría de conjuntos, el algebra booleana y la lógica matemática, pero además de esto prácticamente todos los campos de la computación se respaldan en la teoría de conjuntos. Por ejemplo:

- Una relación es un conjunto y en bases de datos es posible llevar a cabo operaciones entre relaciones, de la misma manera en que se hacen en teoría de conjuntos, de forma que los conceptos de unión, intersección,

complementación, así como otras reglas lógicas que resultan de mezcla estas tres operaciones básicas de conjuntos dan origen a lo que se co noce como álgebra relacional, misma que a su vez proporciona los ele mentos necesarios con los que se manejan las bases de datos relacionales y que permiten obtener la información en forma organiza da y concreta.

- Los lenguajes de programación se definen como un conjunto de conjun tos, y dentro de ellos se puede mencionar el conjunto de símbolos (a alfabeto) con los cuales se forman las palabras de un lenguaje, el con junto de símbolos no terminales que permiten multiplicar y mezcla organizadamente los símbolos del alfabeto, el conjunto de composicio nes o reglas que se deben usar para la estructuración de las palabras válidas en el lenguaje y el conjunto de símbolos terminales que marcan el límite de esas palabras válidas de un lenguaje. Por lo tanto, si un lenguaje es un conjunto de conjuntos, es claro que obedece también a las leyes y reglas de la teoría de conjuntos.

- Las redes de teléfonos, eléctricas, carreteras, de agua potable o de computadoras son relaciones y por lo tanto son conjuntos a los cuales se les pueden aplicar también las operaciones unión, intersección, com plementación, composición y ley de Morgan, de la misma manera que se hace en teoría de conjuntos, por lo tanto también es una aplicación práctica de la teoría de conjuntos. Esta representación gráfica de los conjuntos se conoce en computación como teoría de grafos y será obje to de estudio posteriormente en este libro.

Por lo tanto se puede concluir que para la computación, la teoría de con juntos es fundamental.

3.10 Resumen

Un conjunto es una colección bien definida de objetos llamados elemento o miembros del conjunto. Los conjuntos se indican por medio de una letra mayúscula y los elementos del conjunto se indican por medio de letras mi núsculas, números, símbolos o bien combinaciones de éstos, y los eleme tos se colocan entre llaves y se separan por comas. Algunas veces no es posible hacer la lista de los elementos de un conjunto porque se tra ta de un conjunto infinito, y en lugar de esto el conjunto se indica por medio de la notación abstracta que tiene la siguiente forma

$$A = \{x \mid P(x)\}$$

y que se lee "A es el conjunto de las x tal que x cumple con la condición condiciones P(x)", de forma que cada una de las condiciones planteadas en P(x) se debe cumplir para que un elemento x pertenezca al conjunto A.

Si un elemento x pertenece o no a un conjunto B se indica de la siguiente manera:

$x \in B$ x es elemento del conjunto B.

$x \notin B$ x no es elemento del conjunto B.

Si todos los elementos de A también son elementos de B, se dice que A es subconjunto de B o que A está contenido en B y esto se indica como:

$$A \subseteq B$$

Si todos los elementos de A no están contenidos en B entonces se dice que A no es subconjunto de B, y esto se indica como:

$$A \not\subset B$$

El conjunto vacío ($\varnothing$) es un subconjunto de todos los conjuntos, y todos los conjuntos son subconjuntos del conjunto universo (**U**):

$$\varnothing \subseteq A \qquad A \subseteq \mathbf{U}$$
$$\varnothing \subseteq \mathbf{U} \qquad \varnothing \subseteq \mathbf{U}$$
$$\varnothing \subseteq \varnothing \qquad \mathbf{U} \subseteq \mathbf{U}$$

Si A es un conjunto, entonces al conjunto de todos los subconjuntos de A se le llama conjunto potencia de A y se indica como P(A). El número de subconjuntos del conjunto A está dado por:

$$|P(A)| = 2^n$$

Los diagramas de Venn son representaciones gráficas para mostrar la relación entre los elementos de los conjuntos, y en estos diagramas cada conjunto se representa por medio de un círculo, óvalo o rectángulo. Es recomendable utilizar los diagramas de Venn siempre que sea posible, ya que permiten ilustrar y visualizar con mayor claridad los elementos que pertenecen a un conjunto.

Las operaciones que se pueden realizar entre conjuntos son unión, intersección y complementación. De estas operaciones básicas se derivan propiedades y leyes como la ley conmutativa, asociativa, distributiva, de idempotencia, de identidad y de Morgan. Estas propiedades y leyes son aplicables también en lógica matemática y en álgebra booleana, así como en otras áreas de la computación haciendo únicamente pequeñas modificaciones en su presentación.

La cardinalidad de un conjunto es el número de elementos que pertenecen a ese conjunto y se indica colocando el nombre del conjunto entre dos

pequeñas líneas rectas, por ejemplo $|C| = 8$ significa que los elementos del conjunto C son 8. Por otro lado, existen conjuntos finitos y conjuntos infinitos: los finitos son aquellos en donde es posible determinar con exactitud el número de elementos que pertenecen a él, mientras que en los conjuntos infinitos no es posible saber cuántos elementos forman parte de él.

Los conceptos de la teoría de conjuntos son el fundamento de áreas de las matemáticas como la lógica matemática y la probabilidad, pero sobre todo son básicos en computación ya que son esenciales en álgebra booleana, relaciones, funciones, árboles, redes, lenguajes y autómatas.

3.11 Problemas

3.1. ¿Cuáles son los elementos de los siguientes conjuntos?

a) $A = \{x \mid x$ es una letra de la palabra hola$\}$
b) $B = \{x \mid x$ es un dígito del número 103836$\}$
c) $C = \{x \mid x \in \mathbf{Z}^+; x - 4 \leq 3\}$
d) $D = \{x \mid x$ es un dígito válido en el sistema hexadecimal$\}$
e) $E = \{x \mid x \in \mathbf{Z}; x$ es divisible entre 3; $-4 < x < 17\}$

3.2. ¿Cuáles son los elementos de los siguientes conjuntos?

a) $A = \{x \mid x \in \mathbf{Z}^+; x$ es primo; x es par$\}$
b) $B = \{x \mid x \in \mathbf{Z}^+; 5 > x - 2\}$
c) $C = \{x \mid$ es una letra de la palabra "América" diferente de vocal$\}$
d) $D = \{x \mid x \in \mathbf{Z}^+; x$ es múltiplo de 7; $x < 100$; x es impar$\}$
e) $E = \{x \mid x \in \mathbf{Z}; 0 \leq x^3 < 100\}$

3.3. Escriba el conjunto en la forma $\{x \mid P(x)\}$, donde $P(x)$ es una o varias propiedades comunes de los elementos del conjunto.

a) $A = \{$suma, resta, multiplicación, división$\}$
b) $B = \{3, 6, 9, 12, 15, 18\}$
c) $C = \{1, 2, 3, 5, 7, 11, 13, 17\}$
d) $D = \{$américa, africa, europa, asia, oceanía$\}$
e) $E = \{1, 2, 4, 8, 16, 32, 64\}$

3.4. Escribir el conjunto en la forma $\{x \mid P(x)\}$, donde $P(x)$ es una o varias propiedades comunes de los elementos del conjunto.

a) $A = \{do, re, mi, fa, sol, la, si\}$

b) $B = \{1, 4, 9, 16, 25, 36\}$

c) $C = \{1, 4, 7, 10, 13, 16, 19\}$

d) $D = \{c, o, n, j, u, t\}$

e) $E = \{6, 12, 18, 24, 30\}$

3.5. ¿Cuántos elementos pertenecen al conjunto potencia P(A)? y ¿cuáles son sus elementos? si $A = \{manzana, pera, fresa, sandía\}$.

3.6. ¿Cuántos elementos pertenecen al conjunto potencia P(A)? y ¿cuáles son sus elementos? si $A = \{x \mid x \text{ es una letra vocal del alfabeto}\}$.

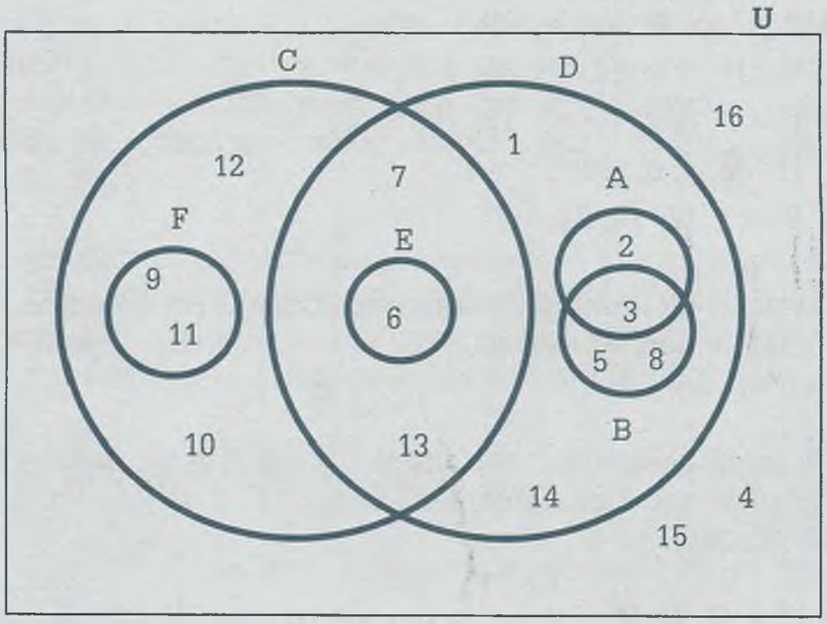
3.7. Sean A, B, C, D, E y F conjuntos no vacíos. Para cada inciso, hacer un diagrama de Venn que cumpla con las condiciones que se plantean:

a)	$A \subseteq (C \cap D)$	$E \subseteq D$	$C \cap D \neq \emptyset$
	$B \subseteq E$	$E \not\subset (C \cap D)$	
b)	$F \subseteq A$	$F \not\subset (A \cap B)$	$C \cap D \neq \emptyset$
	$E \subseteq (D - C)$	$(C \cup D) \subseteq (A \cap B)$	$A \cap B \neq \emptyset$

3.8. Sean A, B, C, D, E y F conjuntos no vacíos. Para cada inciso, hacer un diagrama de Venn que cumpla con las condiciones que se plantean:

a)	$A \not\subset B$	$C \subseteq (A - B)$	$A - B \neq \emptyset$
	$D \subseteq C$	$F \not\subset (A \cup B)$	$F \subseteq E$
	$(A \subseteq B) \subseteq E$		
b)	$A \cap B \neq \emptyset$	$E \subseteq C$	$(F \cup G) \subseteq D$
	$D \subseteq (A \cap B)$	$C \subseteq (A - B)$	$A - B \neq \emptyset$

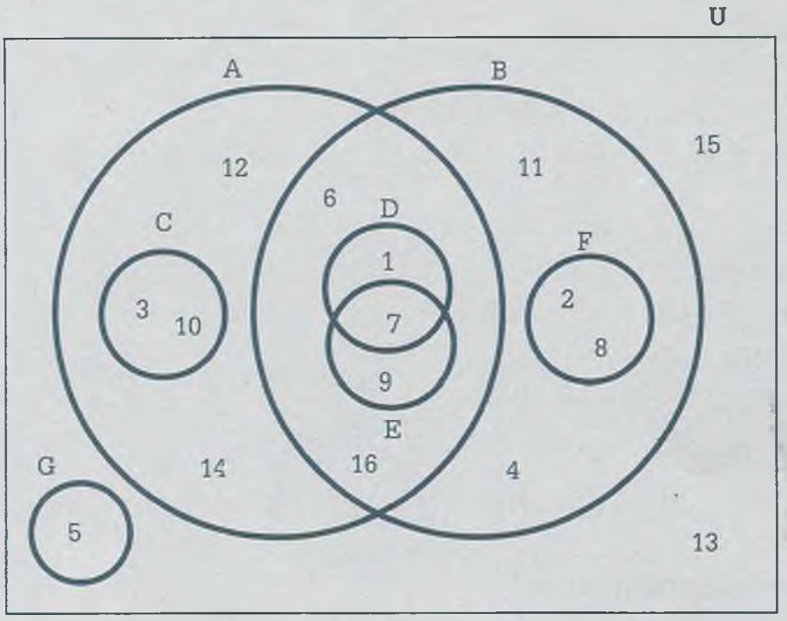
3.9. Considérese el siguiente diagrama de Venn.



Poner en el paréntesis de cada uno de los incisos una "V" si la aseveración es verdadera o bien una "F" si es falsa.

a) $F \subseteq (C - D)$ ( )
b) $E \subseteq D$ ( )
c) $E \subseteq (C \cap D)$ ( )
d) $(A \cap B) = \varnothing$ ( )
e) $(D - C) \subseteq (B - A)$ ( )
f) $(C \cap D) \subseteq \mathbf{U}$ ( )
g) $D = \{1, 2, 3, 5, 6, 7, 8, 13, 14\}$ ( )
h) $B \subseteq A$ ( )
i) $\mathbf{U} - (C \cap D) = \{4, 15, 16\}$ ( )
j) $E - (C \cap D) = \{6\}$ ( )
k) $(C \oplus D) = \{1, 2, 3, 5, 9, 10, 11, 12, 14\}$ ( )
l) $D - \mathbf{U} = \varnothing$ ( )
m) $(B - A) = \{5, 8\}$ ( )
n) $3 \in (A \cup B)$ ( )
ñ) $11 \not\subset (C - D)$ ( )
o) $(F \cup E) \subseteq C$ ( )
p) $(C \cup D)' = \{4, 15, 16\}$ ( )
q) $(C \cap E) = \varnothing$ ( )
r) $(E - F) \subseteq D$ ( )
s) $(B - E) \not\subset (D - C)$ ( )

3.10. Considérese el siguiente diagrama de Venn.



Poner en el paréntesis de cada uno de los incisos una "V" si la aseveración es verdadera o bien una "F" si es falsa.

a) $F \subseteq (B - A)$ ( )
b) $A \cap C \neq \varnothing$ ( )
c) $E - D = \{9\}$ ( )
d) $E \cap D = \varnothing$ ( )
e) $(C \cup E) \subseteq B$ ( )
f) $(D - E) \subseteq (A \cap B)$ ( )
g) $C - G = \{3, 10\}$ ( )
h) $G - F = \varnothing$ ( )
i) $(F - C) \subseteq B$ ( )
j) $A - B = \{3, 10, 12, 14\}$ ( )
k) $D \cap E = \{1, 7, 9\}$ ( )
l) $(D \cap E) \subseteq (A \cup B)$ ( )
m) $16 \notin (D \cup E)$ ( )
n) $(A \cup B)' = \{5, 13, 15\}$ ( )
ñ) $B - A = \{2, 4, 8, 11\}$ ( )
o) $(A \cap B) \cup (A - B) = A$ ( )
p) $2 \in U'$ ( )
q) $[U - (A \cup B)] = G \cup \{13, 15\}$ ( )
r) $D \oplus E = \{1, 9\}$ ( )
s) $A - (A \cap B) = C \cup \{12, 14\}$ ( )

3.11.

I. Sean los conjuntos:

$\mathbf{U} = \{a, b, c, d, e, f, g, h, i, j\}$
$A = \{f, g, i, j\}$; $B = \{a, c, d, f, h, i\}$; $C = \{c, d, e, f, g, h\}$; $D = \{a, b, c\}$.

Calcular:

a) $(A \cup B) \cap (C \cup D)$
b) $[(A \cap D) \cup B] - C$
c) $(A \cap C \cap D)' \cup B$
d) $(D \oplus B) \cap A'$
e) $[(A - B) \cap (D \oplus B)] - (C \oplus D')$

II. Sean los conjuntos:

$\mathbf{U} = \{x \mid x \in \mathbf{R}\}$
$A = \{x \mid x \in \mathbf{R};\ x^2 - 1 = 0\}$
$B = \{-1, 2, 4\}$

Calcular:

a) $(A \cup B)'$
b) $(A \cap B)'$
c) $(B - A')$
d) $(A - B) \oplus B'$
e) $(B \oplus (B - A)') \cap A$

3.12

I. Sean los conjuntos:

$\mathbf{U} = \{x \mid x \in \mathbf{Z};\ 0 < x < 100\}$
$A = \{x \mid x \in \mathbf{Z}^+;\ x \text{ es par};\ x < 10\}$
$B = \{1, 2, 4, 5, 6, 7\}$
$C = \{x \mid x \in \mathbf{Z}^+;\ x \text{ es divisible entre } 3;\ x < 16\}$

Calcular:

a) $(A \cap B)' - C$
b) $(A \cup C) \cap B'$

c) $(C - B)' \cap A'$
d) $(B \oplus A) - (C' - A)$
e) $((C - B) \oplus C') - A$

II. Sean los conjuntos:

$\mathbf{U} = \{x \mid x \in \mathbf{Z};\ 2 < x < 5000\}$
$A = \{x \mid x \in \mathbf{Z};\ x \text{ es par};\ 0 < x < 30\}$
$B = \{7, 8, 9, 13, 21, 28\}$
$C = \{x \mid x \in \mathbf{Z};\ x \text{ es primo};\ 0 < x < 20\}$
$D = \{4, 6, 8, 10, 12, 14, 16\}$

Calcular:

a) $[(A \oplus B) - (C \cap B)]'$
b) $[(A' - C') \oplus B'] \cap D'$
c) $[B - (A' \cap C')] \oplus C$
d) $[(D \oplus C') \cap (A - B')] \cup A'$
e) $[(C \oplus B') - (A' - D)] - B$

3.13. Sean los conjuntos:

$\mathbf{U} = \{x \mid x \in \mathbf{Z}\}$
$A = \{x \mid x \in \mathbf{Z};\ x \text{ es primo};\ 5 < x < 30\}$
$B = \{9, 11, 12, 13, 15, 16, 17, 21, 23\}$
$C = \{6, 7, 8, 9, 15, 17, 20, 21, 22, 23, 25\}$
$D = \{x \mid x \in \mathbf{Z};\ x \text{ es impar};\ 10 < x < 20\}$

Calcular:

a) $[B \oplus (C' \cap A)] - D'$
b) $[(B - C) - D'] \cup (A \oplus B')$
c) $[(C' \cup B) \oplus D] - A'$
d) $[B' \oplus (A' \cap C')] - D$
e) $[(A \cap D') - (C' \oplus A')] - B$

3.14. Sean los conjuntos:

$\mathbf{U} = \{x \mid x \in \mathbf{Z}\}$
$A = \{1, 2, 4, 7\}$

$B = \{x \mid x \in \mathbf{Z};\ x \text{ es impar};\ 0 < x < 20;\ x \text{ es divisible entre } 3\}$

$C = \{2, 3, 4, 5, 8, 10\}$

$D = \{x \mid x \in \mathbf{Z};\ x \text{ es primo};\ 1 \leq x < 30\}$

Calcular:

a) $[(A \oplus B) - (C \cup D)]'$

b) $[(B' \cap C) - A'] \oplus (D \oplus B')$

c) $(([D' - B] \cap A) - C') \oplus A'$

d) $[(A' - B') \oplus (C \cap D')] - C'$

e) $[(C \cup D') - (A' \oplus B')] \cap B$

3.15. Usando leyes de conjuntos demostrar que las igualdades de cada uno de los incisos siguientes son verdaderas.

a) $A' \cap B' \cap C' \cup A \cap B' \cap C' \cup A' \cap B \cap C \cup A' \cap B \cap C' \cup A \cap B \cap C \cup A \cap B \cap C' = B \cup C'$

b) $A' \cap B' \cap C \cup A \cap B' \cap C' \cup A \cap B' \cap C \cup A \cap B \cap C \cup A \cap B \cap C' = A \cup B' \cap C$

c) $(((A \cup B')' \cup C)' \cap (C \cup B)')' = B \cup C$

3.16. Usando leyes de conjuntos demostrar que las igualdades de cada uno de los incisos siguientes son verdaderas.

a) $A' \cap B' \cap C \cup A' \cap B \cap C \cup A' \cap B \cap C' \cup A \cap B' \cap C \cup A \cap B \cap C \cup A \cap B \cap C' = B \cup C$

b) $(((A' \cup B)' \cup (C \cup A)')' \cup (B' \cup C)')' = A \cap B' \cup B' \cap C'$

c) $A' \cap B' \cap C' \cup A' \cap B' \cap C \cup A' \cap B \cap C \cup A' \cap B \cap C' \cup A \cap B' \cap C \cup A \cap B \cap C = A' \cup C$

d) $A' \cap B \cap C' \cap D' \cup A' \cap B \cap C \cap D \cup A' \cap B \cap C \cap D' \cup A \cap B \cap C' \cap D' \cup A \cap B \cap C \cap D \cup A \cap B \cap C \cap D' \cup A \cap B' \cap C' \cap D' \cup A \cap B' \cap C \cap D = B \cap C \cup A \cap C \cap D \cup B \cap D' \cup A \cap C' \cap D'$

3.17. Resolver los problemas de los siguientes incisos usando conjuntos finitos:

I. La compañía "Desarrollo de sistemas S.A." necesita contratar 18 personas que programen en Access y 12 personas que programen en Java. De estos programadores se considera que 10 personas saben programar tanto en Access como en Java. ¿Cuántos programadores deberá contratar la compañía?

II. De una muestra de 42 estudiantes de la carrera de informática se obtuvo el siguiente número de reprobados por materia:

- 28 Matemáticas para computación.
- 26 Fundamentos de programación.
- 17 Administración.
- 16 Matemáticas para computación y fundamentos de programación.
- 12 Fundamentos de programación y administración.
- 8 Matemáticas para computación y administración.
- 4 Matemáticas para computación, fundamentos de programación y administración.

a) ¿Cuántos estudiantes no reprobaron ninguna materia de las antes mencionadas?

b) ¿Cuántos estudiantes reprobaron solamente fundamentos de programación?

c) ¿Cuántos estudiantes reprobaron solamente alguna de las tres materias?

d) ¿Cuántos reprobaron matemáticas para computación y fundamentos para programación, pero no administración?

3.18. Resolver los problemas de los siguientes incisos usando conjuntos finitos:

I. De un grupo de 40 alumnos del Tecnológico de Morelia, algunos están estudiando para presentar examen como se indica a continuación:

- 26 Teoría de la computación.
- 18 Redes de computadoras.
- 20 Inteligencia artificial.
- 13 Teoría de la computación y redes de computadoras.
- 8 Redes de computadoras e inteligencia artificial.
- 10 Teoría de la computación e inteligencia artificial.
- 4 estudian las tres asignaturas.

a) ¿Cuántos de ellos no estudian para ninguna de las tres asignaturas?

b) ¿Cuántos de ellos estudian únicamente para inteligencia artificial?

c) ¿Cuántos están estudiando teoría de la computación y redes pero no inteligencia artificial?

II. Se aplicó una encuesta entre los 714 jóvenes que estudian la carrera de ingeniería en sistemas computacionales de una universidad, para conocer las preferencias de especialidad de su carrera. Los resultados obtenidos son:

- 206 prefieren ingeniería del software.
- 291 prefieren sistemas distribuidos.
- 215 prefieren inteligencia artificial.
- 59 prefieren ingeniería del software y sistemas distribuidos.
- 68 prefieren ingeniería del software e inteligencia artificial.
- 80 prefieren sistemas distribuidos e inteligencia artificial.
- 28 se inclinan por las tres especialidades al mismo tiempo.

a) ¿Cuántos prefieren únicamente sistemas distribuidos como especialidad?

b) ¿Cuántos se inclinan por ingeniería del software e inteligencia artificial, pero no por ingeniería del software?

c) ¿Cuántos no pusieron preferencia de especialidad?

3.19. Sean A, B, C, D, E, conjuntos finitos.

a) ¿Cuántos elementos se suman o se restan en la fórmula $|A \cup B \cup C \cup D \cup E|$?

b) ¿Cuáles son los elementos de la fórmula $|A \cup B \cup C \cup D \cup E|$?

3.20. Sean A, B, C, D, E, F conjuntos finitos.

a) ¿Cuántos elementos se suman o se restan en la fórmula $|A \cup B \cup C \cup D \cup E \cup F|$?

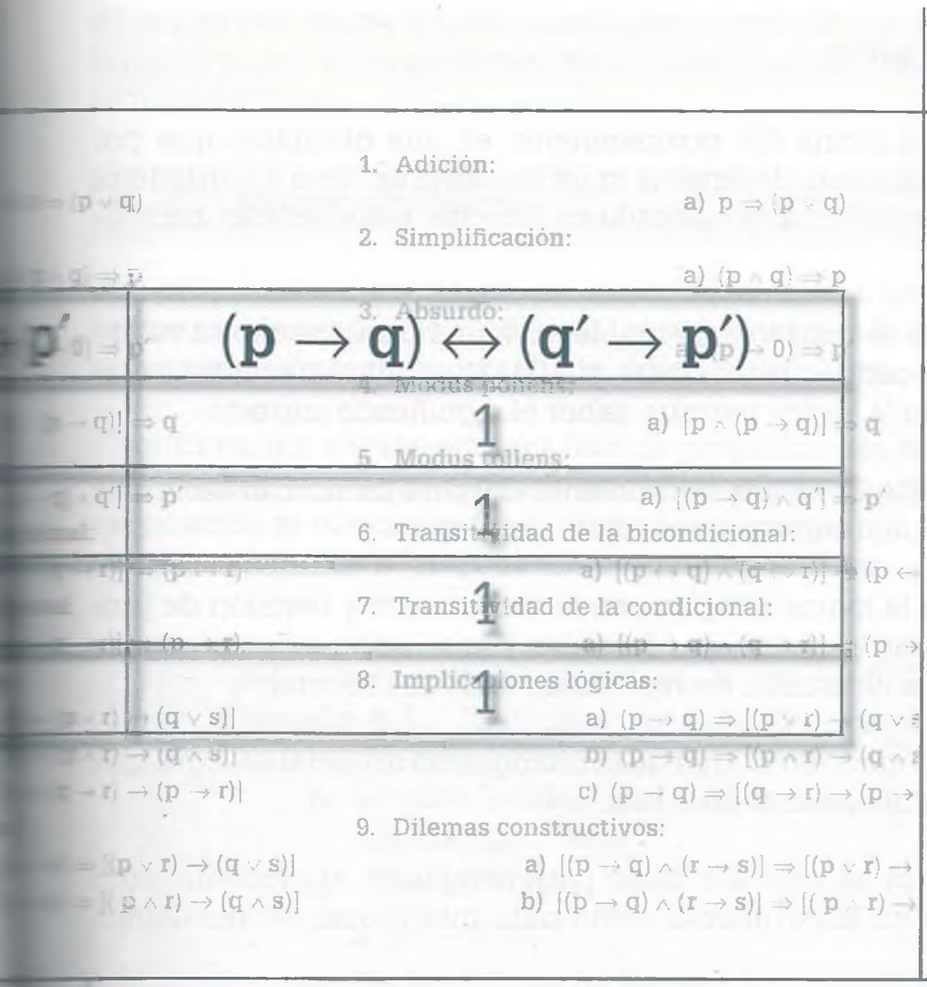
b) ¿Cuáles son los elementos de la fórmula $|A \cup B \cup C \cup D \cup E \cup F|$?

CAPÍTULO IV

Lógica matemática

1. Adición:
   a) $p \Rightarrow (p \vee q)$
2. Simplificación:
   a) $(p \wedge q) \Rightarrow p$
3. Absurdo:
   a) $[p \rightarrow (p \wedge \sim p)] \Rightarrow \sim p$
4. Modus ponens:
   a) $[p \wedge (p \rightarrow q)] \Rightarrow q$
5. Modus tollens:
   a) $[(p \rightarrow q) \wedge \sim q] \Rightarrow \sim p$
6. Transitividad de la bicondicional:
   a) $[(p \leftrightarrow q) \wedge (q \leftrightarrow r)] \Rightarrow (p \leftrightarrow r)$
7. Transitividad de la condicional:
   a) $[(p \rightarrow q) \wedge (q \rightarrow r)] \Rightarrow (p \rightarrow r)$
8. Implicaciones lógicas:
   a) $(p \rightarrow q) \Rightarrow [(p \vee r) \rightarrow (q \vee s)]$
   b) $(p \rightarrow q) \Rightarrow [(p \wedge r) \rightarrow (q \wedge s)]$
   c) $(p \rightarrow q) \Rightarrow [(q \rightarrow r) \rightarrow (p \rightarrow r)]$
9. Dilemas constructivos:
   a) $[(p \rightarrow q) \wedge (r \rightarrow s)] \Rightarrow [(p \vee r) \rightarrow (q \vee s)]$
   b) $[(p \rightarrow q) \wedge (r \rightarrow s)] \Rightarrow [(p \wedge r) \rightarrow (q \wedge s)]$

p	q	p′	q′	$p \rightarrow q$
0	0	1	1	1
0	1	1	0	1
1	0	0	1	0
1	1	0	0	1



$$(p \to q) \leftrightarrow (q' \to p')$$

1

1

1

1

La lógica llena el mundo; los límites del mundo son también sus límites.

Ludwig Wittgenstein

Objetivos

- Entender el concepto de proposición y la forma en que se pueden elaborar proposiciones compuestas usando los conectores lógicos.
- Evaluar proposiciones lógicas por medio de tablas de verdad.
- Comprender los conceptos de tautología, contradicción, equivalencia lógica y regla de inferencia.
- Aprender a representar enunciados en forma de teorema usando para ello simbología lógica.
- Demostrar teoremas por medio del método deductivo directo y contradicción.
- Distinguir entre argumentos válidos y no válidos.
- Representar predicados con notación lógica, usando los cuantificadores existencial y universal.
- Demostrar proposiciones por medio de inducción matemática.

4.1 Introducción

Breve cronología de la lógica

La lógica tuvo su origen en los estudios que llevó a cabo Aristóteles (384-322 a.C.), quien introdujo los cuantificadores universal (∀) y existencial (∃) que ahora se usan frecuentemente en lógica de predicados. Posteriormente en la época de oro de los griegos se trató de usar la lógica para llevar a cabo la demostración formal de las principales leyes matemáticas conocidas, y mucho tiempo después fue Giusseppe Peano el que le dio el nombre de "lógica matemática".

Un desarrollo importante de la lógica matemática ocurrió a mediados del siglo XIX, cuando Leibniz trató de potenciar el razonamiento representando un problema por medio de hipótesis para llegar a una conclusión. También en este siglo George Boole y Augusto De Morgan diseñaron una nueva manera de representar un problema usando para ello los principios fundamentales de la lógica matemática.

A finales del siglo XIX y principios del XX el matemático Británico Alfred North Whitehead y su discípulo Bertrand Russell publicaron los resultados de sus investigaciones que afirmaban que todas las leyes matemáticas se pueden traducir en proposiciones lógicas verdaderas, lo cual significa que el vocabulario matemático es un subconjunto del vocabulario lógico y por lo tanto cualquier demostración lógica es equivalente a cualquier demostración matemática.

Pero fue a mediados del siglo XX cuando la lógica matemática adquirió una destacada importancia con la creación y desarrollo de la computadora, y ahora en el siglo XXI la importancia es mayor al tratar de dotar a las computadoras y robots de una inteligencia artificial que les permita tomar decisiones, al relacionar la información conocida y aplicar reglas de inferencia para llegar a una conclusión.

La lógica estudia la forma del razonamiento, es una disciplina que por medio de reglas y técnicas determina si un teorema es falso o verdadero, además de que es ampliamente aplicada en filosofía, matemáticas, computación y física.

En filosofía la lógica se utiliza para establecer si un razonamiento es válido o no. Tomando en cuenta que una frase puede tener diferentes interpretaciones, en este caso la lógica permite saber el significado correcto.

En matemáticas la lógica es una herramienta útil para demostrar teoremas e inferir resultados, así como para resolver problemas.

En la computación la lógica se aplica en la elaboración y revisión de programas, en el estudio de lenguajes formales y la relación existente entre ellos, así como en la obtención de resultados en forma recursiva.

Con el apoyo de la lógica, en el área de la inteligencia artificial se logra que una máquina tome decisiones precisas.

En la física, la lógica se necesita tanto para establecer el procedimiento para llevar a cabo un experimento como para interpretar los resultados obtenidos.

En general la lógica se aplica en el trabajo cotidiano, por ejemplo para ir de compras al supermercado se tiene que realizar cierto procedimiento lógico que permita realizar dicha tarea, si se desea pintar una pared también se requiere de la aplicación de la lógica.

La lógica es muy importante ya que incluso permite resolver problemas a los que nunca se ha enfrentado el ser humano, utilizando solamente la inteligencia y algunos conocimientos acumulados se pueden crear nuevos inventos, hacer innovaciones a los ya existentes o simplemente utilizar los mismos de tal manera que se obtengan mejores resultados. Al desarrollar la lógica matemática se ejercita el pensamiento abstracto, es posible generalizar la información usando el razonamiento tanto inductivo como deductivo y se pueden llevar a cabo cálculos matemáticos complejos.

Una parte importante de este capítulo es la demostración formal de teoremas, partiendo del hecho de que éstos son la representación de enunciados usando notación lógica. Es importante mencionar que en las demostraciones no hay un procedimiento único para llegar al resultado, éste puede ser más largo o más corto dependiendo de las reglas de inferencia, equivalencias lógicas y tautologías que se utilicen. De hecho puede haber tantas formas de acceder a la solución como personas diferentes tratando de obtenerla. Esto permite que el estudiante tenga confianza en la aplicación

de reglas y fórmulas, de tal manera que cuando llegue a poner en práctica la lógica matemática para resolver un problema sea capaz de encontrar su propia solución.

4.2 Proposiciones

Una proposición o enunciado es una oración, frase o expresión matemática que puede ser falsa o verdadera, pero no ambas a la vez. La proposición es un elemento fundamental de la lógica matemática.

A continuación se presenta una lista de proposiciones válidas y no válidas, y se explica el porqué algunos enunciados no son proposiciones. Cada proposición se indica por medio de una letra minúscula, y luego de los dos puntos se expresa la proposición propiamente dicha.

Ejemplo 4.1. Proposiciones válidas y no válidas.

p: Estados Unidos es el país territorialmente más extenso del continente americano.

q: $-19 + 50 = 31$.

r: $x > (y - 13)7$.

s: Carlos Salinas de Gortari fue presidente de España.

t: Morelia será campeón en la presente temporada de fútbol.

u: ¿Cómo estás?

v: Formatea el disco antes de usarlo.

Las proposiciones **p**, **q** y **s**, tienen un valor de falso o verdadero, por lo tanto son proposiciones válidas. El inciso **r** también es una proposición válida, aunque el valor de falso o verdadero depende del valor asignado a las variables x, y en determinado momento. La proposición **t** está perfectamente expresada, aunque para decir si es falsa o verdadera se tendría que esperar a que terminara la temporada de fútbol, de forma que antes, ahora o después la proposición pueda ser falsa o verdadera. Los enunciados **u** y **v** no son válidos, ya que no pueden tomar un valor de falso o verdadero.

4.2.1 Proposiciones compuestas

Existen conectores u operadores lógicos que permiten formar proposiciones "compuestas". Se dice que una proposición es compuesta cuando está integrada por dos o más proposiciones simples conectadas por medio de

operadores lógicos. A continuación se describen los operadores o conectores lógicos básicos.

Operador *and* (y)

Se utiliza para conectar dos proposiciones que se deben cumplir para que se pueda obtener un resultado verdadero. Su símbolo es $\wedge$:

Ejemplo 4.2. Considérese el siguiente enunciado: "El automóvil arranca si y sólo si el tanque tiene gasolina y la batería tiene corriente."

Sean:

p: El automóvil arranca.
q: El tanque tiene gasolina.
r: La batería tiene corriente.

De esta manera la representación del enunciado anterior, usando simbología lógica, es

$$\mathbf{p = q \wedge r}$$

y su tabla de verdad es la siguiente:

q	r	$p = q \wedge r$
1	1	1
1	0	0
0	1	0
0	0	0

Aquí se tiene que:

1 = verdadero
0 = falso

En la tabla anterior el valor de $\mathbf{q} = 1$ significa que el tanque tiene gasolina, $\mathbf{r} = 1$ significa que la batería tiene corriente y $\mathbf{p = q \wedge r} = 1$ significa que el automóvil puede encender. Se puede notar que si **q** o **r** valen cero, esto implica que el automóvil no tiene gasolina o bien la batería no tiene corriente, y que por lo tanto no puede encender.

Al operador lógico $\wedge$ se le conoce como la multiplicación lógica, porque

$$1 \wedge 1 = 1$$
$$1 \wedge 0 = 0$$
$$0 \wedge 1 = 0$$
$$0 \wedge 0 = 0$$

En lógica matemática en lugar del signo = se utilizan los signos $\equiv$ y $\Leftrightarrow$ para indicar equivalencia lógica, de forma que la proposición del ejemplo anterior puede indicarse como $\mathbf{p} \equiv (\mathbf{q} \wedge \mathbf{r})$ o bien como $\mathbf{p} \Leftrightarrow (\mathbf{q} \wedge \mathbf{r})$.

Operador *or* (o)

Con este operador se obtiene un resultado verdadero cuando alguna de las proposiciones es verdadera. Se indica por medio de los siguientes símbolos: $\{\vee, +, \cup\}$.

Ejemplo 4.3. Se tiene el siguiente enunciado: "Una persona puede entrar al cine si y sólo si compra su boleto o le regalan un pase."

Sean:

p: Una persona entra al cine.
q: Compra su boleto.
r: Le regalan un pase.

De esta manera la representación del enunciado anterior con notación lógica es la siguiente

$$\mathbf{p} \equiv (\mathbf{q} \vee \mathbf{r})$$

y su tabla de verdad es

q	r	$p \equiv (q \vee r)$
1	1	1
1	0	1
0	1	1
0	0	0

A partir de la tabla se ve que la única forma en la que no puede ingresar al cine ($\mathbf{p} \equiv 0$), es que no compre su boleto ($\mathbf{q} \equiv 0$) y que no le regalen un pase ($\mathbf{r} \equiv 0$).

Al operador lógico $\vee$ también se le conoce como la suma lógica, ya que

$$1 \vee 1 \equiv 1$$
$$1 \vee 0 \equiv 1$$
$$0 \vee 1 \equiv 1$$
$$0 \vee 0 \equiv 0$$

Se puede observar que $1 \vee 1 \equiv 1$ se sale de lo esperado ya que $1 + 1 = 2$, sin embargo cuando la suma aritmética es mayor que 1, en lógica matemática y álgebra booleana el resultado se considera 1. Lo único que significa esto es que para que una proposición formada por dos o más proposiciones que se están sumando sea verdadera, es suficiente con que uno de los sumandos sea verdadero. En el ejemplo anterior se considera que ($q \equiv 1$ cuando una persona compra su boleto y además $r \equiv 1$ si a esa misma persona alguien le regala un pase, por lo tanto dicha persona puede entrar al cine aunque le sobre un boleto).

Operador *not* (no)

El operador lógico not tiene como función negar la proposición. Esto significa que si a alguna proposición verdadera se le aplica el operador not, se obtendrá su complemento o negación. Este operador se indica por medio de los siguientes símbolos: {′, ¬, ¯, ~}.

La tabla de verdad relacionada con el operador not es la siguiente

p	p′
1	0
0	1

La negación o complemento de una función, es el valor contrario. Si $\mathbf{p} = 1$ su complemento en binario es $\mathbf{p}' = 0$.

Ejemplo 4.4. Sea **p**: "El automóvil es azul"; entonces su complemento es **p′**: "El automóvil no es azul".

Una doble negación de una proposición es equivalente a afirmar la proposición, esto es, $\mathbf{p} \equiv \mathbf{p}''$. Si una proposición tiene un número impar de negaciones es como si sólo tuviera una, por ejemplo $\mathbf{p}''' \equiv \mathbf{p}'$. Por otro lado un número par de negaciones equivale a una proposición verdadera $\mathbf{p}'''' \equiv \mathbf{p}$.

Operador *or* exclusivo (xor)

Además de los operadores básicos (and, or y not) existe el operador xor, cuyo funcionamiento es semejante al de or con la diferencia de que su resultado es verdadero solamente si una de las proposiciones es cierta, ya que cuando ambas son verdad el resultado es falso. Este operador se indica por medio del símbolo ($\oplus$) y su tabla de verdad es la siguiente

p	q	$p \oplus q$
1	1	0
1	0	1
0	1	1
0	0	0

Como se ve a partir de la tabla, se obtiene un resultado verdadero sólo cuando una de las proposiciones es verdadera, pero no si ambas lo son:

$$p \oplus q \equiv p' \wedge q \vee p \wedge q'$$

Finalmente, con ayuda de estos operadores básicos se pueden formar los operadores compuestos **Nand** (combinación de Not y And), **Nor** (combinación de Not y Or) y **Xnor** (combinación de Xor y Not), los cuales se tratarán con mayor detenimiento en el capítulo de álgebra booleana.

4.2.2 Proposición condicional ($\rightarrow$)

Una proposición condicional es aquella que está formada por dos proposiciones simples (o compuestas) **p** y **q**, y que se indica de la siguiente manera:

$$p \rightarrow q$$

Esto se lee "si **p** entonces **q**".

Ejemplo 4.5. Considérese que un candidato a la presidencia de México dice: "Si salgo electo presidente de la República, entonces el crecimiento anual del país será del 7%."

Una declaración como ésta se conoce como condicional, y para analizarla sean las proposiciones:

p: Salió electo Presidente de la República.

q: El crecimiento anual fue del 7%.

De esta forma el enunciado anterior se puede expresar como

$$\mathbf{p} \rightarrow \mathbf{q}$$

y su tabla de verdad es la siguiente:

p	q	p → q
1	1	1
1	0	0
0	1	1
0	0	1

En esta tabla hay que observar que el único caso en el que $(\mathbf{p} \rightarrow \mathbf{q})$ es 0 es cuando $\mathbf{p} \equiv 1$ y $\mathbf{q} \equiv 0$.

La interpretación de los resultados de la tabla es la siguiente:

1) $\mathbf{p} \equiv 1$ significa que "el candidato salió electo", mientras que $\mathbf{q} \equiv 1$ significa que "el crecimiento anual del país fue de 7%", por lo tanto $(\mathbf{p} \rightarrow \mathbf{q}) \equiv 1$ indica que el candidato dijo la verdad en su campaña.
2) $\mathbf{p} \equiv 1$ y $\mathbf{q} \equiv 0$ significa que el candidato mintió, ya que salió electo y el crecimiento anual no fue del 7% como lo prometió, por lo tanto la afirmación del candidato es falsa: $(1 \rightarrow 0) \equiv 0$.
3) $\mathbf{p} \equiv 0$ y $\mathbf{q} \equiv 1$ significa que aunque no salió electo hubo un crecimiento del 7% anual en el país, crecimiento que posiblemente fue ajeno al candidato presidencial y por lo tanto tampoco mintió, de tal forma que $(0 \rightarrow 1) \equiv 1$.
4) $\mathbf{p} \equiv 0$ y $\mathbf{q} \equiv 0$ significa que el candidato no salió electo y que tampoco el crecimiento anual del país fue del 7%, por lo tanto el candidato no mintió respecto a la afirmación que hizo en su campaña, por lo que $(0 \rightarrow 0) \equiv 1$.

4.2.3 Proposición bicondicional (↔)

Sean **p** y **q** dos proposiciones, entonces se puede indicar la proposición bicondicional de la siguiente forma:

$$\mathbf{p} \leftrightarrow \mathbf{q}$$

Esto se lee como "**p** si sólo si **q**" en donde la proposición que representa el enunciado ($\mathbf{p} \leftrightarrow \mathbf{q}$) es verdadera si **p** es verdadera si y sólo si **q** también lo es. O bien la proposición es verdadera si **p** es falsa y si sólo si **q** también lo es.

Ejemplo 4.6. Considérese el enunciado "Es buen estudiante, si y sólo si, tiene promedio de diez."

Para representar esto con notación lógica en forma de proposición bicondicional se definen las proposiciones

p: Es buen estudiante.

q: Tiene promedio de diez.

La tabla de verdad correspondiente es la siguiente:

p	q	$p \leftrightarrow q$
1	1	1
1	0	0
0	1	0
0	0	1

Como se ve en la tabla, la proposición bicondicional solamente es verdadera si tanto **p** como **q** son falsas o bien si ambas son verdaderas.

Usando los diferentes operadores lógicos expuestos, se pueden representar con notación lógica enunciados compuestos con más de una proposición.

Ejemplo 4.7. Representar con notación lógica los siguientes enunciados:

a) "Si no estudio matemáticas para computación y no hago la tarea de fundamentos de programación, entonces reprobaré el semestre o no podré ir de vacaciones a Cancún."

El enunciado anterior es una proposición condicional integrada por varias proposiciones simples, y para representarlo con notación lógica lo primero

que se hace es determinar cuáles son las proposiciones simples que la integran para asignarles un nombre. En este caso se tienen las siguientes:

p: Estudio matemáticas para computación.
q: Hago la tarea de fundamentos de programación.
r: Reprobaré el semestre.
s: Podré ir de vacaciones a Cancún.

Usando esto y los operadores correspondientes, el enunciado se expresa como

$$(\mathbf{p}' \wedge \mathbf{q}') \rightarrow (\mathbf{r} \vee \mathbf{s}')$$

El enunciado anterior no marca explícitamente en dónde se deben de poner paréntesis, sin embargo se puede inferir que si hay más de una proposición antes de la palabra "entonces" o después de ella, éstas se deben de encerrar entre paréntesis para no tener problemas con la jerarquía de operación de los conectores lógicos.

b) "Si no pago el teléfono, entonces me cortarán el servicio telefónico. Y si pago el teléfono, entonces me quedaré sin dinero o pediré prestado. Y si me quedo sin dinero y pido prestado, entonces no podré pagar la tarjeta de crédito, si sólo si soy una persona desorganizada."

En este caso se tienen las siguientes proposiciones simples:

p: Pago el teléfono.
q: Me cortarán el servicio telefónico.
r: Me quedaré sin dinero.
s: Pediré prestado.
t: Pagar la tarjeta de crédito.
w: Soy una persona desorganizada.

Usando esto y los operadores, el enunciado dado se expresa como

$$(\mathbf{p}' \rightarrow \mathbf{q}) \wedge [\mathbf{p} \rightarrow (\mathbf{r} \vee \mathbf{s})] \wedge [(\mathbf{r} \wedge \mathbf{s}) \rightarrow \mathbf{t}'] \leftrightarrow \mathbf{w}$$

Es conveniente encerrar entre paréntesis cada uno de los textos separados por punto, ya que cada uno de estos textos representa una hipótesis (varias hipótesis juntamente con su conclusión son parte de un teorema, como se verá posteriormente). Se puede observar en el enunciado dado que después de un punto y seguido aparece un conector lógico "Y". En general un punto y seguido significa un operador lógico "$\wedge$" sin necesidad de ponerlo explícitamente, por lo que en los siguientes ejercicios ya no se pondrá el operador lógico sino solamente el punto.

Es más complicado representar correctamente las proposiciones por medio de texto que por medio de notación lógica, ya que por lo general no se usan los paréntesis para agrupar información lo cual genera ambigüedad. Esto no sucede en matemáticas, ya que los paréntesis permiten la evaluación de la proposición respetando la jerarquía de operación de los diferentes operadores lógicos y la agrupación de la información cuando es necesario, ya sea para hacer más clara la proposición o bien para alterar el orden de evaluación. Sin embargo, nunca se debe abusar de los paréntesis ya que en lugar de hacer más clara la proposición la complican.

4.3 Tablas de verdad

Por medio de una tabla de verdad es posible mostrar los resultados obtenidos al aplicar cada uno de los operadores lógicos, así como el resultado de la proposición para todos y cada uno de los valores que pueden tener las diferentes proposiciones simples que integran una proposición compuesta. Con la tabla de verdad se puede observar con claridad el comportamiento particular y generalizado de una proposición y, con base en ello, determinar sus propiedades y características.

Tablas de verdad

Las tablas de valores de verdad son una herramienta desarrollada por Charles Peirce en la década de 1880, siendo sin embargo más popular el formato que Ludwig Wittgenstein desarrolló en su *Tractatus logico-philosophicus*, publicado en 1918 por Bertrand Russell.

Se emplean en lógica para determinar los posibles valores de verdad de una expresión o proposición.

Una tabla de verdad está formada por filas y columnas, y el número de filas depende del número de proposiciones diferentes que conforman una proposición compuesta. Asimismo, el número de columnas depende del número de proposiciones que integran la proposición y del número de operadores lógicos contenidos en la misma.

En general se tiene la siguiente expresión:

$$\text{Número de filas} = 2^n$$

donde n es el número de proposiciones diferentes que integran una proposición compuesta.

Ejemplo 4.8. Construir la tabla de verdad de la siguiente proposición:

$$[(\mathbf{p} \rightarrow \mathbf{q}) \vee (\mathbf{q}' \wedge \mathbf{r})] \leftrightarrow (\mathbf{r} \rightarrow \mathbf{q})$$

En este caso se tiene que Número de filas = $2^3 = 8$ porque son tres las proposiciones diferentes ($\mathbf{p}$, $\mathbf{q}$, $\mathbf{r}$) que integran la proposición. Aunque también está $\mathbf{q}'$ en la proposición compuesta anterior, se entiende que conociendo el valor de $\mathbf{q}$ es posible conocer el de $\mathbf{q}'$, por lo tanto, se trata de la misma proposición.

Charles Sanders Peirce
(1839-1914)

Nació en Cambridge, Massachussets, Estados Unidos (10 de septiembre de 1839-19 de abril de 1914) y fue filósofo, lógico y científico. Es considerado el fundador del pragmatismo y el padre de la semiótica moderna.

Fue profesor de astronomía y matemáticas en Harvard, y aunque se graduó en química en la Universidad de Harvard, nunca logró tener una posición académica permanente a causa de su difícil personalidad (tal vez maniaco-depresiva) y del escándalo que rodeó a su segundo matrimonio después de divorciarse de su primera mujer, Melusina Fay. Desarrolló su carrera profesional como científico en la United Status Coast Survey (1859-1891), trabajando especialmente en astronomía, en geodesia y en medidas pendulares. Desde 1879 hasta 1884 fue profesor de lógica a tiempo parcial en la Universidad Johns Hopkins. Tras retirarse en 1888 se estableció con su segunda mujer, Juliette Froissy, en Milford, donde murió de cáncer después de 26 años de escritura intensa y prolífica.

Ludwig Josef Johann Wittgenstein
(1889-1951)

Es uno de los filósofos modernos fundamentales que en vida publicó solamente un libro: el *Tractatus logico-philosophicus*, que influyó en gran medida a los positivistas lógicos del Círculo de Viena, movimiento del que nunca se consideró parte. Tiempo después, el *Tractatus* fue severamente criticado por el propio Wittgenstein en *Los cuadernos azul y marrón* y en sus *Investigaciones filosóficas*, ambas obras póstumas. Fue discípulo de Bertrand Russell en el Trinity College de Cambridge, donde más tarde también él llegó a ser profesor.

De acuerdo con lo anterior la tabla es la siguiente:

p	q	r	q′	$p \to q$	$(q' \wedge r)$	$(p \to q) \vee (q' \wedge r)$	$r \to q$	$[(p \to q) \vee (q' \wedge r)] \leftrightarrow (r \to q)$
0	0	0	1	1	0	1	1	1
0	0	1	1	1	1	1	0	0
0	1	0	0	1	0	1	1	1
0	1	1	0	1	0	1	1	1
1	0	0	1	0	0	0	1	0
1	0	1	1	0	1	1	0	0
1	1	0	0	1	0	1	1	1
1	1	1	0	1	0	1	1	1

En la tabla de verdad es conveniente colocar los valores de las proposiciones con cierto orden, ya que una tabla de verdad ordenada permite una revisión más rápida. Se debe de tener presente que aunque no se alteran los resultados si no se guarda un orden, al ordenar la tabla sí se cambia la colocación de los mismos. El orden recomendado es primero colocar las proposiciones ordenadas alfabéticamente y los valores de las mismas de menor a mayor (000, 001, 010, ..., 111), y luego las proposiciones complemento requeridas.

Por otro lado, al llevar a cabo la evaluación se debe de aplicar la siguiente jerarquía de operación:

Jerarquía	Operador
1ª.	( )
2ª.	′
3ª.	$\wedge$
4ª.	$\vee$
5ª.	$\to \leftrightarrow$

De acuerdo con esta tabla, lo primero que se evalúa en una proposición es lo que se encuentra entre paréntesis, después la negación, posteriormente la intersección y la unión, y finalmente la condicional y la bicondicional.

Cuando existe más de un paréntesis se evalúa primero el que se encuentra más adentro y de izquierda a derecha, es decir, si se encuentran dos pa

réntesis de forma que no está uno dentro de otro, se evalúa primero el que se encuentra más a la izquierda. Lo mismo ocurre con los operadores condicional y bicondicional, que aunque tienen la misma jerarquía de operación se evalúa primero el que se encuentre más a la izquierda.

Ejemplo 4.9. La tabla de verdad de la proposición $p' \rightarrow (r' \vee q \wedge p) \leftrightarrow r \vee q' \rightarrow p$ es la siguiente:

p	q	r	p′	q′	r′	q ∧ p	(r′ ∨ q ∧ p)	(r ∨ q′)	p′ → (r′ ∨ q ∧ p)	p′ → (r′ ∨ q ∧ p) ↔ r ∨ q′	F
0	0	0	1	1	1	0	1	1	1	1	0
0	0	1	1	1	0	0	0	1	0	0	1
0	1	0	1	0	1	0	1	0	1	0	1
0	1	1	1	0	0	0	0	1	0	0	1
1	0	0	0	1	1	0	1	1	1	1	1
1	0	1	0	1	0	0	0	1	1	1	1
1	1	0	0	0	1	1	1	0	1	0	1
1	1	1	0	0	0	1	1	1	1	1	1

Aquí se tiene que

$$F = p' \rightarrow (r' \vee q \wedge p) \leftrightarrow r \vee q' \rightarrow p$$

Como se ve, primero se evalúa la información dentro del paréntesis, pero incluso dentro de él se debe de respetar la jerarquía de operación por lo que primero se evalúa la $\wedge$ y después la $\vee$, luego como existe otra unión se debe llevar a cabo esta operación. Posteriormente se aplica el operador $\rightarrow$ de la izquierda, ya que aunque su jerarquía es igual que la $\rightarrow$ de la derecha y la $\leftrightarrow$, su posición más a la izquierda le otorga prioridad. La evaluación continúa con la $\leftrightarrow$ y finalmente la $\rightarrow$ de la derecha.

4.3.1 Tautología, contradicción y contingencia

Tautología es aquella proposición (compuesta) que es cierta para todos los valores de verdad de sus variables. Un ejemplo típico es $(p' \vee p)$, ya que el resultado es verdadero para todos los valores que puede tener p, como se muestra en la siguiente tabla de verdad:

p	p′	p ∨ p′
1	0	1
0	1	1

Otro ejemplo es $(p \rightarrow q) \leftrightarrow (q' \rightarrow p')$, cuya tabla de verdad es la siguiente:

p	q	p′	q′	$p \rightarrow q$	$q' \rightarrow p'$	$(p \rightarrow q) \leftrightarrow (q' \rightarrow p')$
0	0	1	1	1	1	1
0	1	1	0	1	1	1
1	0	0	1	0	0	1
1	1	0	0	1	1	1

Las tautologías son muy importantes en lógica matemática, ya que al tener un resultado verdadero para todos los valores de verdad, se consideran leyes que se pueden utilizar para realizar demostraciones de teoremas o para inferir resultados de proposiciones desconocidas. Existen varias tautologías conocidas y a continuación se listan las más comunes que, por supuesto, es posible verificar por medio de su tabla de verdad correspondiente:

Tabla 4.1 Tautologías comunes

1. **Adición:**
   a) $p \Rightarrow (p \vee q)$
2. **Simplificación:**
   a) $(p \wedge q) \Rightarrow p$
3. **Absurdo:**
   a) $(p \rightarrow 0) \Rightarrow p'$
4. **Modus ponens:**
   a) $[p \wedge (p \rightarrow q)] \Rightarrow q$
5. **Modus tollens:**
   a) $[(p \rightarrow q) \wedge q'] \Rightarrow p'$
6. **Transitividad de la bicondicional:**
   a) $[(p \leftrightarrow q) \wedge (q \leftrightarrow r)] \Rightarrow (p \leftrightarrow r)$
7. **Transitividad de la condicional:**
   a) $[(p \rightarrow q) \wedge (q \rightarrow r)] \Rightarrow (p \rightarrow r)$
8. **Extensión de la condicional:**
   a) $(p \rightarrow q) \Rightarrow [(p \vee r) \rightarrow (q \vee s)]$
   b) $(p \rightarrow q) \Rightarrow [(p \wedge r) \rightarrow (q \wedge s)]$
   c) $(p \rightarrow q) \Rightarrow [(q \rightarrow r) \rightarrow (p \rightarrow r)]$
9. **Dilemas constructivos:**
   a) $[(p \rightarrow q) \wedge (r \rightarrow s)] \Rightarrow [(p \vee r) \rightarrow (q \vee s)]$
   b) $[(p \rightarrow q) \wedge (r \rightarrow s)] \Rightarrow [(p \wedge r) \rightarrow (q \wedge s)]$

Todas las tautologías de la lista anterior tienen la siguiente forma:

$$P \Rightarrow Q$$

esto es, si **P** entonces **Q**.

Cuando se tiene una proposición con letras mayúsculas, como en el caso anterior, se entiende que esa letra mayúscula equivale a una proposición compuesta y que como tal está formada por varias proposiciones más simples indicadas con letras minúsculas y conectadas por medio de operadores lógicos.

Para probar que las proposiciones anteriores son tautologías, se debe de cambiar el símbolo $\Rightarrow$ por $\rightarrow$ y evaluar la proposición en la forma normal. Por ejemplo, la tabla de verdad de $[(p \rightarrow q) \wedge q'] \Rightarrow p'$ es:

p	q	p'	q'	$p \rightarrow q$	$(p \rightarrow q) \wedge q'$	$[(p \rightarrow q) \wedge q'] \rightarrow p'$
0	0	1	1	1	1	1
0	1	1	0	1	0	1
1	0	0	1	0	0	1
1	1	0	0	1	0	1

4.3.2 Contradicción

Se dice que una proposición es una contradicción o "absurdo" si al evaluar esa proposición el resultado es falso, para todos los valores de verdad. La contradicción más conocida es $(p \wedge p')$ como se muestra en la siguiente tabla de verdad.

p	p'	$p \wedge p'$
0	1	0
1	0	0

Por ejemplo considérese

p: La puerta es verde.

Entonces la proposición $p \wedge p'$ equivale a decir que "La puerta es verde y la puerta no es verde". Por lo tanto, ocurre una contradicción.

La contradicción $p \wedge p'$ se usa con frecuencia en la demostración de teoremas, ya que si en ésta se obtiene que **p** es verdadera y p' también lo es, resulta que $p \wedge p'$ es verdadera, pero como se sabe que esto es una contradicción entonces se puede concluir que el teorema es falso. Más adelante se verá con todo detalle la utilidad de la contradicción.

4.3.3 Contingencia

Una proposición compuesta cuyos valores, en sus diferentes líneas de la tabla de verdad, dan como resultado unos y ceros se llama contingencia, inconsistencia o falacia. Prácticamente cualquier proposición que se invente por lo general es una contingencia. Considérese el siguiente ejemplo:

p	q	p'	q'	$q' \vee p$	$(q' \vee p) \rightarrow p'$	$[(q' \vee p) \rightarrow p'] \wedge q$
0	0	1	1	1	1	0
0	1	1	0	0	1	1
1	0	0	1	1	0	0
1	1	0	0	1	0	0

Tomando en cuenta el resultado final de esta tabla se dice que se trata de una contingencia.

4.4 Inferencia lógica

Inferencia lógica

En relación con la inferencia lógica se tienen la inferencia inductiva en la que el proceso lógico va de lo particular a lo general, la inferencia deductiva que se caracteriza por ir de lo general a lo particular y por tener asociados los modos de inferencia conocidos como **modus ponendo ponens** y **modus tollendo tollens**, y la inferencia transductiva que va de lo particular a lo particular o de lo general a lo general.

Los argumentos basados en tautologías representan métodos de razonamiento universalmente correctos. Su validez depende solamente de la forma de las proposiciones que intervienen y no de los valores de verdad de las variables que contienen. A esos argumentos y a la forma en que se relacionan entre sí se les llama **reglas de inferencia**, y éstas permiten relacionar dos o más proposiciones para obtener una tercera que es válida en una demostración.

Ejemplo 4.10. Considérese el siguiente argumento:

- Si es un gato, entonces come carne.
- Si come carne, entonces es felino.

$\therefore$ Si es un gato, entonces es felino.

Sean las proposiciones:

p: Es un gato.
q: Come carne.
r: Es felino.

Utilizando éstas, el argumento anterior se puede representar con notación lógica de la siguiente manera:

$$\begin{array}{l} \mathbf{p} \rightarrow \mathbf{q} \\ \mathbf{q} \rightarrow \mathbf{r} \\ \hline \therefore \mathbf{p} \rightarrow \mathbf{r} \end{array}$$

Obsérvese que en esta regla de inferencia se parte de que las proposiciones $\mathbf{p} \rightarrow \mathbf{q}$ y $\mathbf{q} \rightarrow \mathbf{r}$ son verdaderas, porque son hipótesis y parte del enunciado, para obtener con ellas y la inferencia lógica la proposición $\mathbf{p} \rightarrow \mathbf{r}$ que también se considera válida. Con esto no se quiere decir que las tres proposiciones son tautologías y que sus resultados en una tabla de verdad son siempre verdaderos en todos sus casos, sino que dichas hipótesis y la proposición obtenida con la regla de inferencia deberán considerarse verdaderas, porque integrándolas forman un argumento válido y por supuesto verdadero (posteriormente se verá cuáles deben ser las características de los argumentos válidos).

Ejemplo 4.11. Considérese el siguiente argumento:

- Bajan los impuestos.
- Si bajan los impuestos, entonces el ingreso se eleva.

∴ El ingreso se eleva

Sean las proposiciones:

p: Bajan los impuestos.
q: El ingreso se eleva.

Utilizando éstas, el argumento anterior se puede representar con notación lógica de la siguiente manera:

$$\begin{array}{l} \mathbf{p} \\ \mathbf{p} \rightarrow \mathbf{q} \\ \hline \therefore \mathbf{q} \end{array}$$

En el ejemplo 4.10 se aplicó una regla de inferencia conocida como "silogismo hipotético", mientras que en el ejemplo 4.11 se utilizó la que se conoce como "Modus ponens". Las proposiciones a las que se les aplica una regla de inferencia pueden ser bastante complejas, sin embargo la proposición obtenida será válida siempre y cuando se respete la forma de la regla de inferencia.

Ejemplo 4.12. Considérese el siguiente argumento:

$$\begin{array}{l}((p \to s') \vee q) \to (q' \wedge s) \\ (q' \wedge s) \to (s' \vee p) \\ \hline \therefore ((p \to s') \vee q) \to (s' \vee p)\end{array}$$

Obsérvese cómo en este caso se está aplicando el silogismo hipotético mostrado en el ejemplo 4.10. Aquí **p** es $((p \to s') \vee q)$, **q** es $(q' \wedge s)$ y **r** es $(s' \vee p)$.

En la tabla 4.2 se listan las principales reglas de inferencia que se pueden aplicar en una demostración.

Tabla 4.2 Reglas de inferencia

10. **Adición**

$$\begin{array}{l}p \\ \hline \therefore p \vee q\end{array}$$

11. **Simplificación**

$$\begin{array}{l}p \wedge q \\ \hline \therefore p\end{array}$$

12. **Silogismo disyuntivo**

$$\begin{array}{l}p \vee q \\ p' \\ \hline \therefore q\end{array}$$

13. **Silogismo hipotético**

$$\begin{array}{l}p \to q \\ q \to r \\ \hline \therefore p \to r\end{array}$$

14. **Conjunción**

$$\begin{array}{l}p \\ q \\ \hline \therefore p \wedge q\end{array}$$

15. **Modus ponens**

$$\begin{array}{l}p \\ p \to q \\ \hline \therefore q\end{array}$$

16. **Modus tollens**

$$\begin{array}{l}p \to q \\ q' \\ \hline \therefore p'\end{array}$$

Las reglas de inferencia permiten la creación de nuevas proposiciones a partir de información conocida. Posiblemente la obtención de la nueva proposición no sea difícil, pero sí el determinar qué regla de inferencia se deberá usar para obtener una proposición que sea de utilidad.

4.5 Equivalencia lógica

Se dice que dos proposiciones son lógicamente equivalentes, o simplemente equivalentes, si coinciden sus resultados para los mismos valores de verdad, y se indican como $\mathbf{p} \equiv \mathbf{q}$ o bien como $\mathbf{p} \Leftrightarrow \mathbf{q}$.

Ejemplo 4.13. Considérese la siguiente tabla.

p	q	p′	q′	p → q	q → p	q′ → p′	(p → q′) ∧ (q → p)	p ↔ q
0	0	1	1	1	1	1	1	1
0	1	1	0	1	0	1	0	0
1	0	0	1	0	1	0	0	0
1	1	0	0	1	1	1	1	1

Equivalentes Equivalentes

En esta tabla se puede observar que $\mathbf{p} \rightarrow \mathbf{q}$ es lógicamente equivalente a su contra positiva $\mathbf{q}' \rightarrow \mathbf{p}'$ ya que coinciden en todas sus líneas, por lo tanto, se dice que $(\mathbf{p} \rightarrow \mathbf{q}) \equiv (\mathbf{q}' \rightarrow \mathbf{p}')$. También la intersección de una proposición condicional con su recíproca es lógicamente equivalente a la proposición bicondicional, de manera que $[(\mathbf{p} \rightarrow \mathbf{q}) \wedge (\mathbf{q} \rightarrow \mathbf{p})] \equiv (\mathbf{p} \leftrightarrow \mathbf{q})$.

Existen varias proposiciones lógicamente equivalentes, que son de gran utilidad en la demostración de teoremas; en la tabla 4.3 se presenta una lista de éstas.

Tabla 4.3 Proposiciones equivalentes

17. **Doble negación**
    a) $\mathbf{p}'' \equiv \mathbf{p}$

18. Leyes conmutativas

a) $(p \vee q) \equiv (q \vee p)$
b) $(p \wedge q) \equiv (q \wedge p)$
c) $(p \leftrightarrow q) \equiv (q \leftrightarrow p)$

19. Leyes asociativas

a) $[(p \vee q) \vee r] \equiv [p \vee (q \vee r)]$
b) $[(p \wedge q) \wedge r] \equiv [p \wedge (q \wedge r)]$

20. Leyes distributivas

a) $[p \vee (q \wedge r)] \equiv [(p \vee q) \wedge (p \vee r)]$
b) $[p \wedge (q \vee r)] \equiv [(p \wedge q) \vee (p \wedge r)]$

21. Leyes de idempotencia

a) $(p \vee p) \equiv p$
b) $(p \wedge p) \equiv p$

22. Leyes de Morgan

a) $(p \vee q)' \equiv (p' \wedge q')$
b) $(p \wedge q)' \equiv (p' \vee q')$

23. Contrapositiva

a) $(p \rightarrow q) \equiv (q' \rightarrow p')$

24. Variantes de la condicional

a) $(p \rightarrow q) \equiv (p' \vee q)$
b) $(p \rightarrow q) \equiv (p \wedge q')'$
c) $(p \vee q) \equiv (p' \rightarrow q)$
d) $(p \wedge q) \equiv (p \rightarrow q')'$
e) $[(p \rightarrow r) \wedge (q \rightarrow r)] \equiv [(p \wedge q) \rightarrow r]$
f) $[(p \rightarrow q) \wedge (p \rightarrow r)] \equiv [p \rightarrow (q \wedge r)]$

25. Variantes de la bicondicional

a) $(p \leftrightarrow q) \equiv [(p \rightarrow q) \wedge (q \rightarrow p)]$
b) $(p \leftrightarrow q) \equiv [(p' \vee q) \wedge (q' \vee p)]$
c) $(p \leftrightarrow q) \equiv [(p \wedge q) \vee (p' \wedge q')]$

26. Contradicción

a) $(p \wedge p') \equiv 0$

27. Ley de identidad

a) $(p \vee 0) \equiv p$
b) $(p \vee 1) \equiv 1$
c) $(p \wedge 0) \equiv 0$
d) $(p \vee p') \equiv 1$
e) $(p \wedge 1) \equiv p$
f) $(p \wedge q \vee q) \equiv q$

28. Disyunción exclusiva

a) $(p \oplus q) \equiv (p \leftrightarrow q)'$

Augustus De Morgan
(1806-1871)

Fue un matemático y lógico inglés nacido en la India. Fue profesor de matemáticas en el Colegio Universitario de Londres entre 1828 y 1866, y el primer presidente de la Sociedad de Matemáticas de Londres. De Morgan se interesó especialmente por el álgebra y escribió varias obras de lógica. En la moderna lógica matemática, llevan el nombre de De Morgan las siguientes leyes fundamentales del álgebra de la lógica: «la negación de la conjunción es equivalente a la disyunción de las negaciones»; «la negación de la disyunción es equivalente a la conjunción de las negaciones».

Su obra principal es *La lógica formal o el cálculo de inferencias necesarias y probables* (1847).

Es posible demostrar que dos proposiciones son lógicamente equivalentes, no sólo por medio de una tabla de verdad como se hizo anteriormente, sino también con apoyo de las restantes equivalencias lógicas.

Ejemplo 4.14. Usando equivalencias lógicas demostrar que:

$$(p \leftrightarrow q) \equiv [(p \rightarrow q) \wedge (q \rightarrow p)]$$

Demostración:

$(p \leftrightarrow q) \equiv [(p \rightarrow q) \wedge (q \rightarrow p)]$
$(p' \vee q) \wedge (q' \vee p) \equiv (p' \vee q) \wedge (q' \vee p)$ Usando equivalencias 25b y 24a
$(p' \vee q) \wedge (p \vee q') \equiv (p' \vee q) \wedge (p \vee q')$ Aplicando 18a

De aquí se nota claramente que se trata de una equivalencia lógica.

Debido a su utilidad en la demostración de teoremas, en la tabla 4.4 se presentan las principales tautologías, reglas de inferencia y equivalencias lógicas.

Tabla 4.4 Expresiones útiles para la demostración de teoremas*

Tautologías	Reglas de inferencia	Equivalencias lógicas
Adición a) $p \Rightarrow (p \vee q)$ **Simplificación** a) $(p \wedge q) \Rightarrow p$ **Absurdo** a) $(p \rightarrow 0) \Rightarrow p'$ **Modus ponens** a) $[p \wedge (p \rightarrow q)] \Rightarrow q$ **Modus tollens** a) $[(p \rightarrow q) \wedge q'] \Rightarrow p'$ **Transitividad de la bicondicional** a) $[(p \leftrightarrow q) \wedge (q \leftrightarrow r)] \Rightarrow (p \leftrightarrow r)$ **Transitividad de la condicional** a) $[(p \rightarrow q) \wedge (q \rightarrow r)] \Rightarrow (p \rightarrow r)$ **Extensión de la condicional** a) $(p \rightarrow q) \Rightarrow [(p \vee r) \rightarrow (q \vee s)]$ b) $(p \rightarrow q) \Rightarrow [(p \wedge r) \rightarrow (q \wedge s)]$ c) $(p \rightarrow q) \Rightarrow [(q \rightarrow r) \rightarrow (p \rightarrow r)]$ **Dilemas constructivos** a) $[(p \rightarrow q) \wedge (r \rightarrow s)] \Rightarrow [(p \vee r) \rightarrow (q \vee s)]$ b) $[(p \rightarrow q) \wedge (r \rightarrow s)] \Rightarrow [(p \wedge r) \rightarrow (q \wedge s)]$	10. **Adición** p $\therefore p \vee q$ 11. **Simplificación** $p \wedge q$ $\therefore p$ 12. **Silogismo disyuntivo** $p \vee q$ p' $\therefore q$ 13. **Silogismo hipotético** $p \rightarrow q$ $q \rightarrow r$ $\therefore p \rightarrow r$ 14. **Conjunción** p q $\therefore p \wedge q$ 15. **Modus ponens** p $p \rightarrow q$ $\therefore q$ 16. **Modus tollens** $p \rightarrow q$ q' $\therefore p'$	17. **Doble negación** a) $p'' \equiv p$ 18. **Leyes conmutativas** a) $(p \vee q) \equiv (q \vee p)$ b) $(p \wedge q) \equiv (q \wedge p)$ c) $(p \leftrightarrow q) \equiv (q \leftrightarrow p)$ 19. **Leyes asociativas** a) $[(p \vee q) \vee r] \equiv [p \vee (q \vee r)]$ b) $[(p \wedge q) \wedge r] \equiv [p \wedge (q \wedge r)]$ 20. **Leyes distributivas** a) $[p \vee (q \wedge r)] \equiv [(p \vee q) \wedge (p \vee r)]$ b) $[p \wedge (q \vee r)] \equiv [(p \wedge q) \vee (p \wedge r)]$ 21. **Leyes de idempotencia** a) $(p \vee p) \equiv p$ b) $(p \wedge p) \equiv p$ 22. **Leyes de De Morgan** a) $(p \vee q)' \equiv (p' \wedge q')$ b) $(p \wedge q)' \equiv (p' \vee q')$ 23. **Contrapositiva** a) $(p \rightarrow q) \equiv (q' \rightarrow p')$ 24. **Variantes de la condicional** a) $(p \rightarrow q) \equiv (p' \vee q)$ b) $(p \rightarrow q) \equiv (p \wedge q')'$ c) $(p \vee q) \equiv (p' \rightarrow q)$ d) $(p \wedge q) \equiv (p \rightarrow q')'$ e) $[(p \rightarrow r) \wedge (q \rightarrow r)] \equiv [(p \wedge q) \rightarrow r]$ f) $[(p \rightarrow q) \wedge (p \rightarrow r)] \equiv [p \rightarrow (q \wedge r)]$ 25. **Variantes de la bicondicional** a) $(p \leftrightarrow q) \equiv [(p \rightarrow q) \wedge (q \rightarrow p)]$ b) $(p \leftrightarrow q) \equiv [(p' \vee q) \wedge (q' \vee p)]$ c) $(p \leftrightarrow q) \equiv [(p \wedge q) \vee (p' \wedge q')]$ 26. **Contradicción** a) $(p \wedge p') \equiv 0$ 27. **Ley de identidad** a) $(p \vee 0) \equiv p$ b) $(p \vee 1) \equiv 1$ c) $(p \wedge 0) \equiv 0$ d) $(p \vee p') \equiv 1$ e) $(p \wedge 1) \equiv p$ f) $(p \wedge q \vee q) \equiv q$ 28. **Disyunción exclusiva.** a) $(p \oplus q) \equiv (p \leftrightarrow q)'$

*De acuerdo con la información de esta tabla, en las demostraciones se cita el número y el inciso de cada regla utilizada, por ejemplo si de la regla 25 se aplica el inciso c se indicará 25c.

4.6 Argumentos válidos y no válidos

Un argumento consiste en una o más hipótesis y una conclusión, de forma que la conclusión se apoye en las hipótesis. También se puede considerar a un argumento como una serie de proposiciones interrelacionadas que conforman una proposición más compleja, a la cual se le llama *teorema*. Todos los argumentos necesitan de una o más proposiciones iniciales, y a estas proposiciones iniciales se les llama *hipótesis*. La conclusión de un argumento o teorema es una consecuencia de las hipótesis, por esa razón se requiere que las hipótesis sean convincentes y explícitas.

En general los argumentos lógicos a tratar tienen la siguiente forma:

$$P \Rightarrow Q$$

La proposición **P** está integrada por proposiciones más simples llamadas *hipótesis*, las cuales se encuentran relacionadas por el operador lógico $\wedge$, y **Q** es la conclusión del teorema que también puede estar conformada por una o más proposiciones simples, de tal manera que el argumento puede tener la siguiente forma:

$$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \Rightarrow q$$

en donde p_1, p_2,..., p_n son las hipótesis y **q** es la conclusión del razonamiento.

La validez del argumento depende de la estructura existente entre las hipótesis y la conclusión, ya sea por la forma de conectar las hipótesis con la conclusión o por la veracidad de la conclusión misma. La validez es una propiedad de los argumentos. Un argumento puede tener otras propiedades como claro, confuso, endeble, convincente, grande, pequeño, feo o bonito y sin embargo puede no ser válido.

Hay argumentos que son válidos, mientras que otros no lo son. A continuación se ilustra esto en los siguientes ejemplos.

Ejemplo 4.15. Caso en el que el argumento es válido, y tanto las hipótesis como la conclusión son verdaderas.

Considérese lo siguiente:

"Las aves son ovíparas. El gorrión es ave. Por lo tanto; el gorrión es ovíparo."

Considerar que:

$\mathbf{p_1}$: Las aves son ovíparas.
$\mathbf{p_2}$: El gorrión es ave.
$\mathbf{q}$: El gorrión es ovíparo.

Toda la información que se encuentra antes del término "Por lo tanto" conforma las hipótesis. Lo que separa a una hipótesis de otra es el punto y seguido, el cual se representa por una intersección entre cada una de las hipótesis. Por otro lado, la parte que está entre la palabra "Por lo tanto" y el punto final del enunciado, es lo que se conoce como *conclusión*. Dicha conclusión puede estar integrada también por más de una proposición. De esta forma el enunciado anterior se puede representar con notación lógica de la siguiente manera:

$$\mathbf{p_1 \wedge p_2 \Rightarrow q}$$

Como tanto hipótesis como conclusión son verdaderas ($\mathbf{p_1 = 1, p_2 = 1, q = 1}$), entonces se trata de un argumento válido ya que:

$$\mathbf{1 \wedge 1 \Rightarrow 1}$$
$$\mathbf{1 \wedge 1 \rightarrow 1}$$
$$\mathbf{1 \rightarrow 1}$$
$$\mathbf{1}$$

Ejemplo 4.16. En este caso se muestra que un argumento también es válido cuando todas o alguna de las hipótesis es falsa, y la conclusión es verdadera.

Considérese lo siguiente:

"Las mujeres son jóvenes. Miss universo es mujer.
En conclusión, miss universo es joven."

A partir de esto se definen:

$\mathbf{p_1}$: Las mujeres son jóvenes.
$\mathbf{p_2}$: Miss universo es mujer.
$\mathbf{q}$: Miss universo es joven.

En el enunciado anterior la característica de joven es difícil de evaluar, ya que depende con quién se compare, pero suponiendo que una mujer es

joven si tiene entre 17 y 30 años entonces se puede decir que $\mathbf{p_1}$ es "falsa", porque hay mujeres que no son jóvenes; $\mathbf{p_2}$ es "verdadera" y $\mathbf{q}$ es "verdadera". Aunque se tienen hipótesis falsas (con una que sea falsa es suficiente) y la conclusión verdadera, entonces el argumento es completamente válido. Considerando $\mathbf{p_1 = 0}$, $\mathbf{p_2 = 1}$ y $\mathbf{q = 0}$ se tiene:

$$\mathbf{0 \wedge 1 \Rightarrow 1}$$
$$\mathbf{0 \rightarrow 1}$$
$$\mathbf{1}$$

Hay que observar que para evaluar la validez de un argumento, se toma como base la proposición condicional.

Ejemplo 4.17. Caso en el que el argumento es válido, y las hipótesis y la conclusión son falsas.

Considérese lo siguiente:

"Los alemanes son de raza negra. George Bush es de raza negra. Por lo tanto; George Bush es alemán."

A partir de esto se definen:

$\mathbf{p_1}$: Los alemanes son de raza negra.
$\mathbf{p_2}$: George Bush es de raza negra.
$\mathbf{q}$: George Bush es alemán.

En este caso las hipótesis $\mathbf{p_1}$, $\mathbf{p_2}$, y la conclusión $\mathbf{q}$ son falsas, sin embargo el argumento se considera válido.

Utilizando notación lógica, el argumento anterior se puede evaluar de la siguiente forma:

$$\mathbf{0 \wedge 0 \Rightarrow 0}$$
$$\mathbf{0 \rightarrow 0}$$
$$\mathbf{1}$$

Ejemplo 4.18. Un argumento no se considera válido, si está integrado por hipótesis verdaderas y conclusión falsa.

Considérese lo siguiente:

"$c^2 = a^2 + b^2 \cdot c^2 = a^2 + b^2$ se aplica a triángulos rectángulos. Por lo tanto; es la segunda ley de Newton".

Sean:

$\mathbf{p_1}$: $c^2 = a^2 + b^2$.
$\mathbf{p_2}$: $c^2 = a^2 + b^2$ se aplica a triángulos rectángulos.
$\mathbf{q}$: Es la segunda ley de Newton.

En este caso $\mathbf{p_1}$ y $\mathbf{p_2}$ son "verdaderas" porque $c^2 = a^2 + b^2$ es aplicable a triángulos rectángulos, ya que se trata del teorema de Pitágoras. Sin embargo, al tener una conclusión falsa se dice que el argumento es inválido.

En términos de notación lógica y sustituyendo valores se tiene que:

$$\mathbf{p_1 \wedge p_2 \Rightarrow q}$$
$$\mathbf{1 \wedge 1 \rightarrow 0}$$
$$\mathbf{1 \rightarrow 0}$$
$$\mathbf{0}$$

Cuando los argumentos se expresan en nuestro propio lenguaje se debe de tomar en cuenta el contexto, ya que se dan muchos supuestos, como sucede en el ejemplo 4.17 en donde una de las proposiciones es "George W. Bush es de raza negra" y se refiere al presidente de Estados Unidos que se supone que es conocido y que por lo mismo se sabe que no es de raza negra, sin embargo podría haber alguien que se llame igual y sí sea de color, de forma que algo que se considera falso resulta ser verdadero para algunos. Esto es parte de los riesgos que se corren cuando se trata la argumentación lógica.

Cuando no se sabe si las proposiciones que integran un argumento son falsas o verdaderas, es necesario probarlo en todos los casos posibles, teniendo en cuenta que un argumento no es válido solamente cuando a partir de hipótesis verdaderas se desprende una conclusión falsa, esto es, cuando $1 \rightarrow 0$.

Ejemplo 4.19. Considérese el siguiente argumento:

$$(q \wedge p') \wedge (r \rightarrow q') \Rightarrow r'$$

En este caso no se sabe si **p**, **q** o **r** son verdaderas o falsas ya que no representan una proposición conocida y a la cual se le pueda asignar un valor con exactitud, sin embargo resulta que este argumento es válido, ya que si se elabora la tabla de verdad, para todos los valores posibles que pueden tomar **p**, **q** y **r**, se encontrará que en todos los casos el argumento es verdadero; esto es, se trata de una tautología y por lo tanto es argumento válido.

Ejemplo 4.20. Considérese el siguiente argumento:

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

En este caso se trata de un argumento no válido, ya que cuando $p = 1$, $q = 1$ y $r = 0$ se tiene que el argumento es falso.

La forma más fácil de determinar si un argumento es válido o no, cuando no se tienen los valores de las proposiciones, es por medio de la tabla de verdad. Si se trata de una tautología se dice que el argumento es válido, en caso contrario el argumento es inválido.

4.6.1 Tipos de argumentos

Básicamente existen dos tipos de argumentos lógicos: deductivos e inductivos.

- En un argumento deductivo se va de lo general a lo particular, se trata de un procedimiento que parte de un teorema que está formado por hipótesis y una conclusión. Se puede decir que se inicia con una explicación razonable para describir el comportamiento de un conjunto de datos, y que esa explicación se representa por medio de un teorema que deberá demostrarse formalmente por medio de leyes y reglas conocidas (tautologías, reglas de inferencia y equivalencias lógicas en el caso de lógica matemática). El argumento podrá ser válido o inválido. Un argumento deductivo válido se define como aquel que siendo sus hipótesis ciertas, la conclusión también lo es.

- En un argumento inductivo se va de lo particular a lo general, se puede decir que es el conjunto de observaciones y datos cuya tendencia permite visualizar o generalizar el comportamiento de un evento. La veracidad de sus conclusiones se va reforzando con la generación de más y más datos que apuntan en una misma dirección.

En la práctica existen formas de argumentación que no cumplen con los requisitos de los argumentos deductivos o inductivos, sin embargo en este libro solamente se tratará la demostración formal para argumentos deductivos e inductivos debido a que son considerados como los más rigurosos y confiables.

4.7 Demostración formal

Generalmente los argumentos lógicos son razonamientos resultantes del enunciado de un problema que es posible representar, usando notación lógica, como una proposición condicional integrada por varias proposiciones simples, siempre y cuando se identifiquen claramente las proposiciones simples y los conectores lógicos que unen dichas proposiciones. Como se planteó anteriormente, por lo general a la proposición condicional que resulta del planteamiento de un problema se le llama *argumento* o *teorema* y tiene la forma $\mathbf{P} \Rightarrow \mathbf{Q}$, en donde $\mathbf{P}$ y $\mathbf{Q}$ son proposiciones compuestas. A las proposiciones que integran a $\mathbf{P}$ y que están conectadas por operadores $\wedge$ se les llama *hipótesis* y a la proposición $\mathbf{Q}$ se le llama *conclusión*.

Los teoremas representados con notación lógica, producto de un razonamiento, se pueden demostrar usando el "Método directo" o bien el "Método por contradicción" (que son métodos de demostración deductivos). Dependiendo de la naturaleza del teorema, algunas veces es más sencilla la demostración por el método directo y algunas veces es más fácil si se utiliza el método por contradicción.

4.7.1 Demostración por el método directo

Supóngase que $\mathbf{P} \Rightarrow \mathbf{Q}$ es el teorema resultante del planteamiento de un problema usando para ello notación lógica, y que $\mathbf{P}$ y $\mathbf{Q}$ son proposiciones compuestas en las que interviene cualquier número de proposiciones simples que conforman una serie de hipótesis consideradas verdaderas. Se dice que $\mathbf{Q}$ se desprende lógicamente de $\mathbf{P}$, y que por lo tanto el teorema $\mathbf{P} \Rightarrow \mathbf{Q}$ es verdadero. Sin embargo también $\mathbf{P} \Rightarrow \mathbf{Q}$ puede ser falso, si se presenta alguna inconsistencia en la demostración o planteamiento inicial.

Si

$$\mathbf{P} \equiv (p_1 \wedge p_2 \wedge \ldots \wedge p_n)$$

$$Q \equiv q$$

entonces el teorema por demostrar toma la forma

$$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \Rightarrow q$$

en donde $p_1, p_2, \ldots, p_n$ son hipótesis que se consideran verdaderas, ya que son parte del planteamiento del problema, y q es la conclusión a la cual se debe llegar para demostrar la validez del teorema, usando para ello reglas de inferencia, tautologías, equivalencias lógicas y las propias hipótesis del problema. En la demostración se deben de colocar primero las hipótesis, seguidas de las proposiciones obtenidas al aplicar reglas de inferencia, tautologías y equivalencias lógicas, hasta llegar a la conclusión. Todas las líneas de la demostración se deben de numerar, con el fin de evitar confusiones en la obtención de nuevas proposiciones que se deben considerar verdaderas.

En general, las demostraciones formales deben de tener el siguiente formato:

1.–	p_1
2.–	p_2
	.
	.
	.
n.–	p_n
(n+1).–	p_{n+1}
	.
	.
	.
(m−1).–	p_{m-1}
m.–	q

Las líneas 1 a n son las hipótesis resultantes del enunciado a demostrar, y siempre se colocan al principio de la demostración. Las líneas (n + 1) a (m – 1) son proposiciones obtenidas usando reglas de inferencia, tautologías o equivalencias lógicas, y finalmente la línea m es la conclusión q obtenida.

Se puede decir que la demostración de un teorema dependerá de la lógica empleada por cada persona para relacionar la información que ya conoce por medio de reglas de inferencia, tautologías o equivalencias lógicas

hasta llegar a la conclusión, y que el camino no es único. Algunas personas demostrarán el teorema por un camino corto y otras llegarán a la solución por una ruta más larga, porque la vinculación lógica de información es diferente en cada caso. Realmente la demostración de un teorema es equivalente a resolver un problema de la vida real, y como en ésta cada persona puede tener un procedimiento diferente para llegar a los mismos resultados siendo algunos mejores que otros porque dependen del manejo lógico de la información, de las herramientas utilizadas y de la experiencia del propio sujeto.

No todas las personas logran resolver un problema determinado, sobre todo si nunca antes se han enfrentado a ese tipo de problema. Sucede lo mismo en lógica matemática: no todas las personas llegan a demostrar un teorema dado, ya que esto requiere de un razonamiento lógico para vincular la información. También es importante mencionar que no todos los problemas se pueden resolver de la misma manera, además de que no todos los teoremas son verdaderos, en cuyo caso es necesario demostrar que son falsos, lo cual se analizará más adelante.

Si está bien planteado el problema, el número de hipótesis (1 a la n) no cambia, sin embargo el número de proposiciones obtenidas entre (n + 1) y (m – 1) varía dependiendo de las reglas de inferencia, tautologías o equivalencias lógicas que cada persona utilice para llegar a la conclusión.

En el ejemplo 4.21 se demuestra un enunciado y se explica el uso de las herramientas lógicas.

Ejemplo 4.21. Sean las siguientes proposiciones:

p: Trabajo.
q: Ahorro.
r: Compraré una casa.
s: Podré guardar el coche en mi casa.

A partir de esta información represéntese el siguiente enunciado en forma de teorema usando notación lógica, y llévese a cabo la demostración formal aplicando el método directo.

"Si trabajo o ahorro, entonces compraré una casa. Si compro una casa, entonces podré guardar el coche en mi casa. Por consiguiente, si no puedo guardar el coche en mi casa, entonces no ahorro."

En el enunciado anterior, cada párrafo separado por punto y seguido es una hipótesis hasta llegar a una frase como "Por consiguiente", "Por lo

tanto" o "En conclusión", ya que después de esa frase toda la información formará parte de la conclusión.

Como el planteamiento se debe representar en la forma $P \Rightarrow Q$, la información que pertenece a cada elemento es como se muestra a continuación.

P	⇒	Q
Si trabajo o ahorro, entonces compraré una casa. Si compro una casa, entonces podré guardar el coche en mi casa.	**Por consiguiente**	**si no puedo guardar el coche en mi casa, entonces no ahorro**

Como se ve, **P** puede estar integrada por varias hipótesis, cada una de ellas separada por un punto y seguido, y para completar el teorema es necesaria su conclusión correspondiente **Q**.

P			⇒	Q
Si trabajo o ahorro, entonces compraré una casa	**.**	**Si compro una casa, entonces podré guardar el coche en mi casa**	**Por consiguiente**	**si no puedo guardar el coche en mi casa, entonces no ahorro**
$(p \vee q) \rightarrow r$	$\wedge$	$r \rightarrow s$	$\Rightarrow$	$s' \rightarrow q'$

En el cuadro anterior, **P** está integrada por dos hipótesis

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

Mientras que **Q** sólo es la proposición condicional

$$s' \rightarrow q'$$

Tomando en cuenta esto, el enunciado en forma de teorema es el siguiente:

$$[(p \vee q) \rightarrow r] \wedge [r \rightarrow s] \Rightarrow [s' \rightarrow q']$$

La forma general de este enunciado es:

$$(p_1 \wedge p_2 \wedge \ldots \wedge p_n) \Rightarrow q$$

A continuación se demuestra el teorema, señalando en cada paso la tautología, regla de inferencia o equivalencia lógica que se usa en la demostración.

1.	$(p \vee q) \rightarrow r$	Hipótesis
2.	$r \rightarrow s$	Hipótesis
3.	$q \rightarrow (q \vee p)$	Adición; 1
4.	$q \rightarrow (p \vee q)$	3; ley conmutativa; 18a
5.	$q \rightarrow r$	4, 1; silogismo hipotético; 13
6.	$q \rightarrow s$	5, 2; silogismo hipotético; 13
7.	$s' \rightarrow q'$	6; contrapositiva; 23

Como se ve, lo primero que se coloca en la demostración son las hipótesis, ya que es información conocida del problema. La línea 3 es la tautología 1, $[p \Rightarrow (p \vee q)]$, que en este caso no se aplicó a ninguna línea sino que se extrajo directamente de la lista de tautologías, se cambió $\Rightarrow$ por $\rightarrow$ y se cambió la letra **p** por **q** y **q** por **p**, por conveniencia. Para obtener la línea 4, se aplicó a la información que se encuentra en la línea 3 la equivalencia lógica (18a). En la línea 5 se utilizó la información de las líneas 4 y 1 y se aplicó la regla de inferencia (13). En la línea 6 también se usó el silogismo hipotético, pero ahora fue aplicado a la información de las líneas 5 y 2. Finalmente se aplicó a la información de la línea 6 la equivalencia lógica (23) para obtener la conclusión.

Como se mencionó, el procedimiento para demostrar un teorema no es único sino que depende de cada persona. A continuación se presenta otra forma de demostrar el mismo teorema.

1.	$(p \vee q) \rightarrow r$	Hipótesis
2.	$r \rightarrow s$	Hipótesis
3.	$(p \vee q) \rightarrow s$	1, 2; Silogismo hipotético; 13
4.	$s' \rightarrow (p \vee q)'$	3; Contrapositiva; 23
5.	$s' \rightarrow (p' \wedge q')$	4; Ley de De Morgan; 22a
6.	$(q' \wedge p') \rightarrow q'$	Simplificación; 2
7.	$(p' \wedge q') \rightarrow q'$	6; Ley conmutativa; 18b
8.	$s' \rightarrow q'$	5, 7; Silogismo hipotético; 13

Obsérvese cómo las equivalencias lógicas, como es el caso de la ley de De Morgan para obtener la línea 5, se pueden aplicar a toda la línea o parte de ella. Sin embargo, las reglas de inferencia requieren de una o más líneas completas con el formato de la regla, para poderse aplicar, como es el caso del silogismo hipotético para obtener la línea 3, que requiere de la información que se encuentra en las líneas 1 y 2.

Es probable que las tautologías causen un poco de confusión en relación con la forma en que se aplican en las demostraciones, ya que se puede tener parte de la tautología en una línea y colocar el resto en otra, como se muestra a continuación.

Supóngase que en una demostración se tienen las siguientes líneas:

5.	$(p \rightarrow q')$	.....
6.	.....	.....
7.	$(p \rightarrow q') \vee s$	5; Adición; 1

Realmente la tautología que se está aplicando a la línea 5 es la adición $p \Rightarrow (p \vee q)$, porque teniendo p, que en este caso $p \equiv (p \rightarrow q')$, se puede obtener $(p \vee q)$, que en este caso $(p \vee q) \equiv [(p \rightarrow q') \vee s]$.

La mayoría de estas reglas vienen en dos presentaciones, una como tautología y otra como regla de inferencia, de tal forma que en lugar de indicar que se aplicó la adición 1 en la línea 7, se pudo haber indicado que se aplicó la regla de inferencia 10, que también es una adición.

Las tautologías no necesariamente se tienen que aplicar a una línea, sino que se pueden extraer de la lista y colocarse en la demostración como se muestra a continuación:

5.	.....	.....
6.	$[(r' \vee q) \rightarrow p'] \wedge (s' \rightarrow q)] \Rightarrow [(r' \vee q) \rightarrow p']$	Simplificación; 2
7.	.....	.....

En este caso se extrajo la tautología $(p \wedge q) \Rightarrow p$ de la lista y se colocó en la demostración, por lo tanto no es necesario que se indique a qué línea se aplicó, sólo se requiere indicar qué tautología es y qué número tiene. Obsérvese cómo lo único que se debe guardar es la forma, ya que para aplicar la regla se consideró que $p \equiv [(r' \vee q) \rightarrow p']$ y que $q \equiv (s' \rightarrow q)$.

4.7.2 Demostración por contradicción

El procedimiento de la demostración por contradicción es semejante al del método directo, con la diferencia de que las líneas iniciales de dicha demostración no son únicamente las hipótesis, sino que además se incluye una línea con la negación de la conclusión. Se debe de tener presente que el objetivo de la demostración es llegar a una contradicción de la forma $(p \wedge p') \equiv 0$.

Ejemplo 4.22. La demostración por contradicción del teorema

$$[(p \vee q) \rightarrow r] \wedge [r \rightarrow s] \Rightarrow [s' \rightarrow q']$$

es la siguiente:

1. $(p \vee q) \rightarrow r$	Hipótesis
2. $r \rightarrow s$	Hipótesis
3. $(s' \rightarrow q')'$	Negación de la conclusión
4. $[(s' \wedge q'')']'$	3; Variantes de la condicional 24b
5. $s' \wedge q$	4; Doble negación; 17
6. s'	5; Simplificación; 11
7. q	5; Simplificación; 11
8. $(p \vee q) \rightarrow s$	1, 2; Silogismo hipotético; 13
9. $s' \rightarrow (p \vee q)'$	8; Contrapositiva; 23
10. $s' \rightarrow (p' \wedge q')$	9; Ley de De Morgan; 22a
11. $(p' \wedge q')$	6, 10; Modus ponens; 15
12. q'	11; Simplificación; 11
13. $q \wedge q'$	7, 12; Conjunción; 14
14. 0	13; Contradicción; 26

El llegar a un valor de cero significa que el teorema es falso, pero como se consideró como verdadera la negación de la conclusión y se colocó en la demostración, realmente lo que se está demostrando es que el teorema $[(p \vee q) \rightarrow r] \wedge [r \rightarrow s] \Rightarrow [s' \rightarrow q']$ es verdadero.

En este caso el procedimiento por contradicción resultó más complejo, pero no siempre es así ya que existen teoremas que son más fáciles de demostrar por contradicción.

En la demostración por contradicción del ejemplo 4.22 no era necesario llegar a la contradicción con la proposición q como de hecho ocurrió, $q \wedge q'$, sino que se podría haber llegado a la contradicción de p, r o s. Cualquiera de ellas es válida para la demostración del teorema.

Ejemplo 4.23. Representar el siguiente enunciado en forma de teorema usando notación lógica, y hacer la demostración formal mediante el método directo y por contradicción.

"Si no le acelero al automóvil, entonces el automóvil no correrá. Si no le freno al automóvil, entonces el automóvil no se detendrá. Si el automóvil no corre o no se detiene, entonces el automóvil está fallando. De tal manera que si el automóvil no está fallando, entonces puedo acelerar y frenar el automóvil."

Sean las siguientes proposiciones:

p: Le acelero al automóvil.
q: El automóvil corre.
r: Le freno al automóvil.
s: El automóvil se detiene.
t: El automóvil falla.

A partir de estas proposiciones y del enunciado dado se obtienen las hipótesis y la conclusión siguientes:

$p' \rightarrow q'$	Hipótesis
$r' \rightarrow s'$	Hipótesis
$(q' \vee s') \rightarrow t$	Hipótesis
$t' \rightarrow (p \wedge r)$	Conclusión

Entonces el teorema por demostrar queda integrado de la siguiente forma:

$$[p' \rightarrow q'] \wedge [r' \rightarrow s'] \wedge [(q' \vee s') \Rightarrow [t' \rightarrow (p \wedge r)]]$$

Demostración del teorema mediante el método directo:

1.	$p' \rightarrow q'$	Hipótesis
2.	$r' \rightarrow s'$	Hipótesis
3.	$(q' \vee s') \rightarrow t$	Hipótesis
4.	$[p' \rightarrow q'] \wedge [r' \rightarrow s']$	1, 2; Conjunción; 14
5.	$(p' \vee r') \rightarrow (q' \vee s')$	4; Dilema constructivo; 9a
6.	$(p' \vee r') \rightarrow t$	5, 3; Silogismo hipotético; 13
7.	$t' \rightarrow (p' \vee r')'$	6; Contrapositiva; 23
8.	$t' \rightarrow (p \wedge r)$	7; Ley de De Morgan; 22a

Demostración del teorema por contradicción:

1. $p' \rightarrow q'$	Hipótesis
2. $r' \rightarrow s'$	Hipótesis
3. $(q' \vee s') \rightarrow t$	Hipótesis
4. $[t' \rightarrow (p \wedge r)]'$	Negación de la conclusión
5. $[t \vee (p \wedge r)]'$	4; Variantes de la condicional; 24a
6. $t' \wedge (p \wedge r)'$	5; Ley de Morgan; 22a
7. t'	6; Simplificación; 11
8. $(p \wedge r)'$	6; Simplificación; 11
9. $[p' \rightarrow q'] \wedge [r' \rightarrow s']$	1, 2; Conjunción; 14
10. $(p' \vee r') \rightarrow (q' \vee s')$	9; Dilema constructivo; 9a
11. $(p \wedge r)' \rightarrow (q' \vee s')$	10; Ley de De Morgan; 22b
12. $(p \wedge r)' \rightarrow t$	11, 3; Silogismo hipotético; 13
13. t	8, 12; Modus ponens; 15
14. $t' \wedge t$	7, 13; Conjunción; 14
15. 0	14; Contradicción; 26

Es recomendable que las proposiciones que integran la contradicción sean simples, como se muestra en la línea 14. Por lo general, una de ellas se obtiene al relacionar la negación de la conclusión con las demás proposiciones, y la otra resulta de la vinculación de las hipótesis resultantes del planteamiento.

Por último, hay que tener presente que no existe una forma única de hacer una demostración, ya que siempre y cuando no se violen las reglas, cada persona puede usar un procedimiento diferente.

4.8 Predicados y sus valores de verdad

La lógica de proposiciones es muy buena para inferir información cuando es posible determinar claramente si una proposición es falsa o verdadera, pero en la vida real prácticamente nada es totalmente falso o totalmente verdadero, ya que influyen muchos factores. El problema de la lógica de proposiciones es que no puede trabajar con proposiciones en donde una gran cantidad de elementos cumplen con ciertas características y otros no

Sea la proposición:

p: La puerta es verde.

¿Qué pasa si la puerta es verde a medias, es decir, si tiene espacios sin pintar? A pesar de esto, en la lógica proposicional se tiene que especificar si **p** es falsa o verdadera.

La **lógica de predicados**, o lógica de conjuntos, se basa en que las proposiciones son conjuntos de elementos que tienen una propiedad o característica llamada "predicado", y en este contexto una proposición puede ser verdadera para un grupo de elementos de un conjunto, pero falsa para otro.

Con el fin de ilustrar los conceptos, considérese el siguiente ejemplo:

Sean:

U = {**x** | **x** es un habitante del continente africano}
p: "Hablan francés"

A partir de esto se tiene que

p(x): "**x** habla francés"

o bien

p(x): "Todos los africanos hablan francés"
$\forall$**x p(x)**: "Todos los africanos hablan francés"
$\exists$**x p(x)**: "Algún o algunos africanos hablan francés"

En la lógica de predicados se debe definir un conjunto universo, dominio o universo del discurso, que contiene a todos los elementos a los cuales se está sometiendo al predicado. En el ejemplo anterior el dominio es **U** y el predicado es **p**. Además se cuenta con los conceptos "Todos" y "Algunos", que permiten manejar más de un elemento de un conjunto y cuya representación en matemáticas es:

$\forall$ = "Para todo o todos"
$\exists$ = "Existe alguno, algunos o al menos un elemento"

Obviamente la proposición **p(x)** del ejemplo anterior es falsa, porque si bien es cierto que muchos africanos hablan francés, también hay buena parte de los africanos que no hablan ese idioma, como por ejemplo la mayoría de los sudafricanos.

Hay que observar también que

$$\forall \mathbf{x}\ \mathbf{p(x)} \equiv \mathbf{p(x)}$$

De tal manera que si no se le antepone al predicado el cuantificador universal $\forall$, es como si lo tuviera.

Por otro lado, se tiene que

$\exists \mathbf{x}\ \mathbf{p(x)}$: "Algún africano habla francés"

Es verdadera, ya que efectivamente algunos africanos tienen como idioma oficial el francés o hablan el idioma aunque no sea el oficial. Obsérvese que no se especifica cuántos de ellos hablan francés, sólo se plantea si la proposición es falsa o verdadera. Es obvio que $\exists \mathbf{x}\ \mathbf{p(x)} \neq \mathbf{p(x)}$.

En general se acostumbra indicar junto con el predicado cuál es el dominio para esa proposición, de forma que los enunciados anteriores pueden plantearse de la siguiente manera:

$\forall \mathbf{x}\ \mathbf{p(x)} \qquad \mathbf{x} \in \mathbf{U}$

(Para todo **x**; tal que **p**, donde **x** es un elemento de **U**)

$\exists \mathbf{x}\ \mathbf{p(x)} \qquad \mathbf{x} \in \mathbf{U}$

(Existe algún elemento **x**; tal que **p**, donde **x** es elemento de **U**)

Como se puede observar, el concepto de conjunto es muy importante en lógica de predicados, por lo que es conveniente tener en cuenta la definición de conjunto, sus propiedades y algunos conjuntos que se utilizan con más frecuencia en matemáticas.

Es importante mencionar que los operadores lógicos $\{\vee, \wedge, ', \rightarrow, \leftrightarrow\}$ que se usan en lógica de proposiciones, son también válidos en lógica de predicados.

Ejemplo 4.24. Sean:

$\mathbf{U} = \{\mathbf{z} \mid \mathbf{z}$ es una persona$\}$

$\mathbf{A} = \{\mathbf{x} \mid \mathbf{x}$ es un artista$\}$

$\mathbf{B} = \{\mathbf{y} \mid \mathbf{y}$ es un político$\}$

$\mathbf{A} \subseteq \mathbf{U}$ y $\mathbf{B} \subseteq \mathbf{U}$

p: Son ricos

q: Son corruptos

r: Son ricos y corruptos

A partir de aquí se tiene que:

$\forall \mathbf{z}\ \mathbf{p(z)}$: Todas las personas son ricas
$\forall \mathbf{x}\ \mathbf{p(x)}$: Todos los artistas son ricos
$\forall \mathbf{y}\ \mathbf{p(y)}$: Todos los políticos son ricos
$\forall \mathbf{x}\ \mathbf{q(x)}$: Todos los artistas son corruptos
$\forall \mathbf{y}\ \mathbf{q(y)}$: Todos los políticos son corruptos

$\exists \mathbf{z}\ \mathbf{q(z)}$: Algunas personas son corruptas
$\exists \mathbf{x}\ \mathbf{p(x)}$: Algunos artistas son ricos
$\exists \mathbf{y}\ \mathbf{p(y)}$: Algunos políticos son ricos
$\exists \mathbf{x}\ \mathbf{q(x)}$: Algunos artistas son corruptos
$\exists \mathbf{y}\ \mathbf{q(y)}$: Algunos políticos son corruptos

$\forall \mathbf{z}\ \mathbf{r(z)}$: Todas las personas son ricas y corruptas
$\forall \mathbf{x}\ \mathbf{r(x)}$: Todos los artistas son ricos y corruptos
$\exists \mathbf{y}\ \mathbf{r(y)}$: Algunos políticos son ricos y corruptos

$\forall \mathbf{x}\ \forall \mathbf{y}\ \mathbf{r(x, y)}$: Todos los artistas y todos los políticos son ricos y corruptos

$\exists \mathbf{x}\ \exists \mathbf{y}\ \mathbf{r(x, y)}$: Algunos artistas y algunos políticos son ricos y corruptos

$\exists \mathbf{x}\ \forall \mathbf{y}\ \mathbf{r(x, y)}$: Algunos artistas y todos los políticos son ricos y corruptos

Se puede observar que en este caso se tiene que:

$$\exists \mathbf{x}\ \forall \mathbf{y}\ \mathbf{r(x, y)} \equiv \exists \mathbf{x}\ \mathbf{p(x)} \wedge \forall \mathbf{y}\ \mathbf{q(y)} \qquad \mathbf{x, y} \in \mathbf{U}$$

El complemento de un enunciado se indica de la siguiente manera:

$[\forall \mathbf{x}\ \mathbf{p(x)}]' \equiv \forall \mathbf{x}\ \mathbf{p'(x)}$: Ningún artista es rico

ya que el complemento de todos es "ninguno". Sin embargo, el complemento de algunos son los elementos que faltan para completar "todos":

$[\exists \mathbf{x}\ \mathbf{p(x)}]' \equiv \exists \mathbf{x}\ \mathbf{p'(x)}$: Algunos artistas no son ricos
$\forall \mathbf{x}\ \exists \mathbf{y}\ \mathbf{r'(x, y)}$: Ningún artista es rico ni corrupto, y algunos políticos no son ricos ni corruptos

Entonces el enunciado:

> "Todos los artistas son ricos. Algunos políticos son corruptos. En conclusión no todos los artistas y no todos los políticos son ricos y corruptos."

se puede representar como

$$[\forall x\, p(x) \wedge \exists y\, q(y)] \Rightarrow [\exists x\, \exists y\, r'(x, y)] \qquad x \in A;\ y \in B$$

También puede expresarse sacando del corchete los cuantificadores $\forall$ y $\exists$:

$$\forall x\, \exists y\, [p(x) \wedge q(y)] \Rightarrow \exists x\, \exists y\, [r'(x, y)] \qquad x \in A;\ y \in B$$

o bien quitando el cuantificador universal $\forall$ y dejando solamente el existencial $\exists$:

$$\exists y\, [p(x) \wedge q(y)] \Rightarrow \exists x\, \exists y\, [r'(x, y)] \qquad x \in A;\ y \in B$$

Como por lo general no se usan corchetes, también queda perfectamente expresado de la siguiente forma:

$$p(x) \wedge \exists y\, q(y) \Rightarrow \exists x\, \exists y\, r'(x, y) \qquad x \in A;\ y \in B$$

Finalmente el enunciado se evalúa de la misma manera que como se hace en lógica proposicional:

$p(x)$	Es "falsa" (0) ya que no todos los artistas son ricos.
$\exists y\, q(y)$	Es "verdadera" (1) ya que algunos políticos son corruptos.
$\exists x\, \exists y\, r'(x, y)$	Es "verdadera" (1) ya que algunos artistas y algunos políticos no son ricos ni corruptos.

En el momento de evaluar la proposición es posible cambiar $\Rightarrow$ por $\rightarrow$, de forma que el resultado del predicado $p(x) \wedge \exists y\, q(y) \Rightarrow r'(x, y)$ se obtiene sustituyendo valores:

$$(0 \wedge 1) \rightarrow 1 \qquad \text{Es "verdadera"}$$

Ejemplo 4.25. Sea el enunciado:

> "Algunas elecciones son limpias y no es cierto que todas las elecciones sean dudosas o en algunas de ellas no se cuenta con información."

Considerar que:

U = {**x** | **x** es una elección}
p: Son limpias
q: Son dudosas
r: Se cuenta con la información
p(x): Todas las elecciones son limpias
q(x): Todas las elecciones son dudosas
r(x): De todas las elecciones se cuenta con la información

A partir de esto, el enunciado anterior se puede representar de la siguiente manera:

$$\exists x\, p(x) \wedge \forall x\, q'(x) \vee \exists x\, r'(x) \qquad x \in U$$

Si el valor de verdad de cada una de las proposiciones que integran el predicado es:

$\exists x\, p(x)$	Algunas elecciones son limpias (verdadero)
$\forall x\, q(x) \equiv q(x)$	Todas las elecciones son dudosas (falso)
$q'(x)$	No es cierto que todas las elecciones sean dudosas (verdadero)
$\forall x\, r(x) \equiv r(x)$	Se cuenta con la información de todas las elecciones (falso)
$\exists x\, r'(x)$	De algunas elecciones no se tiene información" (verdadero)

De esta forma el enunciado completo se evalúa como verdadero:

$$\exists x\, p(x) \wedge \forall x\, q'(x) \vee \exists x\, r'(x) \equiv 1 \wedge 0' \vee 1 \equiv 1 \vee 1 \equiv 1$$

El orden en que se colocan los argumentos es importante, ya que al cambiar de posición los argumentos que se encuentran en el paréntesis, el significado no siempre es el mismo.

Ejemplo 4.26. Sean:

A = {**x** | **x** es un ladrón de Madrid}

B = {**y** | **y** es una persona que ha sido asaltada en Madrid}

p: "Asaltaron a"

A partir de aquí se plantea que:

$\forall$**x** $\forall$**y p(x, y)**: Todos los ladrones de Madrid asaltaron a todas las víctimas de asalto en Madrid

$\forall$**y** $\forall$**x p(x, y)**: Todos los ladrones de Madrid asaltaron a todas las víctimas de asalto en Madrid

$\forall$**y** $\forall$**x p(y, x)**: Todas las víctimas de asalto en Madrid asaltaron a todos los ladrones de Madrid

$\forall$**x** $\exists$**y p(x, y)**: Todos los ladrones de Madrid asaltaron a algunas víctimas de asalto en Madrid

$\exists$**y** $\forall$**x p(y, x)**: Algunas víctimas de asalto de Madrid asaltaron a todos los ladrones de Madrid

$\exists$**y** $\exists$**x p(y, x)**: Algunas víctimas de asalto de Madrid asaltaron a algunos ladrones de Madrid

Obsérvese que es muy importante la posición de los parámetros dentro del paréntesis, ya que cuando los conjuntos **A** y **B** no tienen los mismos elementos se puede obtener que:

$\forall$**x** $\forall$**y p(x, y)** $\equiv$ $\forall$**y** $\forall$**x p(x, y)**	(sus parámetros no cambian de posición)
$\forall$**x** $\forall$**y p(x, y)** $\neq$ $\forall$**y** $\forall$**x p(y, x)**	(sus parámetros sí cambian de posición)
$\exists$**x** $\exists$**y p(x, y)** $\equiv$ $\exists$**y** $\exists$**x p(x, y)**	(sus parámetros no cambian de posición)
$\exists$**x** $\exists$**y p(x, y)** $\neq$ $\exists$**y** $\exists$**x p(y, x)**	(sus parámetros sí cambian de posición)
$\exists$**x** $\forall$**y p(x, y)** $\equiv$ $\forall$**y** $\exists$**x p(x, y)**	(sus parámetros no cambian de posición.
$\forall$**x** $\exists$**y p(x, y)** $\neq$ $\exists$**y** $\forall$**x p(y, x)**	(sus parámetros sí cambian de posición)

La posición de los parámetros permite saber el significado correcto de los enunciados y no el orden de los cuantificadores, ya que éstos indican solamente la cantidad de elementos del dominio, que se están sometiendo al predicado.

Ejemplo 4.27. Sean:

$\mathbf{U} = \{x, y \mid x \in Z^+, y \in Z^+\}$

$\mathbf{p}$: $(x - 1) \leq y$

¿Cuál es el significado del predicado de cada uno de los siguientes incisos, así como su valor de verdad?

a) $\forall \mathbf{x}\, \forall \mathbf{y}\, [\mathbf{p(x, y)}] \equiv \forall \mathbf{y}\, \forall \mathbf{x}\, [\mathbf{p(x, y)}]$ $\quad \mathbf{x, y} \in \mathbf{U}$

El significado es: "Para todo entero positivo se cumple que $(x - 1) \leq y$."

En este caso el predicado es "**falso**", ya que existen elementos en donde el predicado no es cierto. Por ejemplo, si $x = 5$ no se cumple cuando $y < 4$, si $x = 6$ no se cumple para $y < 5$.

b) $\forall \mathbf{y}\, \exists \mathbf{x}\, [\mathbf{p(x, y)}]$ $\quad \mathbf{x, y} \in \mathbf{U}$

El significado es: "Para algún entero positivo se cumple que $(x - 1) \leq y$; para todo entero positivo." Obsérvese cómo primero se enuncia el parámetro *x*, ya que está colocado primero en el paréntesis $\mathbf{p(x, y)}$.

El predicado es "**verdadero**", ya que cuando $x = 2$ es verdad para todos los valores de *y* que puede tomar, lo mismo ocurre para $x = 1$. Sin embargo, $x = 3$ no se cumple si $y = 1$. Pero como es suficiente que se cumpla para un valor de x, entonces se dice que es cierta.

c) $\exists \mathbf{y}\, \forall \mathbf{x}\, [\mathbf{p(x, y)}]$ $\quad \mathbf{x, y} \in \mathbf{U}$

Significa que: "Para todo entero positivo se cumple que $(x - 1) \leq y$; para cuando menos un entero positivo."

Es "**verdadera**", ya que dado un valor entero positivo *x* cualquiera, siempre se encontrará cuando menos un valor de *y* que permita que la desigualdad $(x - 1) \leq y$; se cumpla.

d) $\forall \mathbf{x}\, \exists \mathbf{y}\, [\mathbf{p(x, y)}]$ $\quad \mathbf{x, y} \in \mathbf{U}$

Significa que: "Para todo entero positivo se cumple que $(x - 1) \leq y$; para cuando menos un entero positivo."

Se tiene lo mismo que en el inciso (c), ya que $\forall \mathbf{x}\, \exists \mathbf{y}\, [\mathbf{p(x, y)}] \equiv \exists \mathbf{y}\, \forall \mathbf{x}\, [\mathbf{p(x, y)}] \equiv \exists \mathbf{y}\, [\mathbf{p(x, y)}]$ considerando que el cuantificador universal $\forall \mathbf{x}$ se puede eliminar. Por lo tanto, es "**verdadera**".

e) $\exists x\, \forall y\, [p(x, y)]$ $\quad x, y \in U$

Igual que el inciso (b), ya que $\exists x\, \forall y\, [p(x, y)] \equiv \forall y\, \exists x\, [p(x, y)] \equiv \exists x\, [p(x, y)]$, lo cual es "**verdadero**" cuando $x = 1, 2$.

f) $\exists x\, \exists y\, [p(x, y)] \equiv \exists y\, \exists x\, [p(x, y)]$ $\quad x, y \in U$

Significa que: "Existe algún entero positivo que cumple con $(x - 1) \leq y$; para al menos algún entero positivo."

Es "**verdadero**".

Del ejemplo anterior se puede inferir que cuando se trata del mismo cuantificador y el conjunto del discurso es el mismo tanto para *x* como para *y*, no importa el orden en que sean colocados los cuantificadores, ya que el significado es el mismo siempre y cuando no cambien de posición los parámetros dentro del paréntesis.

$$\forall x\, p(x, y) \equiv \forall y\, p(x, y)$$
$$p(x, y) \equiv p(x, y)$$
$$\forall x\, \forall y\, p(x, y) \equiv \forall y\, \forall x\, p(x, y)$$
$$\exists x\, \exists y\, p(x, y) \equiv \exists y\, \exists x\, p(x, y)$$
$$\forall x\, \exists y\, p(x, y) \equiv \exists y\, \forall x\, p(x, y)$$

Sin embargo, se debe tener cuidado cuando se tienen cuantificadores universal y existencial en un mismo predicado, pero donde *x*, *y* no pertenecen al mismo conjunto del discurso, ya que el resultado no necesariamente se conserva.

El número de argumentos de un predicado debe ser constante de tal forma que $p(a, b)$ es diferente de $p(a, b, c)$. Sin embargo $p(x)$ es equivalente a $p(w)$, siempre y cuando x y w pertenezcan al mismo universo del discurso.

No siempre se tienen frases que contengan las palabras "todos" o "algunos", a veces existen enunciados con la palabra "ninguno", de forma que ninguno de los elementos del universo del discurso cumple con la condición.

Ejemplo 4.28. Sean:

U = {x | x es alumno de la materia de matemáticas para computación}

p: Aprobó el examen de matemáticas para computación

p(x): Todos los alumnos de matemáticas para computación aprobaron el examen

El enunciado "Ningún alumno aprobó el examen de matemáticas para computación", se puede representar como:

$$(\forall \mathbf{x}\ \mathbf{p(x)})' \qquad \text{o bien} \qquad \forall \mathbf{x}\ \mathbf{p'(x)}$$

En los predicados pueden existir variables libres y variables ligadas. Las variables ligadas a un cuantificador se consideran locales a ese predicado, mientras que las que no tienen cuantificador se consideran libres. Por ejemplo, en el siguiente predicado:

$$\forall \mathbf{x}\ \mathbf{p(x)} \vee \exists \mathbf{z}[\mathbf{q(y)} \wedge \mathbf{r(z)} \wedge \mathbf{s(w)}]$$

son variables libres "w" y "y". Se consideran variables ligadas a "x" y "z".

4.9 Inducción matemática

Como se mencionó anteriormente, una proposición es una oración, frase, igualdad o desigualdad, que puede ser falsa o verdadera, pero no ambas a la vez. La inducción matemática se utiliza cuando se desea probar si una expresión matemática (igualdad o desigualdad) es falsa o verdadera, sin necesidad de representarla con notación lógica. En computación es común desarrollar programas en donde se tiene un "valor inicial", para la primera iteración, un incremento o decremento que puede ser aplicado por medio de una expresión matemática llamada término "*n*-ésimo", que permite obtener los valores de una sumatoria en cada iteración y un "resultado" de la sumatoria, el cual también es posible representar en forma generalizada por medio de una expresión matemática. Esto implica que es posible representar algoritmos en forma matemática y probar si esos algoritmos son falsos o verdaderos, usando para ello inducción matemática. Para usar la inducción matemática en la demostración de algoritmos es necesario que estos se representen como una sumatoria de la siguiente manera:

$$x_1 + x_2 + x_3 \ldots + t = r$$

Inicio — Término *n*-ésimo — Resultado

En la sumatoria anterior, el primer elemento x_1 es el valor obtenido en la primera iteración ($n = 1$) y se conoce como valor inicial. El término n-ésimo es una expresión matemática que permite encontrar cada uno de los elementos de la sumatoria y que deberá estar en función de n, ya que dependiendo del valor de n se determina si se trata del primero, segundo o n-ésimo elemento. Finalmente, el resultado r también es una expresión matemática en función de n que permite encontrar el resultado de sumar los n elementos de la sumatoria. La sumatoria anterior, incluyendo inicio, término n-ésimo y resultado, es la proposición $P(n)$.

El principio de inducción matemática establece que la proposición $P(n)$ es verdadera $\forall n \geq k$ si se cumplen las siguientes condiciones:

a) $P(k)$ es verdadera cuando $k = 1$.

b) $P(k)$ es cierta cuando $k = n + 1$.

Al primer inciso se le conoce como "paso básico" y al segundo se le llama "paso inductivo".

El método consiste en sustituir $n = 1$ en el n-ésimo término de la sumatoria. Si el resultado obtenido es igual al primer término de la sumatoria, se dice que se cumple el paso básico. En caso de que se cumpla el paso básico, se procede a probar si la proposición también es verdadera cuando $k = n + 1$. Se sustituye $(n + 1)$ en lugar de n en el termino n-ésimo de la sumatoria, se agrega dicho término en los dos lados de la igualdad, para que no se altere, y se realizan algunas operaciones algebraicas hasta obtener una forma tal que sea fácil de sustituir $k = n + 1$. Si el resultado, que ahora está en función de k, tiene la misma forma que la igualdad en función de n, se dice que se cumple el paso inductivo y que, por lo tanto, la proposición $P(n)$ es válida o verdadera. En caso de que no se cumpla el paso básico o inductivo se considera que $P(n)$ es falsa.

Cuenta la historia que cuando el matemático alemán Carl Friedrich Gauss tenía diez años, su maestro necesitaba salir del salón de clase y para dejar entretenidos a los alumnos les pidió que llevaran a cabo la siguiente sumatoria:

$$1 + 2 + 3 + \ldots + 998 + 999 + 1\,000$$

Seguramente el maestro esperaba que los alumnos hicieran 1000 sumas para obtener el resultado, sin embargo se dice que cuando se disponía a salir del salón Gauss le dijo que ya tenía el resultado, lo cual le sorprendió por lo que le pidió que le explicara cómo lo había obtenido. Gauss respondió que si se suma el primero y el último elementos de la sumatoria (1 + 1000) el resultado es 1001, si se suman el segundo y el penúltimo (2 + 999) el resultado es 1001, si se suman el tercero y el antepenúltimo también el resultado es 1001, y si se sigue sumando así hasta llegar a sumar los que se encuentran en la parte media de la sumatoria (500 + 501)

Johann Carl Friedrich Gauss
(1777-1855)

ue un matemático, astrónomo y físico emán de una deslumbrante genialidad, ue realizó contribuciones fundamentas en la teoría de números, el análisis atemático, la geometría diferencial, la eodesia, el magnetismo y la óptica.

Considerado "el prín cipe de las matemáticas" y "el matemático más grande desde la antigüedad", Gauss es considerado uno de los matemáticos que más influencia ha tenido a través de la historia.

el resultado también es 1001. Por lo tanto, como el número de parejas al sumar 1000 elementos es 500, el resultado de la sumatoria es 500(1001), como se ilustra a continuación:

$$1 + 2 + 3 + \ldots + 500 + 501 + \ldots + 998 + 999 + 1000 = 500(1001)$$

Ejemplo 4.29. Para demostrar la respuesta de Gauss se usa inducción matemática, por lo que su planteamiento se representa como una proposición en función de n:

$$1 + 2 + 3 + \ldots + n = \frac{n(n+1)}{2}$$

Inicio — Término n-ésimo — Resultado

Se entiende que si está en función de n, la proposición $P(n)$ es verdadera $\forall n \in Z$ y no solamente para múltiplos de 10. Pero para efectos de su representación considérese que $n = 1000$ y por lo tanto $1001 = (n + 1)$, $500 = \frac{n}{2}$ y el término n-ésimo en este caso es n.

Paso básico. Para demostrar que $P(n)$ es verdadera cuando $k = n = 1$, se sustituye 1 en el término n-ésimo, que en este caso es n:

$$n = 1$$

Si al sustituir $n = 1$ en el término n-ésimo se obtiene como resultado el primer elemento de la sumatoria, se dice que el "paso básico" se cumple, como ocurre en este caso.

Paso inductivo. En el paso inductivo se debe probar que $P(n)$ es cierta cuando $k = n + 1$, sustituyendo $(n + 1)$ en todas las "enes" del término n-ésimo y sumándolas a ambos miembros de la igualdad, hasta llegar a una expresión semejante a la que está al lado derecho del signo igual (en este caso $\frac{n(n+1)}{2}$), pero en lugar de que esté en función de n deberá estar en función de k.

Primeramente se sustituye $(n + 1)$ en todas las "enes" del término n-ésimo y se suma a ambos miembros:

$$1+2+3+\ldots+n+(n+1) = \frac{n(n+1)}{2}+(n+1)$$

$$= \frac{n(n+1)+2(n+1)}{2}$$

$$= \frac{(n+1)(n+2)}{2} \qquad \text{Factorizando } (n+1)$$

$$= \frac{(n+1)(n+1+1)}{2}$$

$$= \frac{(\quad+1)}{2} \qquad \text{Sustituyendo } k = n+1$$

Como se obtuvo $\frac{(\quad+1)}{2}$ que es igual a $\frac{n(n+1)}{2}$, se dice que se cumple el paso inductivo. Debido a que tanto el paso básico como el inductivo se cumplen, se afirma que $P(n)$ es verdadera para todo valor entero de n y que por lo tanto Gauss tenía razón para el caso particular de $n = 1000$.

Ejemplo 4.30. Considérese que se desea demostrar por inducción matemática la siguiente proposición $P(n)$:

$$2+5+8+\ldots+(3n-1) = \frac{n(3n+1)}{2}$$

Paso básico. Sea $k = n = 1$, entonces:

$$[3(1)-1] = 2$$

Como al sustituir $n = 1$ en el término n-ésimo $(3n-1)$ se obtiene como resultado el primer elemento de la sumatoria, se dice que el "paso básico" se cumple.

Paso inductivo. Sea $k = n+1$. Sustituyendo $(n+1)$ en todas las "enes" del término n-ésimo y sumándolo a ambos lados de la igualdad se tiene que:

$$2+5+8+\ldots+(3n-1)+[3(n+1)-1] = \frac{n(3n+1)}{2}+[3(n+1)-1]$$

$$= \frac{n(3n+1)+2[3(n+1)-1]}{2}$$

$$= \frac{3n^2+n+2(3n+2)}{2}$$

$$= \frac{3n^2 + n + \quad n + \quad}{2}$$

$$= \frac{3n^2 + \quad n + \quad}{2}$$

$$= \frac{(n+1)(3n+\quad)}{2}$$

$$= \frac{(n+1)(3(n+1)+1)}{2}$$

$$= \frac{k(3k+1)}{2} \qquad \text{Sustituyendo } k = n+1$$

Por lo tanto, también se cumple el paso inductivo y se dice que $P(n)$ es verdadera.

4.10 Aplicación de la lógica matemática

La lógica matemática no es de reciente creación, no surgió con el uso de las computadoras, por el contrario se ha consolidado en nuestro tiempo porque es una herramienta fundamental para mejorar el software y hardware que conocemos.

La historia de la lógica tiene sus inicios en el siglo III a. C. con la "Teoría silogista" de Aristóteles, quien introdujo los cuantificadores $\forall$ y $\exists$, así como reglas de inferencia conocidas como el silogismo hipotético:

$$\begin{array}{l} \mathbf{p \rightarrow q} \\ \mathbf{q \rightarrow r} \\ \hline \therefore \mathbf{p \rightarrow r} \end{array}$$

Esta regla se aplica en matemáticas y programación, algunas veces sin saber que se trata del silogismo hipotético:

$$\begin{array}{l} \mathbf{X > Y} \\ \mathbf{Y > Z} \\ \hline \therefore \mathbf{X > Z} \end{array}$$

También se encuentra disfrazada en algunas líneas de código de la siguiente manera:

If X > Y and Y > Z then X > Z

Aunque en sus inicios se usó principalmente para elaborar demostraciones matemáticas, en su aplicación a la programación el procedimiento de la demostración equivale a desarrollar un algoritmo para resolver un proble-

ma, usando para ello las instrucciones válidas (asignación, ciclos, lectura, escritura, declaración, etc.) de un lenguaje formal. Tanto el procedimiento de demostración como el diseño de algoritmos, dependen exclusivamente de la lógica usada por la persona que los desarrolla. Los caminos en ambas situaciones pueden ser más o menos eficientes, pero lo interesante en ambos casos es que permiten usar la creatividad y reflexión de la persona para lograr el objetivo, ya que no existe una forma única de demostrar un teorema o desarrollar un algoritmo.

En tiempos remotos Crisipo de Sodi (281-206 a. C.) introdujo los operadores lógicos de la conjunción ($\wedge$), la disyunción ($\vee$), la implicación ($\rightarrow$), la disyunción exclusiva ($\oplus$) y la complementación ($'$), así como los valores de "falso" o "verdadero". Con esos operadores lógicos, muchos siglos después Augustus De Morgan (1806-1871) enunció sus famosas leyes de De Morgan:

$$(p \vee q \vee \ldots \vee z)' \equiv (p' \wedge q' \wedge \ldots \wedge z')$$

$$(p \wedge q \wedge \ldots \wedge z)' \equiv (p' \vee q' \vee \ldots \vee z')$$

Que tienen aplicación no sólo en lógica matemática sino también en teoría de conjuntos. A partir de esta información George Boole (1815-1864) creó el álgebra booleana, la cual tiene amplias aplicaciones en la construcción de computadoras, robótica y automatización de sistemas eléctricos, mecánicos y electrónicos.

La lógica matemática también proporciona elementos para la creación de nuevos lenguajes de programación, al permitir estructurar sintáctica y semánticamente el lenguaje que se está desarrollando. En relación con esto, a continuación considérese la semejanza entre las composiciones de un lenguaje formal y las proposiciones lógicas que se vieron en el capítulo.

Sea

$$\Sigma = \{a, g, h, i, l, m, o, r\} \qquad \text{(Alfabeto)}$$

Cuyas composiciones son

$S \rightarrow hA$	$B \rightarrow rD$	$F \rightarrow gC$
$A \rightarrow oB$	$D \rightarrow mE$	$C \rightarrow a$
$B \rightarrow lC$	$E \rightarrow iF$	

Las composiciones permiten saber si una palabra es válida en un lenguaje, y este proceso de validación lo llevan a cabo los compiladores de un lenguaje de programación para determinar si las instrucciones de un programa están correctamente escritas. Actualmente mediante el uso de lenguajes formales y de la variedad de herramientas que proporciona la lógica matemática, se está trabajando en la simulación de lenguajes naturales que permitan una comunicación más amplia con la computadora.

Otra aplicación importante de la lógica matemática se encuentra en las bases de datos, en donde se consideran los archivos como relaciones que pueden manipularse por medio de operadores lógicos para obtener nuevos reportes de información, dando origen a lo que se conoce como "álgebra relacional" en la cual se basan todos los manejadores de bases de datos conocidos. Las redes de computadoras también utilizan el concepto de relación para representar la comunicación entre computadoras, de forma que es posible realizar operaciones lógicas entre matrices booleanas para obtener características necesarias en una red. Por todo lo anterior, se puede decir que la lógica matemática es esencial en la computación ya que permite sentar las bases para el entendimiento formal de prácticamente todas las áreas de ésta (bases de datos, programación, inteligencia artificial, lenguajes formales, sistemas digitales, redes, etcétera).

4.11 Resumen

La lógica es una disciplina que por medio de reglas y técnicas, determina si un razonamiento es válido. El elemento fundamental de la lógica es la proposición.

Una proposición es una oración, frase o expresión matemática que puede ser falsa o verdadera, pero no ambas a la vez.

Los siguientes son dos ejemplos de proposiciones:

p: Miguel de Cervantes Saavedra escribió la obra el *Quijote de la Mancha*.

q: $(y - 1) > (3x + 2)$

Los operadores lógicos básicos son *and* ($\wedge$), *or*($\vee$) y *not*($'$). Además de los operadores básicos, es posible usar las proposición condicional ($\rightarrow$) y bicondicional ($\leftrightarrow$) para representar enunciados más complejos, como se muestra en el siguiente ejemplo.

"Si Compro una bicicleta o me levanto más temprano, entonces, no llegaré tarde a la escuela. Reprobaré el semestre si y sólo si llego tarde a la escuela. En conclusión; si llegué tarde a la escuela y reprobé el semestre, entonces no compré una bicicleta o no me levanté temprano."

Sean

p: Compré una bicicleta.

q: Me levanté más temprano.

r: Llegué tarde a la escuela.

s: Reprobé el semestre.

Por lo tanto, el enunciado anterior se puede representar con notación lógica de la siguiente manera:

$$[(p \vee q) \rightarrow r'] \wedge [s \leftrightarrow r] \Rightarrow [(r \wedge s) \rightarrow (p' \vee q')]$$

Es posible cambiar de tiempo las proposiciones para que tenga sentido, de tal manera que en lugar de decir compraré una bicicleta es posible decir compré una bicicleta o compro una bicicleta sin que esto afecte la representación del enunciado.

El enunciado anterior tiene formato de teorema, en donde las proposiciones $[(p \vee q) \rightarrow r']$ y $[s \leftrightarrow r]$ son las hipótesis y la proposición $[(r \wedge s) \rightarrow (p' \vee q')]$ es la conclusión. Lo que separa a las hipótesis de la conclusión es el símbolo $\Rightarrow$. Enunciados como éstos es posible demostrarlos por medio del método directo o el método por contradicción.

Se dice que una proposición es una tautología, si el resultado es verdadero para todos sus valores de verdad. Ejemplo:

p	p'	$p \vee p'$
0	1	1
1	0	1

Una proposición es una contradicción si el resultado es falso para todos los valores de verdad.

p	p'	$p \wedge p'$
0	1	0
1	0	0

Se dice que dos proposiciones son lógicamente equivalentes si sus resultados son iguales para todos sus valores de verdad.

p	q	p'	q'	$p \wedge q'$	$p' \wedge q$	$p \wedge q' \vee p' \wedge q$	$(p \leftrightarrow q)'$	$p \oplus q$
0	0	1	1	0	0	0	0	0
0	1	1	0	0	1	1	1	1
1	0	0	1	1	0	1	1	1
1	1	0	0	0	0	0	0	0

En la tabla anterior los resultados de las últimas tres columnas son iguales por lo tanto se dice que son lógicamente equivalentes y se escribe:

$$p \wedge q' \vee p' \wedge q \equiv (p \leftrightarrow q)' \equiv p \oplus q$$

En una demostración formal una regla de inferencia permite encontrar proposiciones válidas a partir de otras que también se consideran válidas. Ejemplo: considerar que $[p' \rightarrow (q \vee r')]$ y $[(q \vee r') \rightarrow s]$ son válidas, aplicando la regla de inferencia conocida como "silogismo hipotético" se puede encontrar que $[p' \rightarrow s]$ también es válida.

Es posible demostrar que un teorema es válido con el apoyo de tautologías, equivalencias lógicas y reglas de inferencia.

Además de la lógica proposicional existe también la lógica de predicados o lógica de conjuntos que considera a las proposiciones lógicas como conjuntos de elementos, en donde no todos los elementos de un conjunto cumplen con las condiciones para decir que son verdaderos (o falsos) totalmente, de tal manera que se introducen los cuantificadores universal ($\forall$) y existencia ($\exists$) para la representación de enunciados. En la lógica de predicados se debe además indicar cuál es el dominio (**U**). Ejemplo.

Sea:

U = {x /x es un automóvil}

p: Son caros.

q: Son veloces.

r: Son lujosos.

"Si algunos automóviles son veloces y lujosos, entonces son caros. Algunos son lujosos y no son veloces. En conclusión si todo automóvil es caro, entonces es veloz o lujoso."

$$\exists x[[q(x) \wedge r(x)] \rightarrow p(x)] \wedge \exists x[r(x) \wedge q'(x)] \Rightarrow \forall x[p(x) \rightarrow [q(x) \vee r(x)]]$$

Los operadores lógicos de lógica de proposiciones son también válidos en lógica de predicados y la evaluación se realiza de la misma manera.

Los métodos directo y por contradicción usados en lógica de proposiciones y de predicados son métodos de demostración inductivos en donde se va de lo general a lo particular. Existen también métodos de demostración formal en donde se va de lo particular a lo general, demostrando que la proposición es verdadera para el primer elemento (n = 1) para n y para (n + 1) el cual recibe el nombre de inducción matemática.

4.12 Problemas

4.1. Representar en forma de teorema cada uno de los siguientes enunciados, usando para ello notación lógica:

a) "Si vivo en un lugar bajo, entonces se inunda la casa. Si vivo en un lugar alto, entonces me falta el agua o es zona cara. Por consiguiente, si no es zona cara y no se inunda la casa y me falta el agua, entonces vivo en la montaña."

b) "Está en la selección si y sólo si es buen jugador y tiene una edad menor de 27 años o pertenece al América. Si está en la selección y no es buen jugador o no pertenece al América, entonces es del Morelia. Por lo tanto, si es del Morelia, entonces es buen jugador."

c) "Si estudia informática o sistemas, entonces es alumno del Tecnológico. Es alumno del Tecnológico si y sólo si es buen estudiante. Por consiguiente, si no estudia sistemas o informática y no es alumno del Tecnológico, entonces no es buen estudiante."

d) "El programa corre, si y sólo si no tiene errores de compilación. Si no tiene errores de lógica y no tiene errores de compilación, entonces el programa está bien y los resultados son satisfactorios. Por lo tanto, si tiene errores de compilación o tiene errores de lógica, entonces el programa no corre y los resultados no son satisfactorios."

e) "Si se realiza un buen diseño de la base de datos y se hace una buena programación, entonces se accesará rápidamente la información. Si no se hace buena programación, entonces toma mucho tiempo corregir el programa. Por lo tanto, si no se accesa rápidamente la información y toma mucho tiempo corregir el programa, entonces no se ha realizado un buen diseño de la base de datos."

4.2. Representar en forma de teorema cada uno de los siguientes enunciados, usando para ello notación lógica:

a) "Haré la tarea de matemáticas para computación, si y sólo si tengo tiempo. Iré a la disco, si y sólo si tengo tiempo y tengo dinero. Si no tengo dinero, entonces haré la tarea de matemáticas para computación y veré un buen programa de televisión. Por lo tanto, si veo un buen programa de televisión y tengo tiempo, entonces haré la tarea de matemáticas para computación."

b) "Gana medalla en los juegos olímpicos, si y sólo si es buen deportista y tiene una edad menor a 27 años, o no lo descalifican los jueces. No es mexicano. De tal manera que, si no gana medalla y no es buen deportista, o lo descalifican los jueces, entonces es mexicano."

c) "Si estudia informática o estudia sistemas, entonces es alumno del Tecnológico. Es alumno del Tecnológico, si y sólo si es buen estudiante. Por consiguiente, si no estudia sistemas o informática y no es alumno del Tecnológico, entonces es un mal estudiante."

d) "Si tengo conocimientos de computación y domino el inglés, entonces no tendré problemas para encontrar trabajo. Si tengo problemas para encontrar trabajo, entonces tengo más de 40 años o no me preparé lo suficiente. Por lo tanto, si me preparo lo suficiente y no tengo más de 40 años y domino el inglés, entonces no tendré problemas para encontrar trabajo."

4.3. Elaborar la tabla de verdad para cada una de las siguientes proposiciones compuestas:

a) $[(p \to q)' \to r] \to (p' \vee r' \wedge q)$

b) $p \to q' \vee r \leftrightarrow p \wedge q \to r'$

c) $(p \to r) \leftrightarrow [(q \vee r \wedge p') \to r']'$

d) $[(p \to q) \to r'] \wedge [(p' \vee r) \leftrightarrow q')]$

e) $p \to q \leftrightarrow r' \vee q' \to p' \wedge r$

f) $[[((p \wedge r) \leftrightarrow q') \to p'] \to r']'$

4.4. Elaborar la tabla de verdad para cada una de las siguientes proposiciones compuestas:

a) $[((p \to q)' \to r) \vee p'] \to (r' \wedge q')$

b) $p \leftrightarrow q' \vee r \to p' \to q \wedge (p \vee q \to p')$

c) $[p \vee (r \to s')] \leftrightarrow [(p' \wedge s)' \to r']$

d) $[(p \to q') \to r] \to (p' \vee q \wedge r)'$

e) $p' \to (r' \vee q \wedge p) \leftrightarrow r \vee q' \to p$

f) $[(p \to q') \to (q \vee r \wedge p')] \leftrightarrow [(q \to p)' \to r]$

4.5. Demostrar que las proposiciones de cada uno de los incisos siguientes son lógicamente equivalentes, usando para ello tautologías y/o las equivalencias lógicas restantes:

a) $[(p \to r) \wedge (q \to r)] \equiv [(p \wedge q) \to r]$

b) $[p \vee (q \wedge r)] \equiv [(p \wedge p) \vee (p \wedge r) \vee (p \wedge q) \vee (q \wedge r)]$

4.6. Demostrar que las proposiciones de cada uno de los incisos siguientes son lógicamente equivalentes, usando para ello tautologías y/o las equivalencias lógicas restantes:

a) $[(p \to q) \wedge (p \to r)] \equiv [p \to (q \wedge r)]$

b) $(p \to q) \equiv (p' \vee q)$

c) $[p \wedge (s \vee r')] \equiv [p \to (s \vee r')']'$

d) $[(p \vee s) \to (q \wedge p \vee s')] \equiv [(q \wedge p \vee s')' \to (p \vee s)']$

4.7. Demostrar por medio de una tabla de verdad que la regla 7a realmente es una tautología.

4.8. Demostrar por medio de una tabla de verdad que las reglas 4a, 6a, 8c y 9a realmente son tautologías.

4.9. Establecer si los siguientes enunciados son válidos o no. Explicar su respuesta:

a) $(q' \vee p') \wedge (r \wedge q) \Rightarrow (p \leftrightarrow r')$

b) $(r \to p') \wedge (q' \vee r') \Rightarrow (p' \to q)$

c) $(p' \to r) \wedge [(p' \to r) \to (q' \wedge p)] \Rightarrow (q' \wedge p)$

4.10. Establecer si los siguientes enunciados son válidos o no. Explicar su respuesta:

a) $[(p' \vee r) \to q] \wedge q' \Rightarrow (p' \vee r)'$

b) $(p \leftrightarrow q') \wedge (q \vee r) \Rightarrow (p' \to r')$

c) $[(p' \wedge r) \to (q \to r')] \Rightarrow [[(q \to r') \to (p \vee q')] \to [(p' \wedge r) \to (p \vee q')]]$

d) $(p \to r') \wedge (p' \leftrightarrow q) \Rightarrow (q' \vee r)$

e) $(r \to q') \wedge [(p \wedge q') \to r'] \Rightarrow (q \to r)$

4.11. Representar el siguiente enunciado en forma de teorema y llevar a cabo su demostración usando el método directo y el método por contradicción:

"Si tengo mucho dinero o estoy muy carita, entonces las muchachas me quieren. Ninguna quiere salir conmigo. Si las muchachas me quieren, entonces todas quieren salir conmigo. Por lo tanto, no tengo mucho dinero."

4.12. Representar en forma de teorema el siguiente enunciado y llevar a cabo su demostración usando el método directo y el método por contradicción:

"Si estudio electrónica, entonces realizaré investigación en sistemas digitales. Si estudio informática, entonces manejaré la información de una empresa. Si realizo investigaciones en sistemas digitales o manejo la información de una empresa, entonces estaré feliz. Por lo tanto, si no estoy feliz, entonces no estudié electrónica y no estudié informática."

4.13. Representar en forma de teorema el siguiente enunciado y llevar a cabo su demostración usando el método directo y el método por contradicción:

"Si se ha realizado un buen diseño de la base de datos y se hace una buena programación, entonces se accesa rápidamente la información. Si no se hace buena programación, entonces toma mucho tiempo corregir el programa. Por lo tanto, si no se accesa rápidamente la información y toma poco tiempo corregir el programa, entonces no se ha realizado un buen diseño de la base de datos."

4.14. Demostrar por el método directo el teorema de cada uno de los siguientes incisos:

a) $[p \rightarrow (q \wedge r)] \wedge [(q \vee s) \rightarrow t] \wedge (p \vee s) \Rightarrow (r \vee t)$

b) $[(p \wedge q) \rightarrow r] \wedge [q' \rightarrow s'] \Rightarrow [(r' \wedge s) \rightarrow p']$

c) $[(p' \wedge r) \rightarrow q] \wedge q' \wedge [r \rightarrow s] \Rightarrow [r \rightarrow (s \wedge p)]$

d) $[p' \rightarrow q'] \wedge (r \rightarrow s') \wedge [(q' \vee s') \rightarrow w] \Rightarrow [w' \rightarrow (p \wedge r')]$

e) $[p \leftrightarrow q'] \wedge [r \vee q'] \wedge p' \wedge [p' \rightarrow r] \Rightarrow [[r \rightarrow (r \vee q)] \wedge p]$

4.15. Demostrar por el método directo el teorema de cada uno de los siguientes incisos:

a) $[(r \vee q) \rightarrow q'] \wedge [p' \rightarrow r] \wedge (q \vee r) \wedge [q \rightarrow p] \Rightarrow$
$[q' \wedge [p' \rightarrow (q' \wedge r)]]$

b) $[q \rightarrow (p \wedge s)] \wedge [s' \rightarrow r] \Rightarrow [(s' \wedge p' \vee s') \rightarrow (r \wedge q')]$

c) $[(q \vee r') \rightarrow s] \wedge [t \rightarrow q'] \Rightarrow [(q \vee s') \rightarrow (t' \vee r)]$

d) $[q \wedge r] \wedge [p \rightarrow q'] \wedge [s \rightarrow (q \rightarrow r')]] \Rightarrow [s' \wedge (q \rightarrow p')]$

e) $[(q \vee s) \rightarrow t] \wedge t' \Rightarrow [(q' \wedge t') \wedge (q \rightarrow t)]$

f) $[p \leftrightarrow q'] \wedge [p' \rightarrow r] \Rightarrow [p' \rightarrow (q \vee r)]$

g) $[(p \wedge r) \rightarrow q] \wedge [r' \rightarrow p] \wedge [p \wedge r']' \Rightarrow [r \vee q]$

4.16. Demostrar por contradicción cada uno de los incisos del ejercicio 4.9.

4.17. Demostrar de dos maneras diferentes usando el método directo y por lo menos de una forma usando el método de contradicción, cada uno de los incisos siguientes:

a) $[(p \vee q) \rightarrow r] \wedge [r \rightarrow s] \Rightarrow [s' \rightarrow q']$

b) $[(p \wedge q) \rightarrow r] \wedge [q' \rightarrow s'] \Rightarrow [(r' \wedge s) \rightarrow (q \wedge p')]$

c) $[p \rightarrow (q \wedge r)] \wedge [(q \vee s) \rightarrow t] \wedge (p \vee s) \Rightarrow t$

d) $[p' \rightarrow r] \wedge q' \wedge (p \vee r') \wedge (r \rightarrow q) \Rightarrow [[p' \rightarrow (p \wedge q)] \wedge q']$

e) $[p' \rightarrow q'] \wedge [r' \rightarrow s'] \wedge [(q' \vee s') \rightarrow t] \Rightarrow [t' \rightarrow (p \wedge r)]$

4.18. Demostrar usando inducción matemática que las proposiciones de cada uno de los siguientes incisos son verdaderas:

a) $3 + 6 + 20 + \ldots + [n\,(n!) + 2] = (n+1)! + 2n - 1$

b) $0 + 3 + 8 + \ldots + (n^2 - 1) = \dfrac{n(2n+5)(n-1)}{6}$

c) $0 + 7 + 26 + \ldots + (n^3 - 1) = \dfrac{n[n(n+1)^2 - 4]}{6}$

d) $2 - 3 + 10 - 15 + \ldots + = [(\ 1)^{n+1} n^2 + 1] = \dfrac{n[(\ 1)^{n+1}(n+1) + 2]}{2}$

e) $a(a^{-1} + 1) + a(a^0 + 1) + a(a^1 + 1) + a(a^2 + 1) + \ldots + a(a^{n-1} + 1)$

$= \dfrac{1 - a^{n+1}}{1 - a}$ $\qquad n \in N$

4.19. Demostrar usando inducción matemática que las proposiciones de cada uno de los siguientes incisos son verdaderas:

a) $5 + 15 + 25 + 35 + \ldots + (10n - 5) = 5n^2$

b) $\dfrac{3}{1 \cdot 3} + \dfrac{3}{3 \cdot 5} + \dfrac{3}{5 \cdot 7} + \dfrac{3}{7 \cdot 9} + \cdots + \dfrac{3}{(2n-1)(2n+1)} = \dfrac{3n}{(2n+1)}$

c) $\dfrac{a}{2} + \dfrac{a^2}{2} + \dfrac{a^3}{2} + \dfrac{a^4}{2} + \cdots + \dfrac{a^n}{2} = \dfrac{(a^{n+1} - 1)}{2(a-1)}$ $\qquad a \in R;\ a \neq 0;\ a \neq 1$

d) $2^n \geq n^2$ $\qquad n \in Z^+;\ n \geq 4$

e) $2^0 + 2^1 + 2^2 + 2^3 + \cdots + 2^n = 2^{n+1} - 1$ $\qquad n \in N$

f) $(2^{1-1}) + (2^{2-1}) + (2^{3-1}) + \cdots + (2^{n-1}) = 2^n - 1$

4.20. Representar el siguiente algoritmo en forma de sumatoria, encontrar la fórmula del n-ésimo término, la expresión matemática del resultado y usar inducción matemática para llevar a cabo la prueba de dicha proposición:

$x = 1;$
$s = 0;$
Mientras $(x \leq n)$ hacer
  Inicio
    $e = 2^x + 1;$
    $s = s + e;$
    Imprimir (e);
    $x = x + 1;$
  Fin
  Imprimir (s).

4.21. Representar el siguiente algoritmo en forma de sumatoria, encontrar la fórmula del n-ésimo término, la expresión matemática del resultado y usar inducción matemática para llevar a cabo la prueba de dicha proposición:

$x = 1;$
$s = 0;$
Mientras $(x \leq n)$ hacer
  Inicio
    $e = 3x - 1;$
    $s = s + e;$
    Imprimir (e);
    $x = x + 1;$
  Fin
  Imprimir (s);

4.22. Demostrar usando inducción matemática que el "sort de selección con intercambio" (selection with exchange) lleva a cabo $\frac{n^2 - n}{2}$ comparaciones para ordenar información en el peor de los casos.

4.23. Demostrar usando inducción matemática que el "sort de la burbuja" (bubble sort) lleva a cabo $\frac{n^2 - n}{2}$ comparaciones para ordenar información en el peor de los casos.

4.24. Demostrar por medio de inducción matemática que un árbol tiene $(n-1)$ aristas. Aquí n es el número de nodos.

4.25. Sea $U = \{x \mid x \text{ es un animal}\}$. Encontrar los elementos necesarios para llevar a cabo la representación de cada una de las frases, usando notación lógica. Decir si el enunciado es falso o verdadero.

a) "Todos los animales tienen alas"

b) "Algunos animales vuelan"

c) "Algunos animales tienen alas y vuelan"

d) "Algunos animales tienen alas y no vuelan"

e) "Si es ave, entonces tiene alas"

f) "Si es ave, pone huevos y cacaraquea, entonces es gallina"

g) "Algunas gallinas no ponen huevos"

h) "Si es ave entonces no es mamífero"

4.26. Sea $U = \{x \mid x \text{ es un animal}\}$. Encontrar los elementos necesarios para llevar a cabo la representación de cada una de las frases, usando notación lógica. Decir si el enunciado es falso o verdadero.

a) "Todos los gatos son carnívoros"

b) "Si es carnívoro y no es perro, entonces es un gato"

c) "Es carnívoro si y sólo si es perro o gato"

d) "Ningún gato canta"

e) "Si canta entonces no es perro ni gato"

f) "Es carnívoro. No canta. No es perro ni gato, en conclusión es un león"

4.27. Decir con palabras el significado de cada uno de los siguientes enunciados, así como indicar cuál es el valor de verdad para cada uno de los incisos. Sea $U = \{x, y \mid x, y \in \mathrm{R}\}$; p: "$x^2 - 1 = y$".

a) $\forall x \exists y \, \mathrm{p}(x, y)$

b) $\exists x \forall y \, \mathrm{p}(x, y)$

c) $\forall y \exists x \, \mathrm{p}(x, y)$

d) $\forall x \forall y \, \mathrm{p}(x, y)$

e) $\exists x \exists y \, \mathrm{p}(x, y)$

4.28. Decir con palabras el significado de cada uno de los enunciados, así como indicar cuál es el valor de verdad para cada uno de los incisos.

Sea $U = \{x, y \mid x, y \in \mathrm{R}\}$; p: "$x - y = 1$".

a) $\exists y \, \forall x \, \mathrm{p}(x, y)$

b) $\forall x \, \exists y \, \mathrm{p}(x, y)$

c) $\exists x \, \forall y \, \mathrm{p}(x, y)$

d) $\forall y \, \exists x \, \mathrm{p}(x, y)$

e) $\forall x \, \forall y \, \mathrm{p}(x, y)$

f) $\exists x \, \exists y \, \mathrm{p}(x, y)$

4.29. Representar con notación lógica los enunciados de cada uno de los siguientes incisos.

Sea $U = \{x, y \mid x, y \in \mathrm{R}\}$; p: "$y = 2x - 1$"; q: "$x \geq y$"; r: "$y = \frac{1}{x}$":

a) Si existen algunas "y" que para toda "x" tal que si "$y = 2x - 1$" y "$x \geq y$" entonces "$y = \frac{1}{x}$"

b) Si para toda "y" existe alguna "x" tal que si "$y \neq 2x - 1$" o "$y = \frac{1}{x}$"; entonces "$x < y$"

c) Si para alguna "x", existe alguna "y" tal que "$y = \frac{1}{x}$" o para toda "x" "$x \geq y$" y existe alguna "y" tal que "$y \neq 2x - 1$"

4.30. Representar con notación lógica los enunciados de cada uno de los siguientes incisos.

Sea $U = \{x, y \mid x, y \in \mathrm{R}\}$; p$(x, y)$: "$x \leq y$"; q: "$x - y = 1$"; r: "$x^2 + y^2 = 1$":

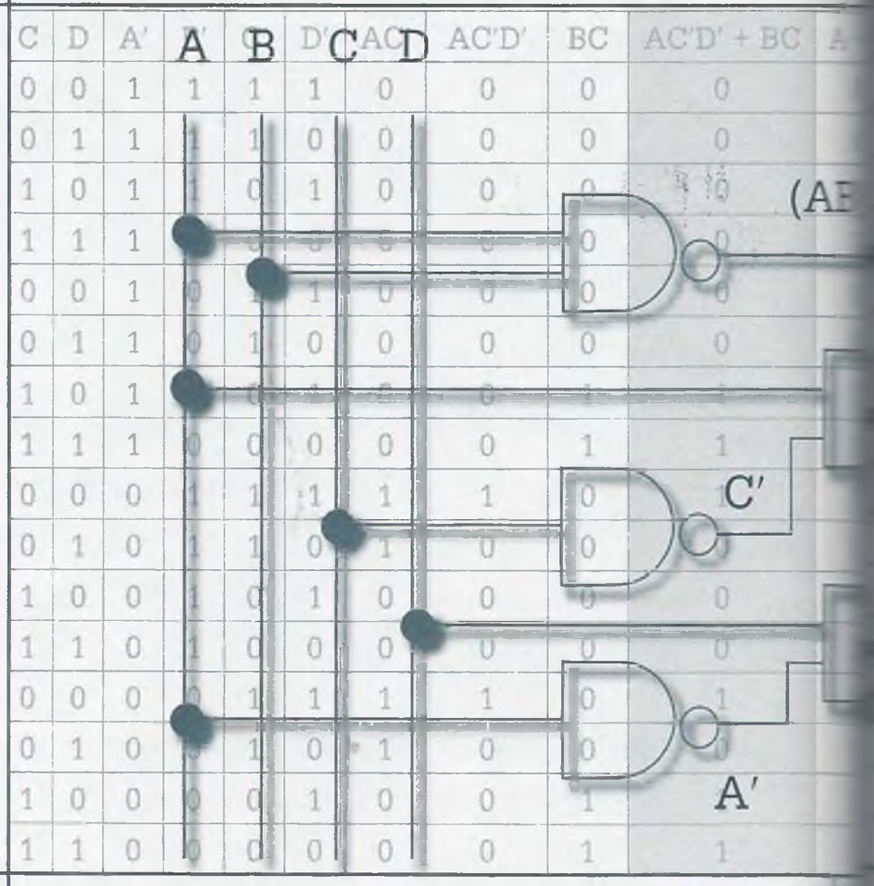
a) Existen algunas "x" que para toda "y" tal que "$x^2 + y^2 = 1$" y "$x > y$" o "$x - y = 1$"

b) Si para toda "x" existe alguna "y" tal que si "$x \leq y$"; entonces para alguna "y" existe alguna "x" tal que "$x - y \neq 1$" o "$x^2 + y^2 = 1$"

c) Si para toda "x" y para toda "y" tal que "$x > y$" y si para alguna "$x \leq y$", entonces "$x^2 + y^2 = 1$"

CAPÍTULO V

Álgebra booleana



F	A	B	C	D	A′	B′	C′	D′	AC′	AC′D′	BC′
0	0	0	0	0	1	1	1	1	0	0	0
0	0	0	0	1	1	1	1	0	0	0	0
0	0	0	1	0	1	1	0	1	0	0	0
0	0	0	1	1	1	1	0	0	0	0	0
0	0	1	0	0	1	0	1	1	0	0	0
0	0	1	0	1	1	0	1	0	0	0	0
1	0	1	1	0	1	0	0	1	0	0	1
1	0	1	1	1	1	0	0	0	0	0	1
1	1	0	0	0	0	1	1	1	1	1	0
0	1	0	0	1	0	1	1	0	1	0	0
0	1	0	1	0	0	1	0	1	0	0	0
0	1	0	1	1	0	1	0	0	0	0	0
1	1	1	0	0	0	0	1	1	1	1	0
0	1	1	0	1	0	0	1	0	1	0	0
1	1	1	1	0	0	0	0	1	0	0	1
1	1	1	1	1	0	0	0	0	0	0	1

$$F = [(AB)'(AC')'(A'D)']'$$

$$F = AB + AC' + A'D$$

Boole interpretó su sistema a la manera aristotélica, como un álgebra de clases y de sus propiedades, y al hacerlo amplió la antigua lógica de clases y la liberó de los límites del silogismo.

Martin Gardner

Objetivos

- Aprender a simplificar expresiones booleanas usando teoremas del álgebra booleana.
- Aprender a simplificar expresiones booleanas por medio de mapas de Karnaugh.
- Representar expresiones booleanas por medio de bloques lógicos.

5.1 Introducción

George Boole
(1815-1864)

Fue un matemático británico que es considerado como uno de los fundadores de las ciencias de la computación debido a su creación del álgebra booleana, la cual es la base de la aritmética computacional moderna.

Con una formación autodidacta, Boole fue profesor a la edad de 16 años y a partir de 1835 comenzó a aprender matemáticas por sí mismo. En este periodo estudió los trabajos de Laplace y Lagrange, comenzó a estudiar álgebra y en el *Transaction of the Royal Society* publicó *Aplicación de métodos algebraicos para la solución de ecuaciones diferenciales*, trabajo por el cual recibió la medalla de la Real Sociedad.

En 1849 Boole ocupó una cátedra de matemáticas en el Queens College, y permaneció en este puesto por el resto de su vida. En 1854 publicó *Las leyes del pensamiento*, obra en la que plantea la lógica en términos de un álgebra simple que se conoce como álgebra booleana y que se aplica en la ciencia de la computación y en el análisis de circuitos.

Otras áreas de interés de Boole fueron las ecuaciones diferenciales en relación con las cuales escribió *Tratado en ecuaciones diferenciales* que publicó en 1859, el cálculo de las diferencias finitas que expuso en *Tratado sobre el cálculo de las diferencias finitas* publicado en 1860, y los métodos generales en probabilidad.

El álgebra booleana fue desarrollada por George Boole y en su libro *An Investigation of the Laws of Thought*, publicado en 1854, muestra las herramientas para que las proposiciones lógicas sean manipuladas en forma algebraica. Debido al carácter abstracto de sus principios no tuvo una aplicación directa sino hasta 1938 en que la compañía de teléfonos Bell de Estados Unidos la utilizó para realizar un análisis de los circuitos de su red telefónica. En ese mismo año Claude E. Shannon, entonces estudiante de postgrado del Instituto Tecnológico de Massachussets, a partir del álgebra de Boole creó la llamada álgebra de conmutación para representar las propiedades de conmutación eléctrica biestables, demostrando con esto que el álgebra booleana se adapta perfectamente al diseño y representación de circuitos lógicos de control basados en relés e interruptores.

Los circuitos lógicos de control tienen una gran importancia ya que las computadoras, los sistemas telefónicos, los robots y cualquier operación automatizada en una empresa, son algunos de los ejemplos de la aplicación de éstos y del álgebra booleana.

Una señal es la representación de información, y puede aparecer en forma de valor o de una cadena de valores de una magnitud física. Existen principalmente dos clases de señales: analógicas y digitales.

La señal analógica tiene como característica principal el continuo cambio de magnitud, de la misma manera que una corriente eléctrica y una presión de gas.

En la señal digital los posibles valores de tensión están divididos en un número infinito de intervalos, a cada uno de los cuales está asignado un valor o una cadena de valores como información. Una señal digital puede obtenerse de una manera analógica asignando ciertos umbrales de sensibilidad.

La señal binaria es una señal digital con sólo dos valores posibles: conectado-desconectado, verdadero-falso, 1-0.

5.2 Expresiones booleanas

El álgebra booleana trabaja con señales binarias. Al mismo tiempo una gran cantidad de sistemas de control, también conocidos como digitales, usan señales binarias y éstas son un falso o un verdadero que proviene de sensores que mandan la información al circuito de control, mismo que lleva a cabo la evaluación para obtener un valor que indicará si se lleva a cabo o no una determinada actividad, como encender un foco, arrancar un equipo de ventilación en un cine o ejecutar una operación matemática en una computadora.

Los sensores pueden ser "ópticos", como los que se usan en tiendas departamentales (de proximidad); "magnéticos", como los que permiten detectar armas en aeropuertos; de "temperatura", como los que utiliza un sistema de calefacción, los refrigeradores o bien el mismo termostato que controla el sistema de enfriamiento del motor de un vehículo; de "nivel", ya que un flotador como el que tiene un tinaco o una cisterna para controlar la cantidad de agua, es un sensor que puede mandar información a un circuito de control.

En cada uno de estos grupos de sensores existen tipos, tamaños y modelos, de acuerdo con el uso y funcionamiento, de forma que existen infrarrojos, láser, fotoeléctricos y de ultrasonido, entre otros.

Para resolver un problema práctico en el cual se desea automatizar un proceso, es necesario realizar un análisis detallado de lo que se quiere lograr así como de los tipos de sensores necesarios para obtener las señales. Una vez que se conoce esto se plantea el funcionamiento del circuito lógico en una expresión matemática, la cual recibe el nombre de función booleana, y cada una de las variables de que está integrada esta función representa un sensor que provee al circuito de una señal de entrada.

Ejemplo 5.1. Supóngase que en una industria refresquera se desea que un sistema automático saque de la banda de transportación un refresco que no cumple con los requisitos mínimos de calidad, y que para esto se cuenta con cuatro sensores en diferentes puntos del sistema de transportación para revisar aspectos importantes de calidad. Supóngase además que los sensores son A, B, C y D y que el sistema F sacará al refresco si los sensores emiten el siguiente grupo de señales:

A	B	C	D	F
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	1
0	1	0	0	0
0	1	0	1	0
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	1

Claude Elwood Shannon
(1916-2001)

Ingeniero eléctrico y matemático estadounidense, es considerado como el fundador de la teoría de la información.

En 1936 obtuvo los títulos de ingeniero electricista y matemático, y ese mismo año comenzó a desempeñarse como asistente de investigación en el departamento de ingeniería eléctrica en el Instituto de Tecnología de Massachusetts (MIT), en donde trabajó en el computador analógico más avanzado de ese tiempo (Vannevar Bush's Differential Analyzer).

En esta época surgió su interés por los circuitos de relevadores complejos e intentando simplificar sistemas telefónicos de relés se dio cuenta de que éstos podían usarse para hacer cálculos. Combinando esto con su gusto por la lógica y el álgebra booleana pudo desarrollar esta idea durante el verano de 1937, que pasó en los laboratorios Bell en la ciudad de Nueva York.

En su tesis de maestría demostró que el álgebra booleana se podía utilizar en el análisis y la síntesis de la conmutación de los circuitos digitales. La tesis despertó mucho interés cuando apareció en 1938 en las publicaciones especializadas, y un cuarto de siglo más tarde H. H. Goldstine la citó en su libro "Las computadoras desde Pascal hasta Von Neumann" y la calificó como una de las aportaciones teóricas fundamentales que ayudó a cambiar el diseño de los circuitos digitales.

Shannon pasó quince años en los laboratorios Bell y durante este periodo trabajó en muchas áreas, siendo lo más notable todo lo referente a la teoría de la información, un desarrollo que fue publicado en 1948 bajo el nombre de "Una Teoría Matemática de la Comunicación". En este trabajo demostró que todas las fuentes de información (telégrafo eléctrico, teléfono, radio, la gente que habla, las cámaras de televisión, etc.) se pueden medir y que los canales de comunicación tienen una unidad de medida similar. Mostró también que la información se puede transmitir sobre un canal si, y solamente si, la magnitud de la fuente no excede la capacidad de transmisión del canal que la conduce, y sentó las bases para la corrección de errores, supresión de ruidos y redundancia.

En el área de las computadoras y de la inteligencia artificial, en 1950 publicó un trabajo que describía la programación de una computadora para jugar al ajedrez, convirtiéndose en la base de posteriores desarrollos.

Claude Elwood Shannon falleció el 24 de febrero del año 2001, a la edad de 84 años, después de una larga lucha en contra de la enfermedad de Alzheimer.

Álgebra booleana

El álgebra booleana es un sistema algebraico que consiste en un conjunto B que contiene dos o más elementos y en el que están definidas dos operaciones, denominadas respectivamente "suma u operación OR" (+) y "producto u operación AND" (·), las cuales satisfacen las siguientes propiedades:

1) **Existencia de neutros.** En B existen el elemento neutro de la suma (0) y el elemento neutro del producto (1), tales que para cualquier elemento x de B:

   $x + 0 = x$ $\quad x \cdot 1 = x$

2) **Conmutatividad.** Para cada x, y en B:

   $x + y = y + x$ $\quad x \cdot y = y \cdot x$

3) **Asociatividad.** Para cada x, y, z en B:

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

4) **Distributividad.** Para cada x, y, z en B:

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

5) **Existencia de complementos.** Para cada x en B existe un elemento x′, llamado complemento de x, tal que:

   $x + x' = 1$ $\quad x \cdot x' = 0$

1	0	1	0	1
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	0

La función booleana que equivale a la tabla de verdad anterior es:

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

Esto implica que el refresco será extraído de la banda de transportación en cualquiera de los siguientes casos, ya que para cualquiera de ellos se tiene que F = 1:

$A = 0, \quad B = 0, \quad C = 0, D = 1$
$A = 0, \quad B = 0, \quad C = 1, D = 1$
$A = 1, \quad B = 0, \quad C = 0, D = 1$
$A = 1, \quad B = 0, \quad C = 1, D = 1$
$A = 1, \quad B = 0, \quad C = 1, D = 0$

La función booleana indica solamente los casos en donde el refresco será extraído, pero existen varios casos más en donde se dejará pasar porque cumple con los requisitos mínimos de calidad.

Se puede decir que en general una expresión booleana es un sistema de símbolos que incluyen 0, 1, algunas variables y las operaciones lógicas.

5.3 Propiedades de las expresiones booleanas

Las expresiones booleanas poseen las siguientes propiedades:

a) Están compuestas de literales (A, B, C, ...) y cada una de ellas representa la señal de un sensor. Un ejemplo es F = A′BD + AB′CD.

b) El valor de las señales o de la función sólo puede ser 0 o 1, falso o verdadero.

c) Además de literales, en la expresión booleana se puede tener el valor de 0 o 1. Por ejemplo: F = A′BD1 + AB′CD + 0.

d) Las literales de las expresiones booleanas pueden estar conectadas por medio de los operadores lógicos And ($\wedge$), Or ($\vee$) y Not ($'$). El operador And es una multiplicación lógica que se indica por medio de un paréntesis, un punto o simplemente poniendo juntas las variables que se multiplican, por ejemplo el producto de A y B se expresa como $(A)(B) = A \cdot B = AB$; el Or es una suma lógica que se indica con el signo +; y el operador Not es el complemento o negación de una señal que se indica por un apostrofo ($'$). En la siguiente expresión se muestra la forma en que se representan los operadores:

$$F = A'BD1 + AB'CD + 0$$
$$= A' \wedge B \wedge D \wedge 1 \vee A \wedge B' \wedge C \wedge D \vee 0$$

e) Es posible obtener el valor de una expresión booleana sustituyendo en cada una de las literales el valor de 0 o 1, teniendo en cuenta el comportamiento de los operadores lógicos. En las siguientes tablas se muestra la manera en la que se aplica esta propiedad:

And

A	B	$A \wedge B = AB$
1	1	1
1	0	0
0	1	0
0	0	0

Or

A	B	$(A \vee B) = A + B$
1	1	1
1	0	1
0	1	1
0	0	0

Not

A	A'
1	0
0	1

Hay que tener presente que en álgebra booleana:

$$1 + 1 = 1$$
$$1 + 1 + 1 = 1$$
$$0 + 1 = 1$$
$$0 + 0 = 0$$

ya que el valor máximo es 1.

f) Además de las operaciones básicas, también es posible aplicar la ley de De Morgan de forma semejante a como se aplica en teoría de conjuntos. El siguiente ejemplo muestra la aplicación de esta propiedad:

$$(ABCD)' = A' + B' + C' + D'$$
$$(A + B + C + D)' = A'\,B'\,C'\,D'$$

Teoremas del álgebra booleana

A partir de las propiedades de las operaciones del álgebra booleana se pueden demostrar los siguientes teoremas.

1) **Teorema 1. Idempotencia.**

   $x + x = x$ $\quad x \cdot x = x$

2) **Teorema 2. Identidad de los elementos 0 y 1.**

   $x + 1 = 1$ $\quad x \cdot 0 = 0$

3) **Teorema 3. Absorción.**

   $x + (x \cdot y) = x$
   $x \cdot (x + y) = x$

4) **Teorema 4. Complemento de 0 y 1.**

   $0' = 1$ $\quad 1' = 0$

5) **Teorema 5. Involución.**

   $(x')' = x$

6) **Teorema 6. Leyes de Morgan.**

   $(x + y)' = x' \cdot y'$ $\quad (x \cdot y)' = x' + y'$

5.4 Optimización de expresiones booleanas

Cuando se plantea un problema, en general la expresión booleana obtenida no necesariamente es la óptima, esto es, la más fácil, clara y sencilla de implementar utilizando compuertas lógicas. La expresión que resulta del planteamiento del problema puede ser simplificada empleando para ello teoremas y postulados del álgebra booleana o bien mapas de Karnaugh.

5.4.1 Simplificación de expresiones booleanas mediante teoremas del álgebra de Boole

Los teoremas que se van a utilizar se derivan de los postulados del álgebra booleana, y permiten simplificar las expresiones lógicas o transformarlas en otras que son equivalentes. Una expresión simplificada se puede implementar con menos equipo y su circuito es más claro que el que corresponde a la expresión no simplificada.

A continuación se presenta una lista de teoremas, cada uno con su "dual".

Tabla 5.1 Teoremas del álgebra de Boole

Número	Teorema	Dual
1a.	$0A = 0$	$1 + A = 1$
2a.	$1A = A$	$0 + A = A$
3a.	$AA = A$	$A + A = A$
4a.	$AA' = 0$	$A + A' = 1$
5a.	$AB = BA$	$A + B = B + A$
6a.	$ABC = A(BC)$	$A + B + C = A + (B + C)$
7a.	$(AB...Z)' = A' + B' +...+ Z'$	$(A + B +...+ Z)' = A'B'...Z'$
8a.	$AB + AC = A(B + C)$	$(A + B)(A + C) = A + BC$
9a.	$AB + AB' = A$	$(A + B)(A + B') = A$
10a.	$A + AB = A$	$A(A + B) = A$
11a.	$A + A'B = A + B$	$A(A' + B) = AB$
12a.	$CA + CA'B = CA + CB$	$(C + A)(C + A' + B) = (C + A)(C + B)$
13a.	$AB + A'C + BC = AB + A'C$	$(A + B)(A' + C)(B + C) = (A + B)(A' + C)$

En esta tabla *A* representa no sólo una variable, sino también un término o factor, o bien una expresión.

Para obtener el "dual" de un teorema se convierte cada 0 (cero) en 1 (uno) y cada 1 (uno) en 0 (cero), los signos más (+) se convierten en paréntesis

puntos o simplemente no se ponen, y los puntos en signos más (+). Además de esto, las variables no se complementan ya que al hacerlo se obtendría el complemento en lugar del dual.

Por otro lado, los teoremas 1 a 4 se aplican en cualquier caso y los teoremas 5 a 9 son propiedades que tiene el álgebra booleana, semejantes a las reglas de conjuntos correspondientes a las propiedades conmutativa, asociativa y de De Morgan. Por lo general los teoremas 11 a 13 se aplican en combinación, dependiendo de la expresión booleana.

La aplicación de los teoremas es muy sencilla: simplemente se comparan partes de la expresión con los teoremas que permitan hacer más simple la expresión, y esto se realiza hasta que ya no sea posible simplificar.

Ejemplo 5.2. Para simplificar la expresión booleana

$$F = A'B + (ABC)' + C(B' + A)$$

los teoremas de la tabla 5.1 se aplican de la siguiente manera:

$F = A'B + (ABC)' + C(B' + A)$	
$F = A'B + A' + B' + C' + C(B' + A)$	Después de aplicar 7a.
$F = A'B + A' + B' + C' + CB' + CA$	Por 8a a la inversa
$F = A'B + A' + B' + CB' + C' + CA$	Por 5a.
$F = A'(B + 1) + B'(1 + C) + C' + CA$	Por 8a.
$F = A'1 + B'1 + C' + CA$	Por 1b.
$F = A' + B' + C' + CA$	Por 2a.
$F = A' + B' + C' + A$	Por 11a.
$F = (A + A') + B' + C'$	Por 5a.
$F = (1 + B') + C'$	Por 4b.
$F = 1 + C'$	Por 1b.
$F = 1$	Por 1b.

La expresión booleana en su forma más simple es F = 1, y este resultado indica que si se sustituyen las diferentes combinaciones con los valores binarios 0 o 1 de las variables A, B y C en la expresión inicial, entonces el resultado será siempre igual a 1 (lo que se conoce en lógica matemática como tautología).

En general luego de un proceso de simplificación el resultado no siempre es 1, en cambio lo que se espera es obtener una expresión más simple conformada por menos variables.

Ejemplo 5.3. La simplificación de la expresión booleana

$$F = Z'X + XY'Z + X'Z'W$$

es la siguiente:

$F = Z'X + XY'Z + X'Z'W$	
$F = Z'(X + X'W) + XY'Z$	Por 8a
$F = Z'(X + W) + XY'Z$	Por 11a
$F = Z'X + Z'W + XY'Z$	Por 8a, a la inversa
$F = X(ZY' + Z') + Z'W$	Por 8a
$F = X(Z' + Y') + Z'W$	Por 11a
$F = XZ' + XY' + Z'W$	Por 8a, a la inversa

En los ejemplos anteriores se utilizó un teorema a la vez, y esto se hizo para que no haya confusión en la aplicación de los mismos. Obviamente que cuando ya se tiene suficiente práctica, se pueden aplicar varios teoremas a la vez. Tampoco es necesario indicar qué teorema se usa, sin embargo aquí se hace para ilustrar la simplificación.

Comprensiblemente las expresiones booleanas a simplificar son el resultado del planteamiento de un problema que se busca resolver, tal y como se ilustró al inicio del capítulo con la función booleana

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

Comúnmente este tipo de expresiones booleanas son factibles de ser simplificadas, como se muestra a continuación:

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$
$$F = A'B'D(C' + C) + AB'D(C' + C) + AB'CD'$$
$$F = A'B'D + AB'D + AB'CD'$$
$$F = B'D(A' + A) + AB'CD'$$
$$F = B'D + AB'CD'$$

$F = B'(D + D'AC)$

$F = B'(D + AC)$

$F = B'D + AB'C$

Es conveniente mencionar que con las funciones booleanas se pueden elaborar circuitos equivalentes tanto con la función booleana simplificada como con la que se obtuvo inicialmente, sin embargo el circuito lógico de la función booleana sin simplificar será más grande, complejo y usará más equipo electrónico en su implementación.

5.4.2 Simplificación de expresiones booleanas usando mapas de Karnaugh

El método del mapa de Karnaugh es un procedimiento simple y directo para minimizar las expresiones booleanas, y fue propuesto por Edward W. Veitch y modificado ligeramente por Maurice Karnaugh.

El mapa representa un diagrama visual de todas las formas posibles en que se puede plantear una expresión booleana en forma normalizada. Al reconocer varios patrones se pueden obtener expresiones algebraicas diversas para la misma expresión, y de éstas se puede escoger la más simple, la cual en general es la que tiene el menor número de variables además de que esta expresión posiblemente no sea única.

Las tablas o mapas se dividen en cierto número de casillas, dependiendo de la cantidad de variables que intervengan en la expresión. El número de casillas se puede calcular con la fórmula

$$\text{número de casillas} = 2^n$$

en donde n es el número de variables. Así a una expresión de 2 variables le corresponderá un mapa de 4 casillas, a una de 3 variables un mapa de 8 casillas y así sucesivamente.

Un minitérmino es aquel que forma parte de la expresión y que se puede escribir de la manera más simple formando lo que se conoce en álgebra elemental como un monomio.

Por ejemplo, la expresión

$$F = X'Y + XY$$

consta de dos minitérminos, X'Y y XY, y como se muestra a continuación en las casillas respectivas de la tabla correspondiente se pone un 1 si el minitérmino *se encuentra en la expresión o un 0 si no está:*

Maurice Karnaugh

Fue Ingeniero de telecomunicaciones estadounidense graduado en la universidad de Yale en 1952 y director emérito del ICCC (International Council for Computer Communication). Trabajó como investigador en los Laboratorios Bell desde 1952 a 1966 y en el centro de investigación de IBM de 1966 a 1993, desde 1975 es miembro del IEEE (Institute of Electrical and Electronics Engineers) por sus aportaciones sobre la utilización de métodos numéricos en las telecomunicaciones y es el creador del método tabular o mapa de Karnaugh.

Un mapa de Karnaugh (también conocido como tabla de Karnaugh o diagrama de Veitch, abreviado como K-Mapa o KV-Mapa) es un diagrama utilizado para la minimización de funciones algebraicas booleanas y consiste en una serie de cuadrados cada uno de los cuales representa una línea de la tabla de verdad. Puesto que la tabla de verdad de una función de N variables posee 2N filas, el mapa K correspondiente debe poseer también 2N cuadrados. Cada cuadrado contiene un 0 o un 1, dependiendo del valor que toma la función en cada fila. Las tablas de Karnaugh se pueden utilizar para funciones de hasta 6 variables.

	Y	
X	0	1
0	0	1
1	0	1

Para simplificar la expresión, en la tabla se agrupan los 1 de casillas adyacentes en bloques cuadrados o rectangulares de 2, 4, 8, 16,..., 2^n y se descartan las variables cuyo valor, 1 o 0, cambia de una casilla a otra. La regla es agrupar la información con el menor número posible de bloques ya que de cada uno de éstos se obtiene cuando menos una literal, y los bloques deben estar conformados por el mayor número de casillas porque entre más grande sea el número de casillas agrupadas por bloque, más simple será la expresión booleana resultante.

En el mapa anterior la variable X no conserva su valor ya que en la primera línea vale 0 y en la segunda 1, por lo tanto se elimina. Sin embargo, Y mantiene el valor de 1 en ambas casillas, ya que en este caso el bloque que agrupa la información se encuentra solamente en la columna de la derecha. De esta forma se obtiene que la expresión simplificada del mapa de Karnaugh es F = Y.

Como se ve, la simplificación anterior consiste en la aplicación de los postulados del álgebra booleana, pero de manera gráfica.

Para simplificar una expresión que incluye tres variables se tiene que el mapa consta de 8 casillas. Hay que observar que la secuencia en que se coloca la expresión en la tabla no es la binaria ascendente, sino una de forma que solamente exista un cambio de 0 a 1 o de 1 a 0 a la vez, esto es una en la que no debe cambiar más que un bit en cada paso. A esta forma de arreglar los bits se le llama código reflejado.

Ejemplo 5.4. Representar en un mapa de Karnaugh y determinar la expresión booleana simplificada de:

$$F = XY'Z' + XY'Z + XYZ' + X'YZ'$$

La solución es la siguiente:

	YZ			
X	00	01	11	10
0				1
1	1	1		1

En este caso se forman dos bloques, mismos que permiten eliminar una variable en cada uno de ellos de forma que la expresión simplificada es:

$$F = XY' + YZ'$$

En general se tiene que cuando el número de variables que integran la expresión booleana es impar, el número de filas del mapa es menor que el número de columnas. También es conveniente ordenar las variables alfabéticamente colocando las primeras variables como filas y las restantes como columnas.

Ejemplo 5.5. Como se muestra en el siguiente mapa, un 1 de una celda puede estar contenido en más de un bloque.

	YZ			
X	00	01	11	10
0		1	1	
1		1	1	1

En el caso de esta tabla se tiene que la expresión booleana sin simplificar es:

$$F = X'Y'Z + X'YZ + XY'Z + XYZ + XYZ'$$

la cual ya simplificada queda como:

$$F = Z + XY$$

En el ejemplo anterior se formaron dos bloques, y en el mayor se eliminaron las variables X, Y debido a que de una casilla a otra cambian de valor. Además se observa que entre más grande sea el bloque, la expresión resultante es menor.

Si en un mapa de Karnaugh se unen los dos extremos, ya sea horizontal o verticalmente, entonces las celdas de las esquinas del mismo quedarán juntas y por lo tanto se considerarán como celdas adyacentes. Esto permite realizar una mejor simplificación.

Ejemplo 5.6. Simplificar la siguiente expresión booleana:

$$F = W'X' + W'XY'Z + W'XYZ + WXY'Z' + WX'Y'Z' + WX'YZ'$$

Como se ve, no siempre la expresión original tiene todas las variables en cada uno de sus minitérminos. En donde es así, el minitérmino equivale a las variables que se dan inicialmente, en este caso W'X' juntamente con todas las posibles combinaciones de las variables faltantes:

$$W'X' = W'X'YZ + W'X'Y'Z + W'X'Y'Z' + W'X'YZ'$$

Después se colocan los unos en las celdas correspondientes y se procede a realizar la agrupación y simplificación de los bloques.

	YZ			
WX	00	01	11	10
00	1	1	1	1
01		1	1	
11	1			
10	1			1

Hay que observar cómo cada uno de los bloques tiene cuando menos un 1 que es exclusivo de él. Además se tienen dos bloques de 4 celdas adyacentes, uno de ellos enmarcado en un cuadrado mientras que al otro lo conforman las esquinas del mapa, y en cada uno de ellos se eliminan 2 variables. Aparte de esto, se tiene un pequeño bloque de dos celdas.

La función simplificada queda como sigue:

$$F = \underline{X'Z'} + \underline{W'Z} + \underline{WY'Z'}$$

Del bloque de las esquinas

Del cuadrado de 4 celdas

Del bloque de 2 celdas

Ejemplo 5.7. Usando mapas de Karnaugh es posible simplificar la expresión booleana

$$F = A'B'C'D + A'B'CD + AB'C'D + AB'CD + AB'CD'$$

que resultó del problema de la embotelladora planteado al principio del capítulo.

En este caso se tiene la siguiente tabla:

	CD			
AB	00	01	11	10
00		1	1	
01				
11				
10		1	1	1

La expresión simplificada es:

$$F = B'D + AB'C$$

Ejemplo 5.8. Simplificar la expresión booleana

$$F = A'B'C'D + A'B'C + CD + AB'CD + AB'CD'$$

y obtener la expresión simplificada en sumas de productos y en productos de sumas.

Primero que nada se sabe que:

$$A'B'C = A'B'CD' + A'B'CD$$
$$CD = A'B'CD + A'BCD + ABCD + AB'CD$$

Usando la información, tanto los minitérminos que se complementaron con variables como los inicialmente completos, se tiene el siguiente mapa de Karnaugh:

	CD			
AB	00	01	11	10
00		1	1	1
01			1	
11			1	
10			1	1

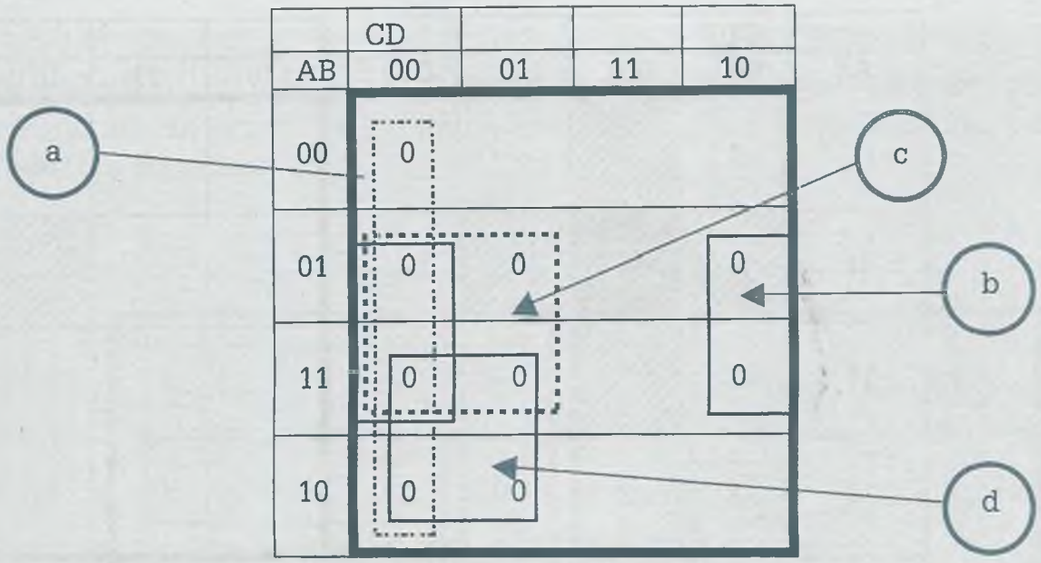
Hay que observar cómo un 1 puede estar considerado en diferentes bloques, como ocurre con el que está en la posición 0011.

En este mapa se tienen nuevamente 3 bloques, 2 de cuatro celdas y 1 de dos. Eliminando los que cambian de valor de una celda a otra se tiene que:

$$F = B'C + CD + A'B'D$$

Ésta es la expresión booleana simplificada en sumas de productos.

En el caso del "producto de sumas" se utiliza el mismo mapa de Karnaugh, pero en las celdas vacías se colocan ceros y se agrupa la información de manera semejante a cuando se tienen unos, como se muestra en el siguiente mapa:



La información se agrupó en este caso en cuatro bloques de 4 celdas cada uno de ellos, y para evitar confusiones en su lectura se le asignó una letra a cada bloque de tal forma que se obtiene la siguiente expresión complemento debido a que se usaron las celdas de ceros y no las de unos:

$$F' = C'D' + BD' + BC' + AC'$$

El asignarle una letra o número a un bloque permite ordenar mejor el resultado de tal forma que el primer término C'D' es la lectura del bloque "a", BD' lo es del bloque "b" y así sucesivamente. El orden en que se asigne la letra no es importante, ya que puede variar de persona a persona.

Complementando ambos miembros de la expresión booleana resulta que:

$$(F')' = (C'D' + BD' + BC' + AC')'$$

Aplicando ahora la ley de De Morgan:

$$F = (C + D)\,(B' + D)\,(B' + C)\,(A' + C)$$

Ésta es la expresión booleana simplificada en productos de sumas.

Hay que observar que no es igual la expresión booleana simplificada en sumas de productos que la que se obtuvo en productos de sumas, sin embargo se puede decir que son lógicamente equivalentes. Esto se puede demostrar usando teoremas del álgebra booleana o bien elaborando las tablas de verdad correspondientes.

A medida que crece el número de variables de la expresión booleana, se hace más complicado el mapa de Karnaugh ya que el número de celdas está dado por 2^n. Un mapa de 5 variables es equivalente a dos mapas de 4, como se muestra a continuación.

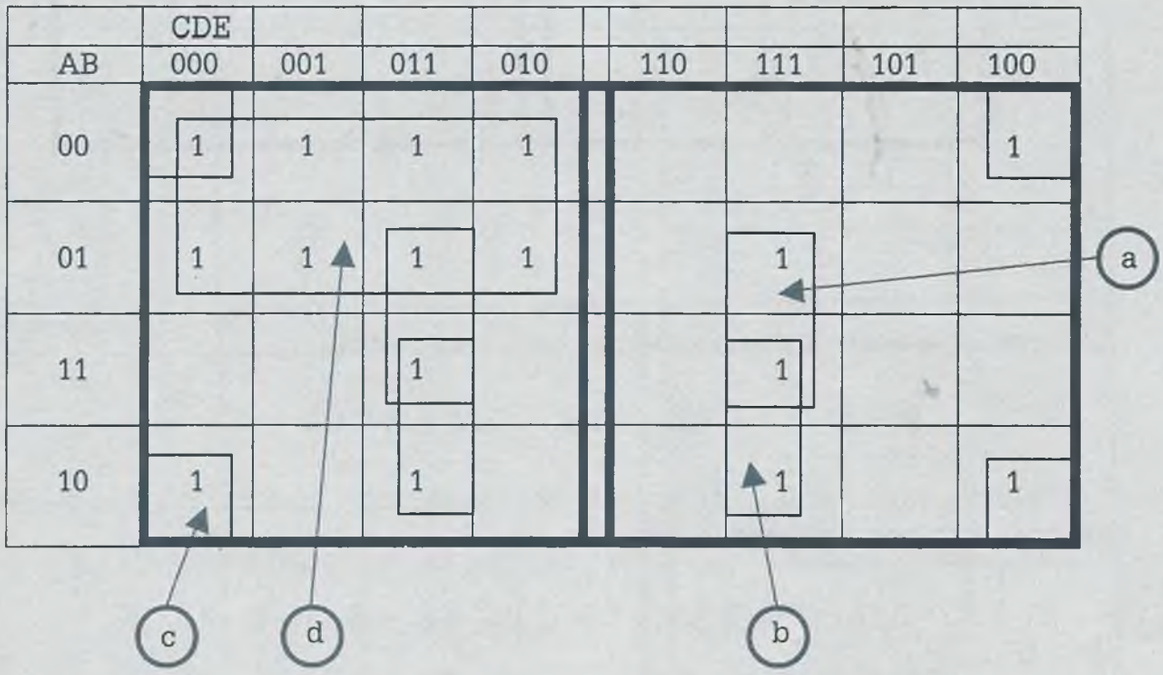
	CDE							
AB	000	001	011	010	110	111	101	100
00	4							
01								
11	1							
10	X	2		3				5

Cuanto crece el mapa, también se ve incrementada la cantidad de celdas adyacentes para agrupar la información. Por ejemplo, en un mapa de 4 variables una celda es adyacente a 4 celdas, mientras que en un mapa de 5 variables cada celda tiene 5 celdas adyacentes y así sucesivamente. En el mapa anterior la celda con sombreado oscuro es adyacente a las 5 celdas con sombreado más claro, la celda con la letra X es adyacente a las celdas numeradas con 1, 2, 3, 4, 5, de tal manera que cada celda se puede agrupar para formar un bloque de dos casillas, con cinco celdas más.

	CDE							
AB	000	001	011	010	110	111	10 1	100
00								
01								
11								
10								

En el mapa anterior el par de celdas con sombreado oscuro se pueden agrupar con las celdas de sombreado claro para formar un bloque de 4 casillas. Obsérvese cómo la frontera entre los dos mapas de 4 actúa como espejo.

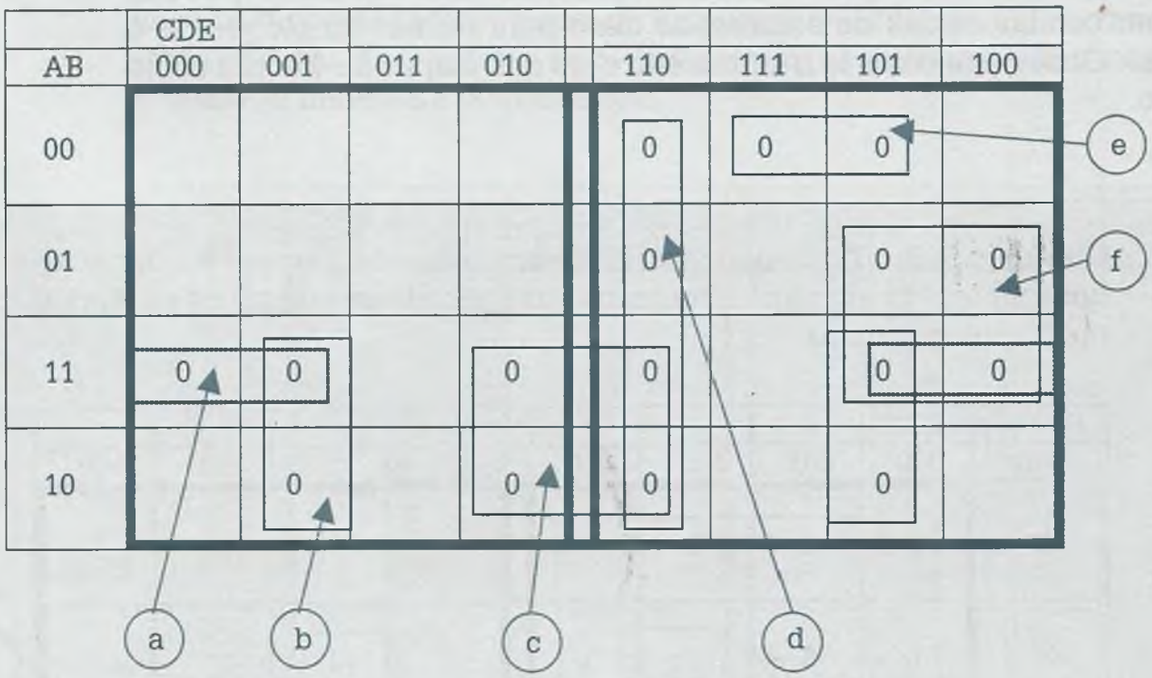
Ejemplo 5.9. Considérese el siguiente mapa de Karnau, y a partir de él determínese la expresión booleana simplificada en sumas de productos y productos de sumas.



Primero se tiene que la expresión booleana simplificada en sumas de productos es:

$$F = BDE + ADE + B'D'E' + A'C'$$

Para obtener la expresión booleana simplificada en productos de sumas se ponen ceros en las celdas vacías, se agrupa la información en bloques y se hace la lectura correspondiente.



La expresión booleana que se lee a partir de la tabla es:

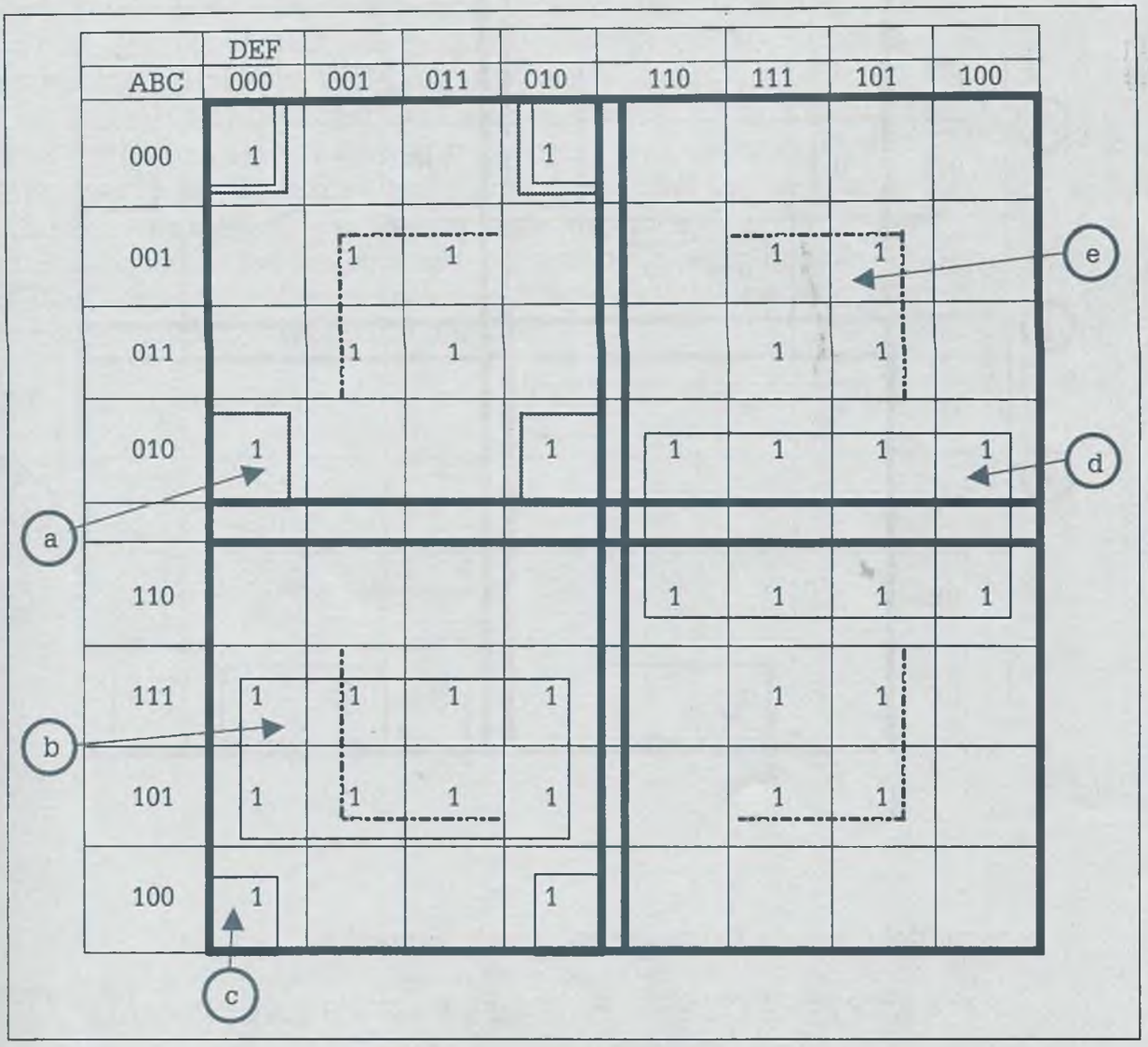
$$F' = ABD' + AD'E + ADE' + CDE' + A'B'CE + BCD'$$

Complementando ambos miembros de la igualdad y aplicando la ley de De Morgan se tiene finalmente que:

$$F = (A' + B' + D)(A' + D + E')(A' + D' + E)(C' + D' + E)(A + B + C' + E')(B' + C' + D)$$

El mapa de seis variables se divide en 4 mapas de cuatro variables. Cada una de las celdas es adyacente a 6 casillas con las mismas reglas ya conocidas.

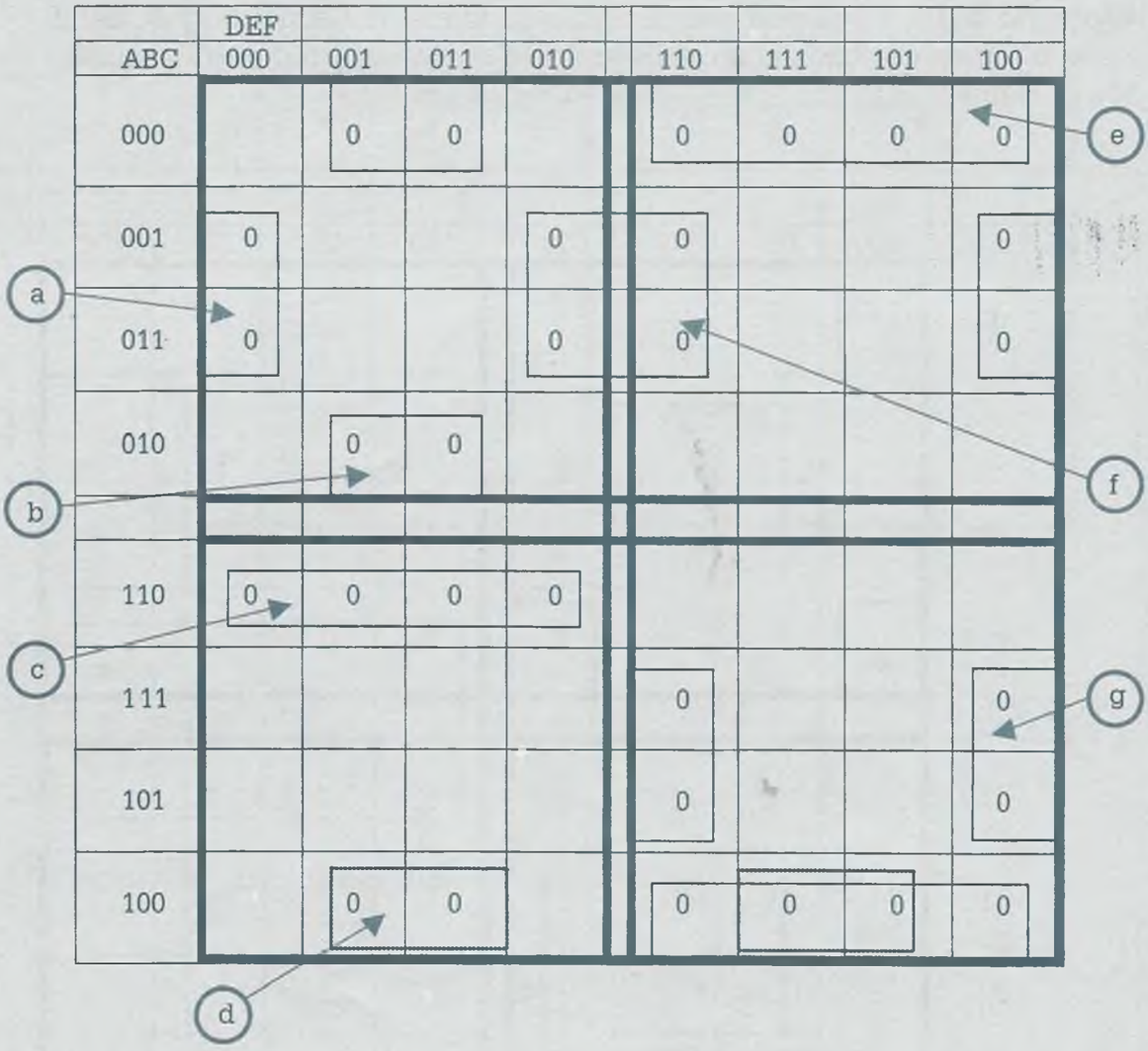
Ejemplo 5.10. Considérese el siguiente mapa de Karnaugh y determínese la expresión booleana más simple en sumas de productos y productos de sumas.



Se tiene que la expresión booleana simplificada en sumas de productos es:

$$F = A'C'D'F' + ACD' + B'C'D'F' + BC'D + CF$$

Para obtener la expresión de productos de sumas se tiene la tabla siguiente:



A partir del mapa de Karnaugh se tiene la expresión

$$F' = A'CE'F' + A'C'D'F + ABC'D' + B'C'F + B'C'D + A'CEF' + ACDF'$$

Complementando ambos miembros de la igualdad y aplicando la ley de De Morgan resulta que

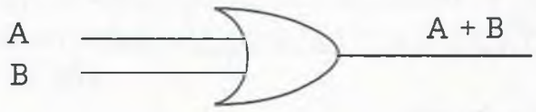
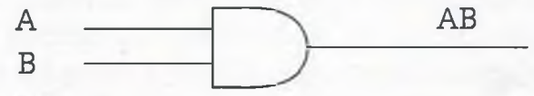
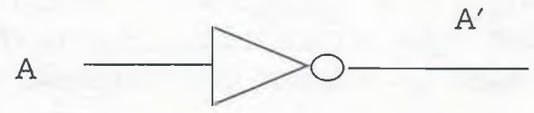
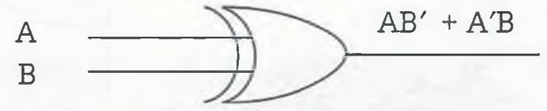
$$F = (A + C' + E + F)(A + C + D + F')(A' + B' + C + D)(B + C + F')$$
$$(B + C + D')(A + C' + E' + F)(A' + C' + D' + F)$$

En algunos mapas de Karnaugh la solución no es única, ya que a veces la información se puede agrupar de manera diferente. Lo que importa al simplificar es obtener la expresión booleana simplificada óptima, independientemente de qué variables son eliminadas. Esto mismo puede suceder con los teoremas del álgebra booleana.

5.5 Compuertas lógicas

Un bloque lógico es una representación simbólica gráfica de una o más variables de entrada a un operador lógico, para obtener una señal determinada o resultado. Los símbolos varían de acuerdo con la rama donde se utilizan, o bien del fabricante. Cada bloque lógico representa un dispositivo que permite manipular la señal según el campo de acción: en mecánica se les llama válvulas (paso del aire o aceite); en electricidad apagadores, contactos (paso de corriente eléctrica); y en electrónica puertas o compuertas (paso de pulsos eléctricos). En este libro sólo se abordarán los símbolos usados en electrónica para la representación de las compuertas, ya que son los que interesan al área de la computación, sin embargo el tratamiento teórico por medio del álgebra booleana es válido para todos ellos independientemente del área.

Tabla 5.2 Compuertas básicas

Compuerta	Símbolo
O (Or)	
Y (And)	
No (Not)	
O-exclusivo (Xor)	

Las compuertas pueden recibir una o más señales de entrada. En la tabla 5.2 A y B son señales que entran a la compuerta y pueden tener un valor de 1 o 0 dependiendo de si existe o no la señal, la cual procede de un sensor o bien de la salida de una compuerta anterior. Esos valores de entrada generan una sola salida, que a su vez también es 0 o 1 dependiendo de la compuerta de que se trate y de los valores de las señales de entrada.

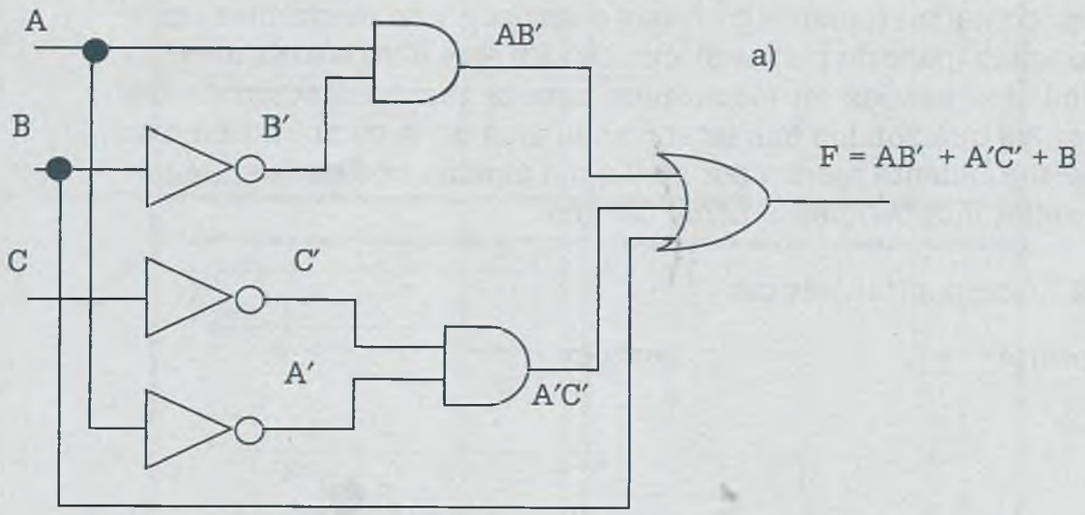
Para representar expresiones booleanas mediante compuertas lógicas es conveniente tener en cuenta las tablas de verdad de las compuertas básicas (operadores lógicos) Or, And y Not vistas en el capítulo de lógica matemática.

Ejemplo 5.11. Representar las siguientes expresiones booleanas usando compuertas lógicas básicas:

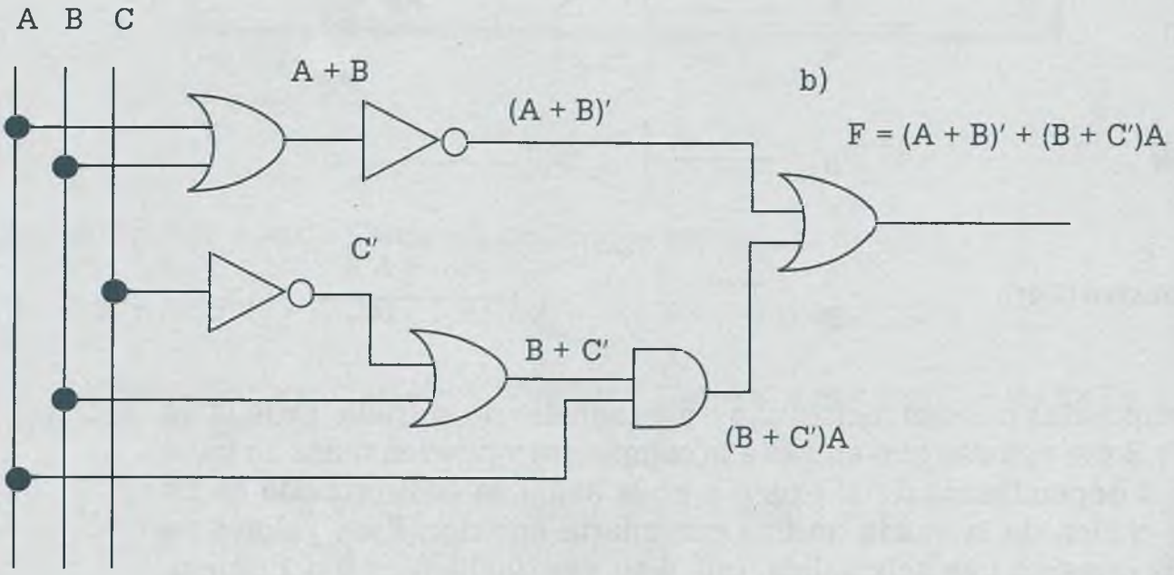
a) $F = AB' + A'C' + B$

b) $F = (A + B)' + (B + C')A$

La representación de (a) es:

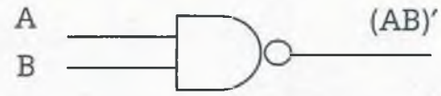


La representación de (b) es:



También existen compuertas lógicas compuestas como Nand y Nor, que son una combinación de los operadores Not y And para la primera y Not y Or para la segunda. En la tabla 5.3 se muestran los símbolos correspondientes.

Tabla 5.3 Compuertas compuestas

Compuerta	Símbolo
Nor	
Nand	
Xnor	

Generalmente los circuitos digitales se construyen con compuertas Nand y Nor, ya que son más fáciles de encontrar en el mercado, son más comunes desde el punto de vista del hardware y están disponibles en la forma de circuitos integrados. Debido a la preferencia de uso de estas compuertas en el diseño de los circuitos, es importante reconocer la relación que existe entre los circuitos construidos con compuertas And, Or y Not y su diagrama equivalente Nand o Nor.

Cuando se simplifica una función el resultado se puede presentar en "sumas de productos" o en "productos de sumas", y en forma natural la presentación en suma de productos permite una implementación usando compuertas Nand mientras que el producto de sumas se puede representar más fácilmente con compuertas Nor, sin embargo es posible implementar cualquier expresión booleana sólo con compuertas Nand o sólo con compuertas Nor.

Ejemplo 5.12. ¿Cuál es el circuito de la expresión booleana

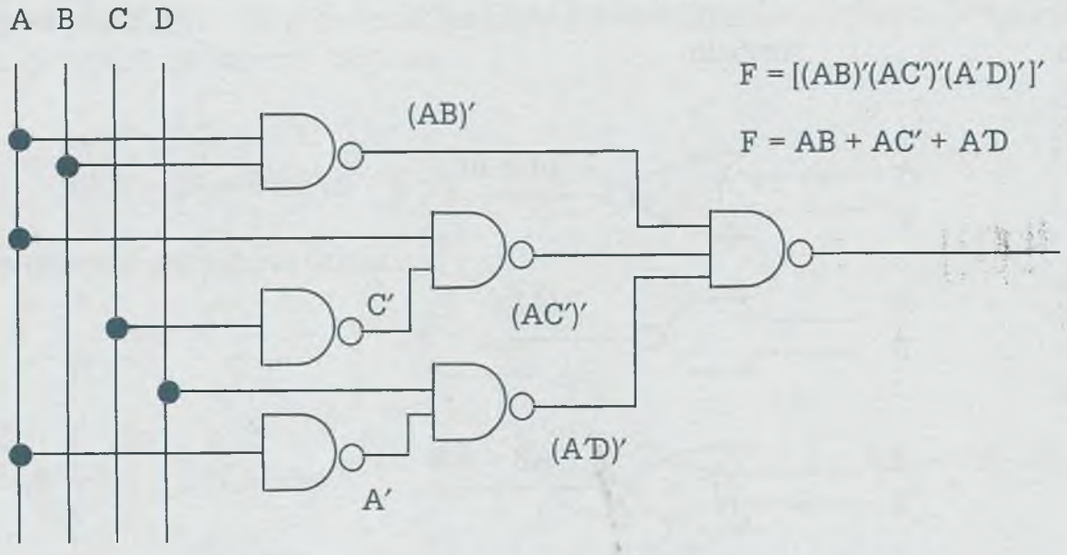
$$F = A(B + C') + A'D$$

hecho sólo con compuertas Nand?

Para obtener el circuito pedido es recomendable llevar la expresión dada a suma de productos:

$$F = AB + AC' + A'D$$

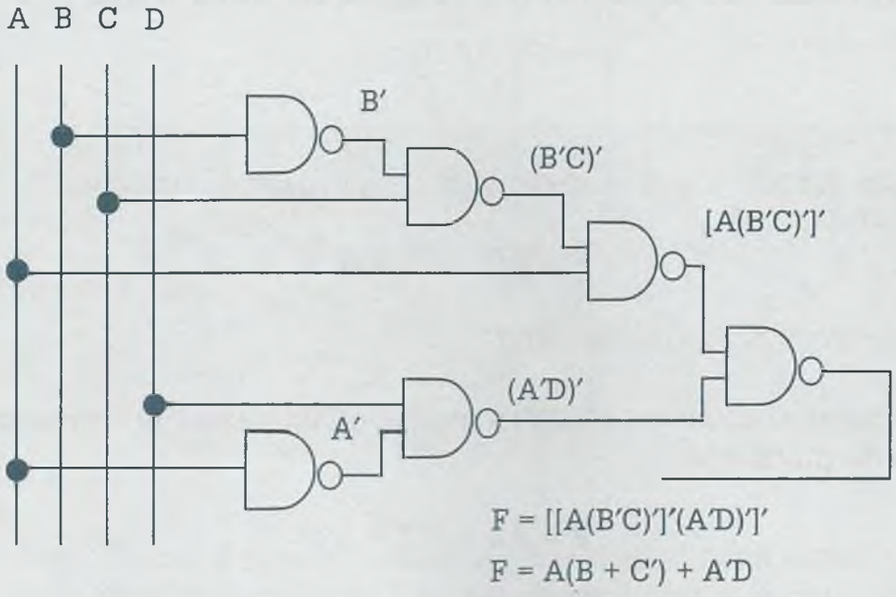
Por lo tanto, el circuito es el siguiente:



Hay que observar que al final se aplicó la ley de De Morgan para quitar la complementación del corchete y obtener el resultado. También se debe destacar que cuando entran dos o más señales a una compuerta Nand primero las multiplica y después complementa dicha multiplicación, pero cuando entra una señal sólo la complementa.

Por otro lado, si no se hubieran hecho las operaciones necesarias para quitar el paréntesis y tener la expresión en sumas de productos, también se podría representar únicamente con compuertas Nand aunque esto algunas veces es un poco más complicado:

$$F = A(B + C') + A'D$$



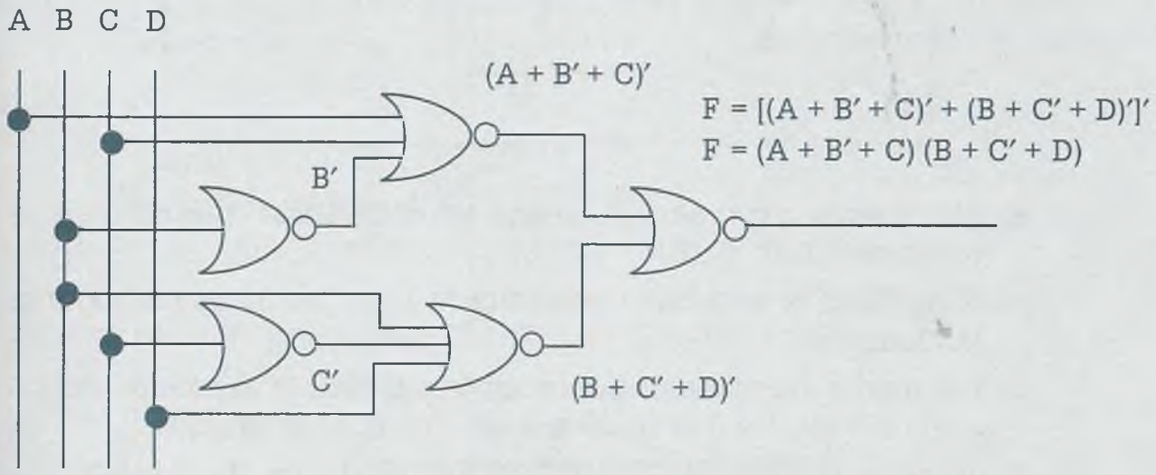
De la misma manera, el bloque lógico Nor facilita su uso cuando la expresión se encuentra dada en productos de sumas.

Ejemplo 5.13. Representar la expresión booleana

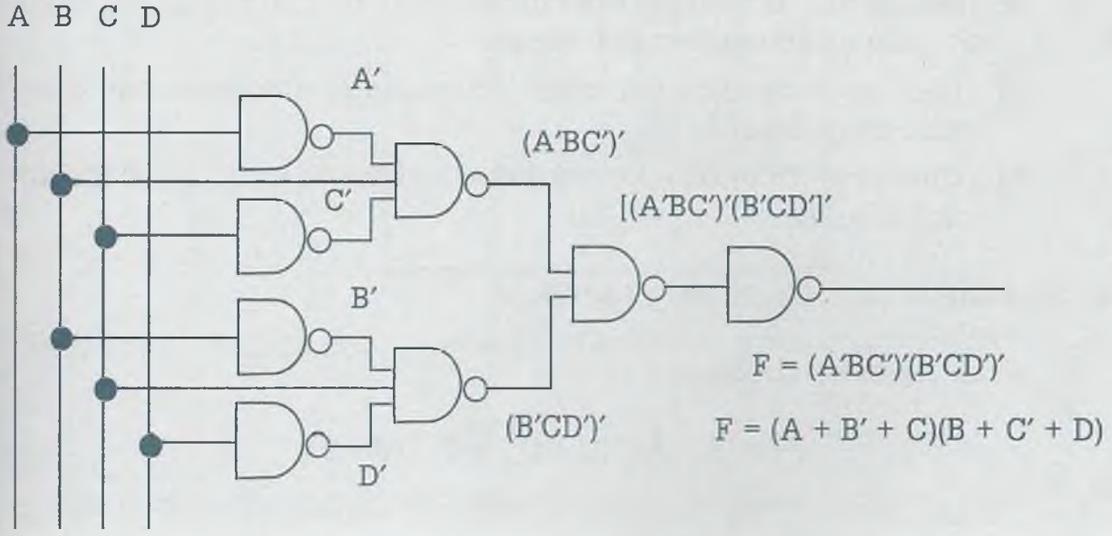
$$F = (A + B' + C)(B + C' + D)$$

usando sólo compuertas Nor.

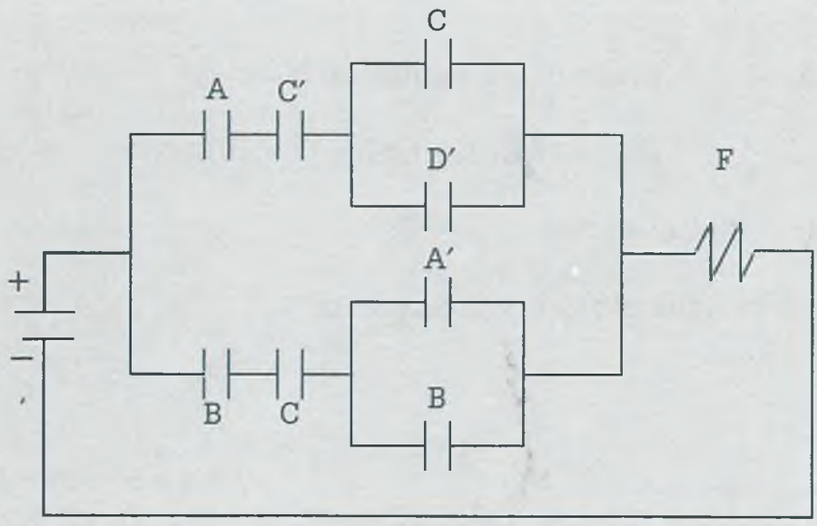
En este caso se tiene el siguiente esquema



La misma expresión booleana representada con compuertas Nand quedaría de la siguiente manera:



Ejemplo 5.14. Considérese el siguiente circuito:



a) ¿Cuál es la expresión booleana sin simplificar que representa dicho circuito?

b) Simplificar la expresión booleana usando teoremas del álgebra booleana.

c) Por medio del mapa de Karnaugh simplificar la expresión del inciso (a) y expresar el resultado en sumas de productos.

d) ¿Cuál es la expresión simplificada en productos de sumas?

e) Comprobar, por medio de una tabla de verdad, que la expresión booleana obtenida en el inciso (c) es lógicamente equivalente a la obtenida en el inciso (d).

f) Representar el resultado del inciso (c) en un circuito lógico, usando para ello compuertas básicas.

g) ¿Cuál es el circuito del inciso (c) basado en compuertas Nand exclusivamente?

h) ¿Cuál es el circuito lógico del inciso (c) basado en compuertas Nor exclusivamente?

La solución de cada inciso es la siguiente:

a) La expresión booleana es

$$F = AC'(C + D') + BC(A' + B)$$

b) Simplificando mediante teoremas resulta que

$F = AC'C + AC'D' + A'BC + BBC$

$F = 0 + AC'D' + A'BC + BC$

$F = AC'D' + BC(A' + 1)$

$F = AC'D' + BC$

c) Se sabe que $AC'C = 0$ y $BBC = BC$, y sustituyendo esto en la expresión $F = AC'C + AC'D' + A'BC + BBC$ resulta que la expresión booleana a representar en el mapa es $F = AC'D' + A'BC + BC$. Aplicando la condición de que para representar un minitérmino en el mapa de Karnaugh éste debe contener todas las letras, a continuación se agregan las variables faltantes con sus posibles combinaciones:

$AC'D' = AB'C'D' + ABC'D'$

$A'BC = A'BCD' + A'BCD$

$BC = A'BCD' + A'BCD + ABCD' + ABCD$

A partir de la información se obtiene el siguiente mapa:

	CD			
AB	00	01	11	10
00				
01			1	1
11	1		1	1
10	1			

Del mapa se obtiene que la expresión booleana en sumas de productos es:

$$F = AC'D' + BC$$

lo cual concuerda con el resultado obtenido usando teoremas.

d) Para obtener el producto de sumas se colocan ceros en las casillas vacías y se agrupa la información:

	CD			
AB	00	01	11	10
00	0	0	0	0
01	0	0		
11		0		
10		0	0	0

A partir del mapa se puede leer que:

$$F' = A'C' + B'C + C'D$$

Complementado y aplicando leyes de De Morgan resulta que:

$$(F)' = (A'C' + B'C + C'D)'$$
$$F = (A + C)(B + C')(C + D')$$

e) Las expresiones boolenas obtenidas en los incisos (c) y (d) son

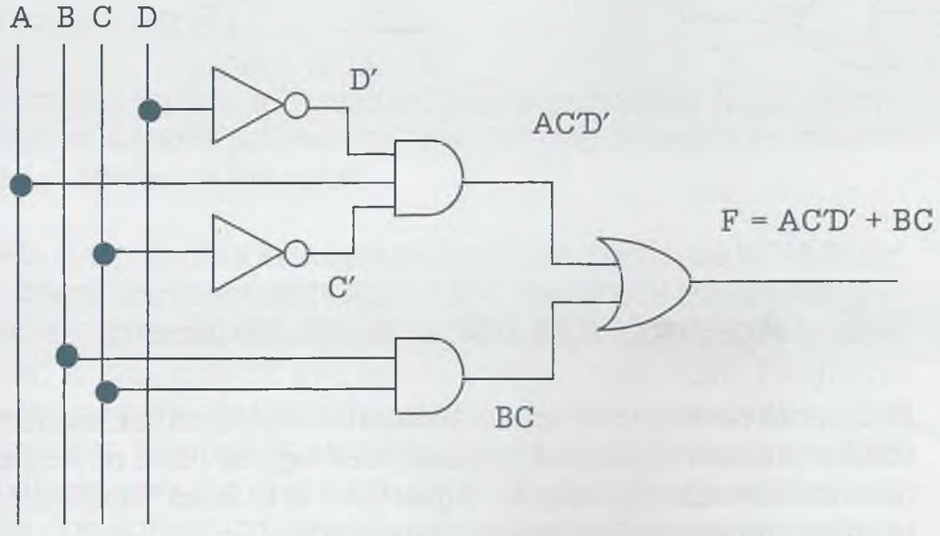
$$F = AC'D' + BC$$
$$F = (A + C)(B + C')(C + D')$$

y a partir de éstas se tiene la siguiente tabla:

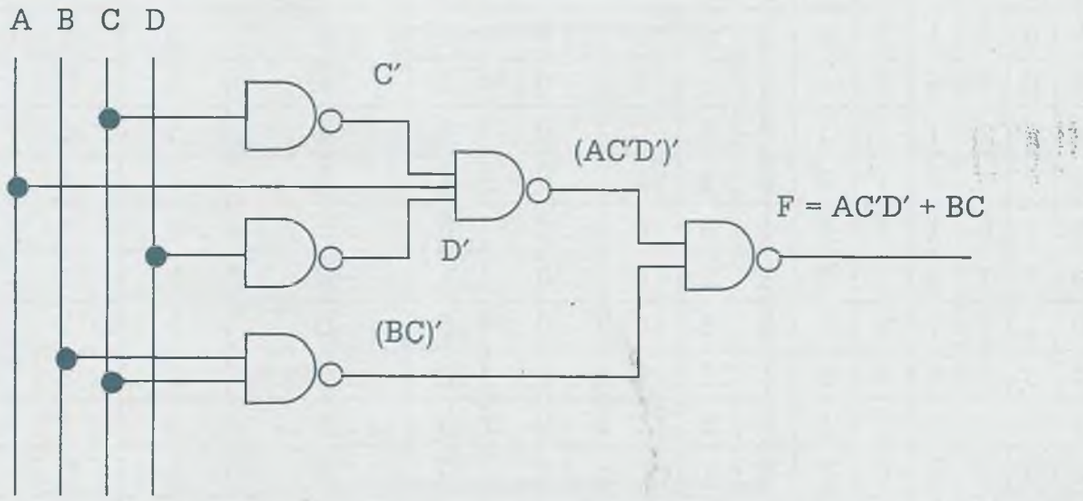
A	B	C	D	A′	B′	C′	D′	AC′	AC′D′	BC	AC′D′ + BC	A + C	B + C′	C + D′	F
0	0	0	0	1	1	1	1	0	0	0	0	0	1	1	0
0	0	0	1	1	1	1	0	0	0	0	0	0	1	0	0
0	0	1	0	1	1	0	1	0	0	0	0	1	0	1	0
0	0	1	1	1	1	0	0	0	0	0	0	1	0	1	0
0	1	0	0	1	0	1	1	0	0	0	0	0	1	1	0
0	1	0	1	1	0	1	0	0	0	0	0	0	1	0	0
0	1	1	0	1	0	0	1	0	0	1	1	1	1	1	1
0	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1
1	0	0	0	0	1	1	1	1	1	0	1	1	1	1	1
1	0	0	1	0	1	1	0	1	0	0	0	1	1	0	0
1	0	1	0	0	1	0	1	0	0	0	0	1	0	1	0
1	0	1	1	0	1	0	0	0	0	0	0	1	0	1	0
1	1	0	0	0	0	1	1	1	1	0	1	1	1	1	1
1	1	0	1	0	0	1	0	1	0	0	0	1	1	0	0
1	1	1	0	0	0	0	1	0	0	1	1	1	1	1	1
1	1	1	1	0	0	0	0	0	0	1	1	1	1	1	1

Aquí se observa que las columnas sombreadas concuerdan en todas sus líneas, por lo tanto esto demuestra que F = AC′D′ + BC es lógicamente equivalente a F = (A + C)(B + C′)(C + D′).

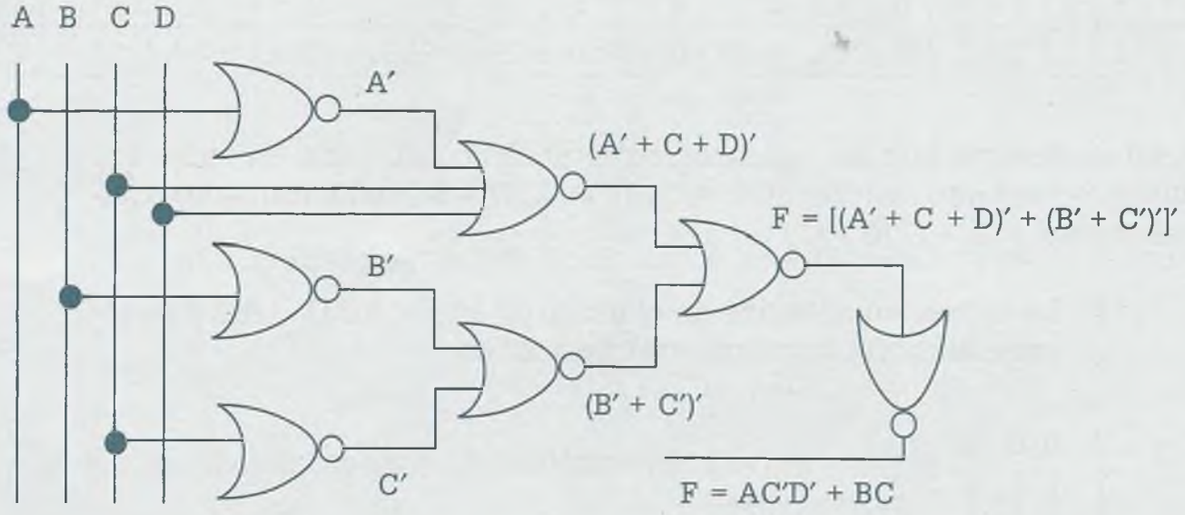
f) La expresión obtenida en el inciso (c) es F = AC′D′ + BC, y su representación con compuertas básicas es:



g) La representación de la expresión booleana F = AC′D′ + BC, sólo con compuertas Nand es la siguiente:



h) La representación de la expresión booleana F = AC′D′ + BC, sólo con compuertas Nor es la siguiente:



5.6 Aplicaciones del álgebra booleana

El álgebra booleana es una extensión de la lógica matemática, ya que uti-
liza los mismos principios y operadores lógicos (and, or, not, xor, nand, no
así como los mismos valores, y gracias a esto John Von Neuman pudo crea
la computadora de la primera generación.

Los dispositivos con los que se implementan las funciones booleanas se llaman "compuertas", y al combinarse han permitido inicialmente la creación del "bulbo", posteriormente la del "transistor" y actualmente la del "chip", elementos con los cuales se construye todo tipo de aparato electrónico digital.

La electrónica digital es una parte de la electrónica que maneja información codificada en dos únicos estados: "falso" y "verdadero", o más comúnmente 0 y 1. Electrónicamente se asigna a cada uno un voltaje o rango de voltaje determinado. Esta particularidad permite que, usando el álgebra booleana y con un sistema de numeración binario, se puedan realizar complejas operaciones lógicas o aritméticas sobre señales de entrada. La electrónica digital ha alcanzado una gran importancia debido a que se utiliza en el diseño de sistemas de automatización, robótica, etc., además de que constituye la piedra angular de las computadoras.

Las computadoras llevan a cabo su trabajo por medio de un microprocesador, el cual es un circuito de alta escala de integración (LSI) compuesto por muchos circuitos simples como flip-flops, contadores, decodificadores, comparadores, etc., todos en una misma pastilla de silicio en donde se utilizan compuertas del álgebra booleana para llevar a cabo las operaciones lógicas.

Las microoperaciones que lleva a cabo el microprocesador se realizan en lenguaje binario a nivel bit. Por ejemplo, si A = 110010, B = 011011 entonces el resultado de llevar a cabo las siguientes operaciones en donde intervienen los operadores lógicos ($\wedge$, $\vee$, $\oplus$, $'$) es:

$$A \wedge B = 110010 \wedge 011011 = 010010$$
$$A \vee B = 110010 \vee 011011 = 111011$$
$$A \oplus B = 110010 \oplus 011011 = 101001$$
$$A' = (110010)' = 001101$$

Basada en el álgebra booleana, la unidad lógica aritmética (ALU: Arithmetic Logic Unit) es la parte del microprocesador que realiza las operaciones aritméticas y lógicas en los datos.

Se sabe que toda computadora está integrada por las memorias ROM (Read Only Memory: Memoria de sólo lectura) y RAM (Random Access Memory: Memoria de acceso aleatorio). Cuando arranca una computadora, ésta debe saber qué hacer, lo cual implica que pueda correr un pequeño programa que le indique lo que debe realizar, qué programas debe ejecutar y en qué lugar debe comenzar. Esta información se guarda en un pequeño programa de sólo lectura que recibe el nombre de ROM, el cual está en lenguaje binario y utiliza operadores lógicos del álgebra booleana para la manipulación de la información. La información en este caso se graba eléctricamente y se borra también de la misma manera. Este tipo de memoria se

John von Neuman
(1903-1957)

Fue un matemático húngaro-estadounidense que realizó contribuciones importantes en física cuántica, análisis funcional, teoría de conjuntos, informática, economía, análisis numérico, hidrodinámica, estadística y muchos otros campos de la matemática.

Fue pionero de la computadora digital moderna, trabajó con Eckert y Mauchly en la Universidad de Pennsylvania y publicó un artículo acerca del almacenamiento de programas. El concepto de programa almacenado permitió la lectura de un programa dentro de la memoria de la computadora, y después la ejecución de las instrucciones del mismo sin tener que volverlas a escribir. La primera computadora en usar el citado concepto fue la llamada EDVAC (Electronic Discrete Variable Automatic Computer), desarrollada por Von Neumann, Eckert y Mauchly. Los programas almacenados dieron a las computadoras flexibilidad y confiabilidad, haciéndolas más rápidas y menos sujetas a errores que los programas mecánicos.

llama Memoria ROM programable eléctricamente (EEPROM). En las computadoras ésta se encuentra en lo que se llama BIOS, la cual es una memoria donde se guarda información de la "tarjeta madre" de los conectores y dispositivos de la PC.

La RAM puede borrarse y grabarse las veces que se desee, la desventaja es que la información grabada en ella sólo se puede utilizar mientras se tenga energía, y se usa como almacenamiento temporal. Existen dos variantes para la memoria RAM: SRAM y DRAM. La SRAM es conocida como memoria estática y en ella los valores binarios o información que se almacena utilizan compuertas del álgebra booleana, por lo que mientras se tenga energía la información en ella se mantendrá intacta. La DRAM es conocida como memoria dinámica y está hecha con celdas que almacenan los datos como cargas en condensadores; la presencia o ausencia de carga en el condensador se interpreta como 1 o 0 binarios, manipulados mediante álgebra booleana. La DRAM es una memoria que requiere refrescarse periódicamente para mantener memorizados los datos, de ahí el nombre de memoria dinámica.

Como se puede ver, la computadora está integrada por elementos que utilizan el álgebra booleana para su desarrollo y funcionamiento. Sin embargo, no es para lo único que se utiliza el álgebra booleana, ya que otra de sus aplicaciones que actualmente está teniendo mucho éxito es la relacionada con la construcción de robots.

Un robot está integrado por elementos mecánicos, eléctricos y electrónicos y el área de conocimiento en este caso es la "mecatrónica". El motor eléctrico es un dispositivo que convierte la energía eléctrica en energía mecánica rotacional, que se utiliza para darle movimiento a los medios de locomoción del robot como son ruedas, brazos y tenazas. El motor puede ser de corriente continua o motor de pasos.

Los medios de locomoción permiten al robot desplazarse de un lugar a otro por medio de ruedas, barras u orugas. Algunos robots deben sostener o manejar objetos y para ello se utilizan tenazas. Algunas veces el movimiento no se proporciona directamente a los medios de locomoción, sino que es necesaria una interfase de transmisión para aumentar la fuerza, reducir la intensidad de giro o cambiar la naturaleza del movimiento (de circular a lineal) por medio de pistones, engranes, levas o poleas.

El funcionamiento de los distintos elementos del robot depende de la señal que se mande de los distintos sensores. Los sensores permiten al robot manejarse con cierta inteligencia al interactuar con el medio, ya que detectan situaciones en las cuales el robot debe llevar a cabo la actividad programada. Entre los diferentes sensores que se utilizan con frecuencia en robots están los sensores ópticos, magnéticos, de ultrasonido, presión, temperatura, nivel e incluso cámaras de video.

Para que el robot lleve a cabo todas las actividades, es necesario el circuito de control (cerebro del robot) que le permita decidir qué hacer cuando se presente una determinada situación. Por ejemplo, qué debe hacer el robot si a su paso se interpone una barrera (girar 90° a la izquierda y avanzar, girar 180° y avanzar, detenerse, etc.). ¿Qué hacer si detecta temperaturas altas (emitir un sonido, parar)? ¿Qué hacer si encuentra un objeto de cierto color (tomarlo y transportarlo)? En fin, todas esas actividades que puede llevar a cabo el robot y para lo cual fue creado, deben estar programadas en el circuito de control y nuevamente el álgebra booleana es la base para el diseño de dicho circuito, el cual se representa inicialmente por medio de una expresión booleana que se simplifica por medio de teoremas del álgebra booleana o mapas de Karnaugh y se implementa usando las compuertas lógicas.

5.7 Resumen

El álgebra booleana es un área de las matemáticas que ocupa un lugar privilegiado, sobre todo por la aplicación de la misma a la computación. Por medio del álgebra booleana es posible diseñar hardware que es la parte fundamental de las computadoras, los robots y todos los sistemas de funcionamiento automático.

Los robots, computadoras o cualquier sistema de funcionamiento automático requieren del uso de elementos mecánicos, eléctricos y electrónicos para llevar a cabo alguna actividad. La forma ordenada en que deben trabajar dichos elementos se controla por medio de un circuito implementado a base de compuertas lógicas.

Cuando se desea que un sistema trabaje de manera automática, primero se representa el funcionamiento de dicho sistema por medio de una expresión booleana. Esta expresión booleana está integrada por variables y cada una de éstas representa la señal de un sensor, la cual puede ser falso o verdadero.

Por lo general la expresión booleana resultante del planteamiento de un problema no es la más simple, sino que tiene variables redundantes que pueden ser eliminadas por medio de:

a) Teoremas del álgebra booleana.
b) Mapas de Karnaugh.

El método para simplificar expresiones booleanas usando teoremas del álgebra booleana consiste en usar éstos para eliminar las variables redundantes hasta obtener una expresión simplificada que realice lo mismo que la expresión inicial que tenía las variables redundantes, pero que al ser más simple el circuito de control es por lo tanto más rápido, económico y eficaz.

El método para simplificar expresiones booleanas mediante mapas de Karnaugh consiste en representar la expresión booleana con ***n*** variables diferentes en una tabla de forma cuadrada o rectangular que tiene 2^n celdas y que recibe el nombre de mapa de Karnaugh. La expresión booleana simplificada es el resultado de agrupar la información de celdas adyacentes en bloques rectangulares o cuadrados de 1, 2, 4, 8,..., 2^n, y después leer la expresión conservando las variables que no cambian de valor de un reglón con respecto a otro o de una columna con relación a otra en cada uno de los bloques en que fue agrupada la información y eliminando las variables que sí sufren un cambio de valor de un renglón con respecto a otro o de una columna con relación a otra.

Por último, esta función booleana simplificada, ya sea por teoremas o mapas de Karnaugh, se representa por medio de símbolos gráficos (bloques lógicos) de cada uno de los operadores lógicos and, or, not, xor, nand, nor y xnor, considerando que las compuertas más comunes son las nand y las nor, mismas que al combinarse permiten suplir las demás compuertas.

5.8 Problemas

5.1. Obtener la tabla de verdad para la siguiente expresión booleana:

F = A′B′C′ + A′B′CD + A′BC + A′BC′D + ABC′ + ABC + AB′D + AB′C′D′

5.2. Obtener la tabla de verdad para cada una de las siguientes expresiones booleanas:

a) F = A′B′C′D′ + A′B′CD′ + A′BC′D + A′BCD + ABC′D′ + ABCD′ + AB′C′D′ + AB′CD′

b) F = (A + BD′)(C′DB + AB′ + DA)′

c) F = [A′(BC + D′)′ + B′A]

5.3. Simplificar las siguientes expresiones booleanas usando los teoremas del álgebra booleana, y verificar los resultados por medio de mapas de Karnaugh.

a) F = A′B′D′ + A′BD′ + A′BD + ABD

b) F = A′CD + ACD + A′B′D + A′B′C + AB′D + AB′CD′

c) F = A′B′C′D′ + A′B′CD′ + A′BC′D + A′BCD + ABC′D′ + ABCD′ + AB′C′D′ + AB′CD′

d) F = A′B′C′D′E + A′B′C′DE + A′B′C′DE′ + A′BC + ABC + ABC′D′E′ + ABC′D′E + ABC′DE + AB′C′D′E + AB′C′DE + AB′CDE + AB′CD′E

e) F = ((A+B)′ + C′ + D′)((AC)′ + (A + (BC)′)′ + D)

f) F = A′B′C′D + A′B′CD + A′B′CD′ + A′BCD + ABCD′ + AB′C′D + AB′CD + AB′CD′

g) F = A′B′C′D′ + A′B′CD + A′B′CD′ + ABC′D + ABCD + ABCD′ + AB′C′D′ + AB′CD + AB′CD′

5.4. Simplificar las siguientes expresiones booleanas usando los teoremas del álgebra booleana y verificar los resultados por medio de mapas de Karnaugh.

a) F = A′B′C′D′ + A′B′CD + A′B′CD′ + A′BC′D + A′BCD + A′BCD′ + ABCD + ABCD′ + AB′C′D′ + AB′CD′

b) F = W′X′Y′Z′ + W′X′YZ + WXY′Z + WXYZ + WX′Y′Z′ + WX′Y′Z + WX′YZ + WX′YZ′ + W′XY′Z′

c) F = W′X′Y′Z + W′XY′Z′ + W′XYZ + W′XYZ′ + WXY′Z + WXYZ + WXYZ′

d) F = A′B′C′D′ + A′B′CD′ + A′BC′D′ + A′BCD′ + ABC′D′ + ABCD + ABCD′ + AB′CD + AB′CD′

e) F = A′B′C′D′ + A′B′CD + A′B′CD′ + A′BC′D + A′BCD + A′BCD′ + ABCD + ABCD′ + AB′C′D′ + AB′CD′

f) F = B′CD + ABC + A′BD′ + ABC′D′ + AB′C′D

g) F = A′BC + BC′D′ + ABC + AB′C′D′ + AB′CD

5.5. En cada uno de los siguientes incisos obtener la expresión booleana simplificada en sumas de productos y en productos de sumas. Plantear el mapa y la agrupación correspondiente.

a)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1						
01	1	1			1	1	1	
11		1	1			1	1	1
10							1	1

b)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1	1	1	1	1	1	
01	1						1	1
11	1	1	1				1	1
10		1			1	1		

c)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1					1	
01			1	1	1	1		1
11	1	1	1	1	1		1	1
10			1				1	

d)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1			1	1
01	1			1				
11	1	1		1			1	1
10	1	1					1	1

e)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1	1			
01		1	1			1		1
11	1		1			1	1	1
10				1	1	1	1	

5.6. En cada uno de los siguientes incisos obtener la expresión booleana simplificada en sumas de productos y en productos de sumas. Plantear el mapa y la agrupación correspondiente.

a)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1				1
01		1	1	1	1	1		
11	1		1	1	1	1	1	1
10				1			1	1

b)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1	1		1	1
01	1			1	1		1	1
11	1		1	1				
10	1			1			1	1

c)

	CDE							
AB	000	001	011	010	110	111	101	100
00			1	1	1	1	1	1
01	1		1	1				
11	1			1				1
10			1	1	1	1		1

d)

	CDE							
AB	000	001	011	010	110	111	101	100
00			1	1			1	1
01		1	1				1	1
11	1				1		1	1
10	1				1		1	1

e)

	CDE							
AB	000	001	011	010	110	111	101	100
00				1		1		1
01	1		1		1		1	
11		1		1		1		1
10	1		1		1		1	

f)

	CDE							
AB	000	001	011	010	110	111	101	100
00	1	1	1	1	1	1	1	1
01								
11								
10	1	1	1	1	1	1	1	1

5.7. Representar con compuertas básicas (And, Or y Not), con compuertas Nand (exclusivamente) y con compuertas Nor (exclusivamente), la expresión lógica:

$$F = AB'C + A'B'D' + AD$$

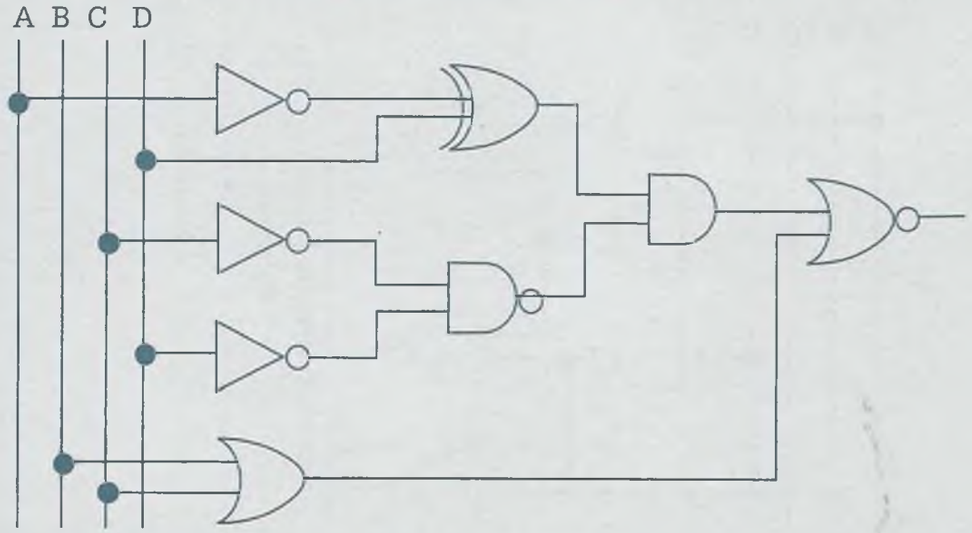
5.8. Representar con compuertas básicas (And, Or y Not), con compuertas Nand (exclusivamente) y con compuertas Nor (exclusivamente), la expresión lógica:

$$F = (B' + C + D')(A + C' + D)B'$$

5.9. Obtener las compuertas Not, And, Or, Nor, X-or y X-nor con base en compuertas Nand exclusivamente.

5.10. Obtener las compuertas Not, And, Or, Nand, X-or y X-nor con base en compuertas Nor exclusivamente.

5.11. Considérese el siguiente circuito lógico:



a) Obtener la función booleana de salida (sin simplificar).

b) Obtener la función booleana simplificada en sumas de productos.

c) Obtener la función booleana simplificada en productos de sumas.

d) Elaborar la tabla de verdad que muestre que las expresiones booleanas obtenidas en los incisos (a) y (b) son lógicamente equivalentes.

e) Implementar el diagrama de la expresión booleana obtenida en el inciso (b) usando exclusivamente compuertas Nand.

f) Hacer el diagrama correspondiente de la expresión booleana obtenida en el inciso (c) usando exclusivamente compuertas Nor.

5.12. En relación con los circuitos de cada uno de los incisos (i) a (v) obtener:

a) La función booleana de salida.

b) La función booleana más simple en sumas de productos.

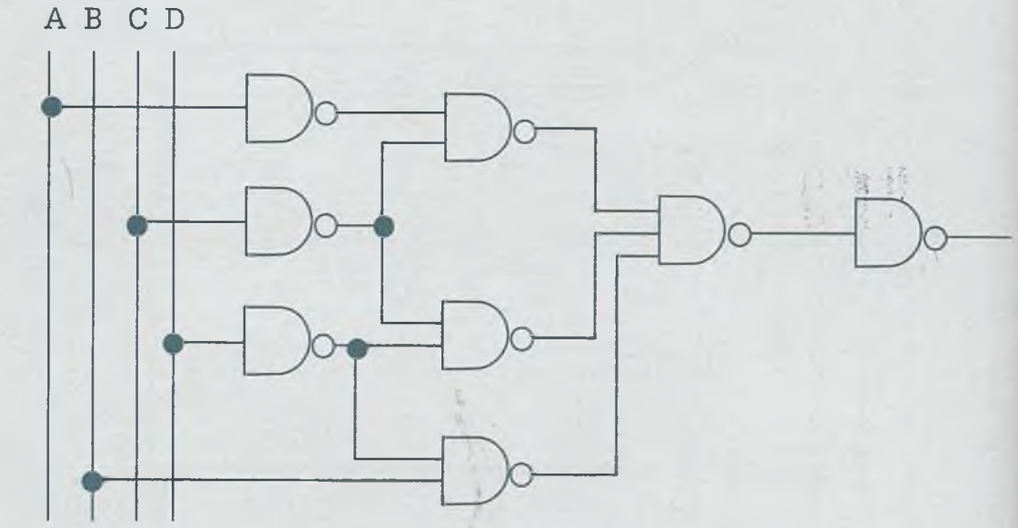
c) La función booleana simplificada en productos de sumas.

d) La tabla de verdad que muestre que las expresiones booleanas obtenidas en los incisos (a), (b) y (c) son lógicamente equivalentes.

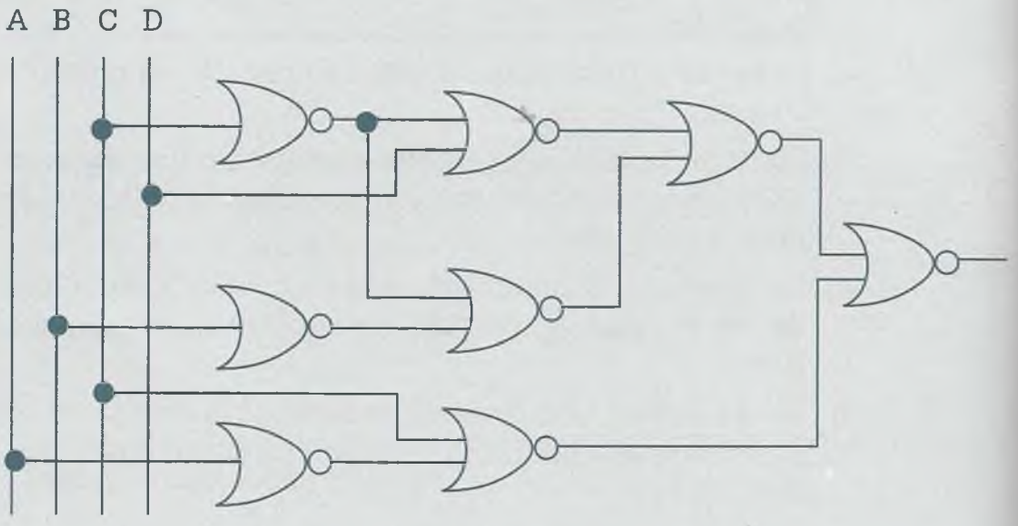
e) El circuito lógico de la expresión booleana del inciso (b), con base en compuertas Nand exclusivamente.

f) El circuito lógico de la expresión booleana obtenida en el inciso (c), a base de compuertas Nor exclusivamente.

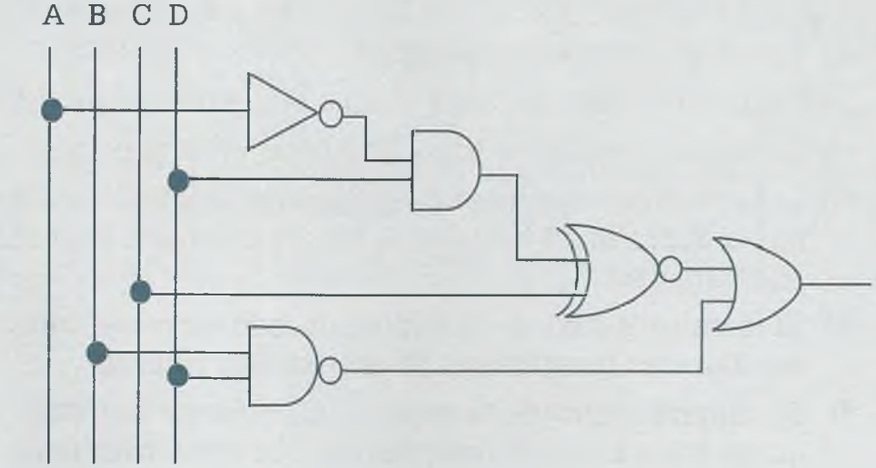
i)



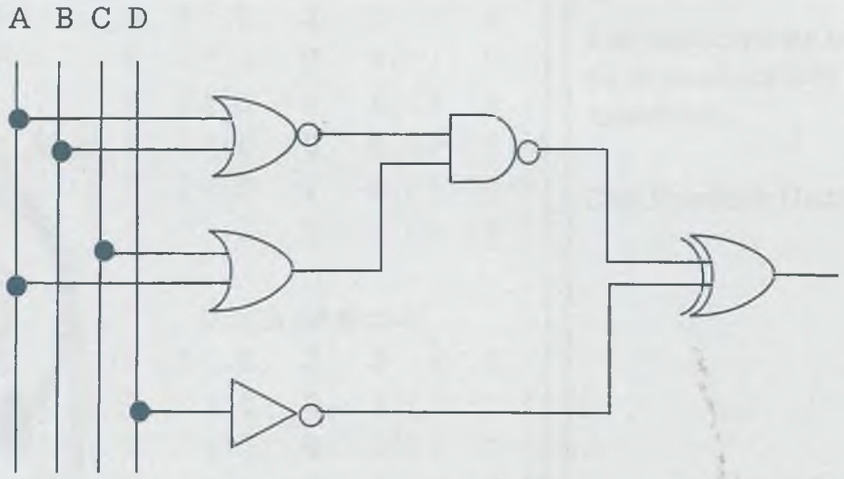
ii)



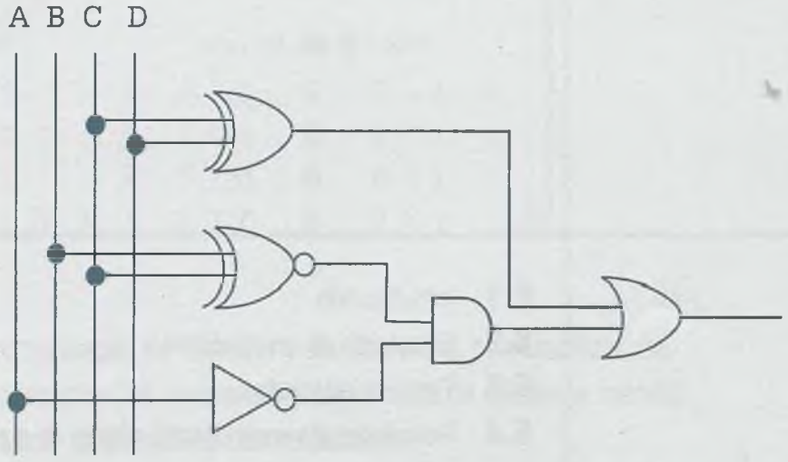
iii)



iv)



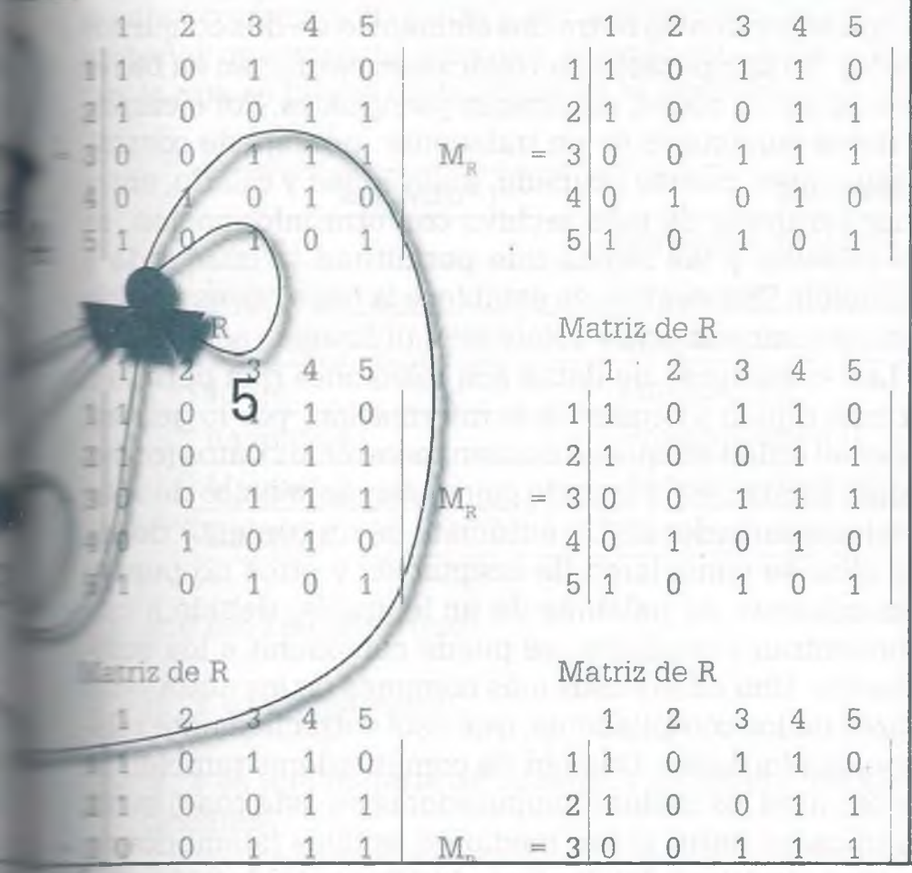
v)



CAPÍTULO

VI

Relaciones



	1	2	3	4	5
1	1	0	1	1	0
2	1	0	0	1	1
3	0	0	1	1	1
4	0	1	0	1	0
5	1	0	1	0	1

M_R =

Matriz de R

	1	2	3	4	5
1	1	0	1	1	0
2	1	0	0	1	1
3	0	0	1	1	1
4	0	1	0	1	0
5	1	0	1	0	1

M_R =

Matriz de R

	1	2	3	4	5
1	1	0	1	1	0
2	1	0	0	1	1
3	0	0	1	1	1
4	0	1	0	1	0
5	1	0	1	0	1

M_R =

Matriz de R

Las matemáticas no se ocupan más que de la enumeración y comparación de relaciones.

Carl Friedrich Gauss

Objetivos

- Comprender el concepto de relación y su diferencia con una función.
- Aprender a representar las funciones y relaciones de diferente manera.
- Aprender a realizar operaciones con relaciones.
- Saber cuáles son las características de las relaciones de equivalencia y la manera en que una relación puede adquirirlas.
- Aplicar los conceptos de relación y función en la computación.

6.1 Introducción

Una relación es una correspondencia entre dos elementos de dos conjuntos con ciertas propiedades. En computación las relaciones se utilizan en bases de datos, estructuras de datos, redes, autómatas y lenguajes. Por ejemplo, se pueden guardar datos personales de un trabajador: número de control, registro federal de causantes, puesto ocupado, antigüedad y salario, entre otros. Para relacionar los datos de este archivo con otra información, se establece el campo relación y las reglas que permitirán la búsqueda y asignación de información. Una vez que se establece la relación, es posible llevar a cabo varias operaciones entre relaciones utilizando para ello el álgebra relacional. Las estructuras de datos son relaciones que permiten acceder de manera más rápida y ordenada la información; por lo general la relación la establece el orden en que se deseen recorrer los datos (orden alfabético, antigüedad, salario, etc.) usando como elemento físico de relación entre los nodos los apuntadores. Un autómata es un conjunto de estados, y algunos de ellos se consideran de aceptación y otros no pero la finalidad es el reconocimiento de palabras de un lenguaje; debido a que estos estados se encuentran vinculados, se puede considerar a los autómatas como una relación. Uno de los usos más comunes de los autómatas se encuentra en el área de los compiladores, que está estrechamente relacionada con los lenguajes formales. Una red de computadoras también se considera una relación: aquí los nodos (computadoras en este caso) están relacionados o comunicados entre sí por medio de señales (alámbricas o inalámbricas). Las redes eléctricas, telefónicas, de agua potable y alcantarillado, también se pueden considerar como relaciones, solamente que en este caso los nodos pueden ser válvulas, bombas, coladeras, lámparas, postes, centrales telefónicas, entre otros, y la relación se establece por medio de tubos, cables y satélites. Pero la forma de tratar las relaciones, independientemente del área del conocimiento, es muy semejante por lo que la información tratada en este capítulo es de gran utilidad.

Las funciones son una clase especial de relación y se utilizan prácticamente en todas las áreas de las matemáticas, en particular en cálculo diferencial e integral, geometría analítica, trigonometría y álgebra. En computación las funciones tienen aplicación directa en lenguajes de programación, ya que cada uno de éstos tiene sus propias librerías de funciones estándar permitiendo al usuario adicionar más funciones con el objeto de hacerlos más ricos, fáciles y poderosos en el momento de programar.

6.2 Elementos de una relación

La definición de relación es la siguiente: dados dos conjuntos no vacíos A y B, una relación R es un conjunto de pares ordenados en donde el primer elemento a está relacionado con el segundo elemento b por medio de cierta propiedad o característica. La relación se indica como aRb:

$$R = \{(a, b) \mid a \in A \text{ y } b \in B\}$$

Una relación es una tabla que muestra la correspondencia de unos elementos con respecto a otros; por ejemplo la relación entre maestros y las materias que imparte cada uno, cumple con las características de relación por lo que se puede representar de la siguiente manera:

Maestro	Materia
Jorge	Sistemas digitales
Domingo	Lenguajes algorítmicos
Ignacio	Estructuras de datos
Jorge	Graficación
Raymundo	Programación II
Manuel	Sistemas operativos
Ezequiel	Sistemas digitales

En este caso se tiene que:

A = {x | x es un maestro}

B = {y | y es una materia de la carrera de ingeniería en sistemas computacionales}

R = {(Jorge, Sistemas digitales), (Jorge, Graficación), (Domingo, Lenguajes algorítmicos), (Ignacio, Estructuras de datos), (Raymundo, Programación II), (Manuel, Sistemas operativos), (Ezequiel, Sistemas digitales)}

Se entiende que el conjunto A está integrado por todos los maestros, aunque no aparezcan en la relación, y que el conjunto B también tiene más materias que las que se consideran en la tabla anterior.

En términos de relación se dice que Jorge está relacionado con Sistemas digitales y Graficación, y que Ignacio está relacionado con Estructuras de datos. Se nota claramente cuáles son los elementos de cada uno de los conjuntos que conforman la relación, pero cada uno de los conjuntos puede tener más información que puede llevar consigo en el momento en que se establece la relación. Por ejemplo, los elementos fundamentales del primer conjunto son los nombres de los maestros, pero a ese conjunto podrían pertenecer campos como código, edad, salario, especialidad, que también son datos propios de cada uno de los maestros, así como en el caso de las materias es típico el número de créditos, código, requisitos, etcétera. Incluso es común en el caso de bases de datos que el campo con el que se establece la relación sea el código de la materia y el código del maestro, y no el nombre de cada uno de ellos.

Las relaciones se forman si se cumple cierta proposición, esa proposición puede ser textual, como en el caso anterior ("Imparten la materia"), pero también puede ser planteada en lenguaje matemático.

Ejemplo 6.1. Sean los conjuntos

$$A = \{a \mid a \in Z;\ 10 < a < 30\}$$
$$B = \{b \mid b \in Z^{+};\ b \leq 20\}$$

y sea R una relación de A en B, en donde el elemento $a \in A$ es divisible entre 13 y $b \in B$ es primo.

Como resultando se obtiene la siguiente relación:

$$R = \{(13, 2), (13, 3), (13, 5), (13, 7), (13, 11), (13, 13), (13, 17), (13, 19), (26, 2), (26, 3), (26, 5), (26, 7), (26, 11), (26, 13), (26, 17), (26, 19)\}$$

Hay que observar que las relaciones también se pueden representar como un conjunto de pares ordenados, en donde el elemento $a \in A$ está relacionado con el segundo elemento $b \in B$, por medio de cierta condición establecida. En este caso la condición es que el primer elemento de los pares ordenados sea un entero entre 10 y 30, divisible entre 13, y el segundo elemento es un entero positivo primo, menor o igual a 20. Otra forma de representar de este conjunto es

$$R = \{(a, b) \mid a \in Z,\ b \in Z^{+};\ a \text{ es divisible entre } 13;\ 10 < a < 30;\ b \text{ es primo};\ b \leq 20\}$$

Si los elementos de un conjunto se pueden relacionar, se dice que los conjuntos que integran la relación están ordenados y a la relación se le llama "relación de orden" en el conjunto. Sin embargo, existen muchos conjuntos cuyos elementos no son comparables. Por ejemplo, en el conjunto de los boxeadores profesionales no es posible tener una pelea entre Julio César Chávez y Mike Tyson (suponiendo que estuvieran activos) debido a que no son del mismo peso, por lo tanto el conjunto de peleas posibles entre boxeadores profesionales es un conjunto parcialmente ordenado, para todos sus pesos, pero no entre las mismas categorías.

6.2.1 Producto cartesiano

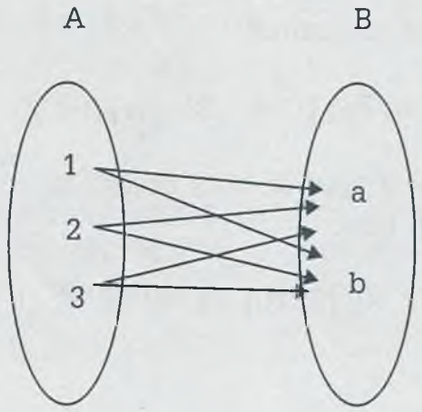
El producto cartesiano de los conjuntos A y B, que se denota como $A \times B$, es la combinación de todos los elementos del conjunto A con todos los elementos del conjunto B. En teoría de conjuntos equivale al conjunto universo.

Una relación R de A en B (R: A → B) es un subconjunto del producto cartesiano A × B. Si R ⊆ A × B y (a, b) ∈ R, entonces a su vez el producto cartesiano también es una relación.

Ejemplo 6.2. Sean los conjuntos

$$A = \{1, 2, 3\} \quad \text{y} \quad B = \{a, b\}$$

El producto cartesiano A × B contiene todos los pares ordenados que resultan de relacionar todos los elementos del conjunto A con todos los elementos del conjunto B, como se muestra en la siguiente figura:



$$A \times B = \{(1, a), (1, b), (2, a), (2, b), (3, a), (3, b)\}$$

La figura del ejemplo 6.2 muestra otra forma de representar una relación: un "diagrama de flechas" en el que se muestran claramente los elementos que pertenecen a cada conjunto, así como la relación entre ellos. También es fácil ver que A × B ≠ B × A.

6.2.2 Relación binaria

No siempre los elementos de la relación son pares ordenados, ya que pueden tener más de dos elementos como en el siguiente caso:

$$R = \{(a, 1, \Delta), (a, 2, \square), (b, 1, \Delta), (c, 3, \square), (c, 2, \Delta)\}$$

Aquí la relación está formada por tercias de elementos pertenecientes a los conjuntos A = {a, b, c}, B = {1, 2, 3} y C = {□, Δ}. En este caso se trata de una relación terciaria y no binaria, ya que los elementos no son pares ordenados sino tercias.

Una de las relaciones más importantes en la computación es la relación binaria, ya que se puede representar por medio de una matriz, tabla o gráfica. Además de ser más fácil de manejar, se le llama relación binaria porque sus elementos son pares ordenados que se forman a partir de dos conjuntos.

En toda relación de pares ordenados no vacía se tienen dos conjuntos: el dominio de R (Dom(R)), que es el conjunto de todos los primeros elementos de los pares de una relación el cual es un subconjunto del conjunto A (Dom(R) $\subseteq$ A), y el codominio de R (Cod(R)), conjunto que está formado por los segundos elementos de los pares de la relación R y que también es un subconjunto de B (Cod(R) $\subseteq$ B).

Ejemplo 6.3. Sean los conjuntos

$$A = \{2, 4, 5, 6, 7, 11\} \quad y \quad B = \{b \mid b \in Z;\ 1 \leq b \leq 10\}$$

Considérese que aRb si y sólo si b es divisible entre a. Por lo tanto, los elementos de la relación son:

$R = \{(2, 2), (2, 4), (2, 6), (2, 8), (2, 10), (4, 4), (4, 8), (5, 5), (5, 10), (6, 6), (7, 7)\}$

$Dom(R) = \{2, 4, 5, 6, 7\}$

$Cod(R) = \{2, 4, 5, 6, 7, 8, 10\}$

6.2.3 Matriz de una relación

Si A y B son dos conjuntos finitos con m y n elementos, respectivamente, y R es una relación de A en B, entonces es posible representar a R como una matriz $M_R = [m_{ij}]$ cuyos elementos se definen como:

$$m_{ij} = \begin{cases} 1 \text{ si } (a, b) \in R \\ 0 \text{ si } (a, b) \notin R \end{cases}$$

Ejemplo 6.4. Sean los conjuntos

$$A = \{1, 2, 3, 4, 5\} \text{ y } B = \{1, 2, 3, 4, 5, 6, 7\}$$

y sea la relación R: A → B tal que

$$R = \{(1, 2), (1, 3), (2, 2), (2, 5), (3, 2), (3, 7), (4, 2), (4, 5), (5, 6)\}$$

Esta relación se puede representar en forma de matriz como sigue:

$$M_R = \begin{array}{c|ccccccc} & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 4 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}$$

Los elementos del conjunto A se representan como filas y los del conjunto B como columnas. Se coloca un 1 si el par ordenado se encuentra en la relación y un 0 en caso contrario.

La representación matricial es muy importante ya que se presta para llevar a cabo las operaciones entre relaciones, sobre todo cuando se tienen relaciones muy grandes.

6.2.4 Grafo de una relación

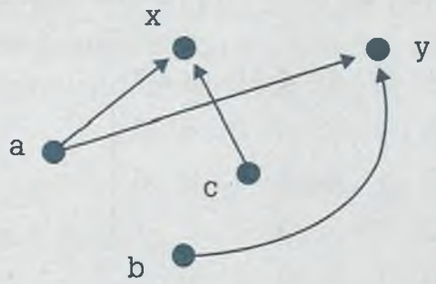
Es posible representar una relación por medio de una gráfica integrada por nodos y flechas, y a este tipo de gráfica se le conoce como "grafo dirigido" de R. Para hacer un grafo sólo se tienen que colocar los elementos de los conjuntos A y B como nodos, y la relación que existe entre los elementos se indica por medio de una flecha que va del elemento del conjunto A al elemento del conjunto B con el que está relacionado.

Ejemplo 6.5. Sean los conjuntos

$$A = \{a, b, c\} \quad y \quad B = \{x, y\}$$

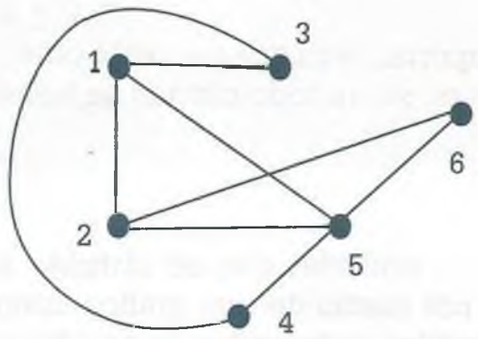
y sea la relación R: A → B tal que

$$R = \{(a, x), (a, y), (b, y), (c, x)\}$$

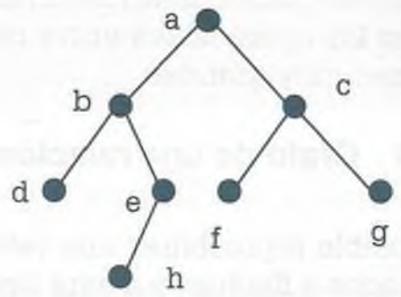


Grafo dirigido

Los grafos pueden ser de dos tipos: "dirigidos", como el del ejemplo 6.5 en el que los nodos están relacionados por medio de una flecha que indica la relación, o "no dirigidos", como el siguiente grafo en el que no existe direccionamiento:



Red



Árbol

Los grafos no dirigidos tienen mucha aplicación tanto en el área de la computación como en los sistemas de comunicación, ya que por medio de un grafo no dirigido es posible representar una red carretera, una red telefónica, una red de computadoras, una red de redes y un árbol, entre otros. En un grafo no dirigido la relación es en ambos sentidos (se considera que las líneas tienen cabezas de flecha en ambos extremos), por lo que no es necesaria la flecha. Este tipo de grafo se expone en el siguiente capítulo.

Es muy común que los conjuntos A y B tengan los mismos elementos. Por ejemplo en el caso A = B = {x | x es un animal} se entiende que tanto A como

B tienen como elementos a todos los animales, con los cuales se pueden establecer diversas relaciones. En este caso se dice que $R \subseteq A \times A$ es una relación de A, en lugar de una relación de A en A.

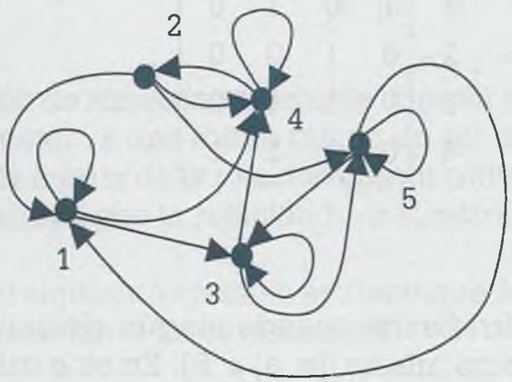
Ejemplo 6.6. Sean los conjuntos

$$A = B = \{1, 2, 3, 4, 5\}$$

y la relación

$$R = \{(1, 1), (1, 3), (1, 4), (2, 1), (2, 4), (2, 5), (3, 3),(3, 4), (3, 5), (4, 2), (4, 4), (5, 1), (5, 3), (5, 5)\}$$

cuyo grafo y representación matricial son los siguientes:



Grafo de R

$$M_R = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 0 & 1 & 1 & 0 \\ 2 & 1 & 0 & 0 & 1 & 1 \\ 3 & 0 & 0 & 1 & 1 & 1 \\ 4 & 0 & 1 & 0 & 1 & 0 \\ 5 & 1 & 0 & 1 & 0 & 1 \end{array}$$

Matriz de R

Se puede observar que cuando A = B, es posible establecer una relación de un elemento a él mismo como ocurre con los pares (1, 1), (2, 2), (4, 4) y (5, 5), y que la matriz de la relación siempre será cuadrada.

6.3 Tipos de relaciones

Las relaciones y funciones deben cumplir con ciertos requisitos para que sean consideradas como tales, y como cada una de ellas tiene sus características propias es posible establecer cierta clasificación. En la siguiente clasificación de relaciones se considera que los conjuntos A y B son iguales, lo que implica que su representación matricial siempre es cuadrada.

6.3.1 Relación reflexiva

Una relación es reflexiva cuando todo elemento de un conjunto A está relacionado consigo mismo, esto es, cuando se cumple que aRa para todo elemento de A. Una característica de este tipo de relación es que su matriz correspondiente contiene unos en toda su diagonal principal y los elementos restantes de la matriz pueden ser unos o ceros, como se muestra en el siguiente ejemplo.

Sean A = B = {1, 2, 3, 4} y

$$R = \{(1, 1), (1, 3), (2, 2), (3, 2), (3, 3), (4, 3), (4, 4)\}$$

Entonces la matriz de esta relación es

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 0 & 1 & 0 \\ 2 & 0 & 1 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 1 & 1 \end{array}$$

6.3.2 Relación irreflexiva

Se dice que una relación es irreflexiva cuando ningún elemento del conjunto A está relacionado consigo mismo ((a, a) $\notin$ R). En este caso la matriz de la relación deberá contener únicamente ceros en la diagonal. Si la diagonal de la matriz tiene ceros y unos, la relación correspondiente no es reflexiva ni irreflexiva.

En el siguiente ejemplo se tiene la matriz de una relación que sólo contiene ceros en su diagonal, por lo tanto ésta es una relación irreflexiva ya que ningún elemento está relacionado consigo mismo.

Sean A = B = {1, 2, 3, 4} y

$$R = \{(1, 3), (1, 4), (2, 4), (3, 2), (4, 3)\}$$

Entonces la matriz de la relación es

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 0 & 0 & 1 & 1 \\ 2 & 0 & 0 & 0 & 1 \\ 3 & 0 & 1 & 0 & 0 \\ 4 & 0 & 0 & 1 & 0 \end{array}$$

6.3.3 Relación simétrica

Se dice que una relación R: A → B es simétrica cuando (a, b) ∈ R y (b, a) ∈ R. Si (a, b) está en la relación pero (b, a) no, entonces la relación no es simétrica.

En el siguiente ejemplo la matriz de esta relación tiene unos o ceros en los pares colocados simétricamente, esto es, si (a, b) ∈ R entonces (b, a) ∈ R. Pero si (a, b) ∉ R entonces (b, a) ∉ R.

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 0 & 1 \\ 2 & 1 & 0 & 1 & 1 \\ 3 & 0 & 1 & 0 & 1 \\ 4 & 1 & 1 & 1 & 0 \end{array}$$

La condición de simetría se debe de cumplir para todos los pares colocados simétricamente, y una forma rápida de saber si la relación es simétrica es comparar la matriz de la relación con su transpuesta: si son iguales entonces se concluye que la relación R es simétrica.

Como en el siguiente ejemplo se tiene que $M_R \neq M_R^{-1}$ entonces se concluye que la relación R no es simétrica:

$$M_R^{-1} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 0 & 1 \\ 2 & 1 & 0 & 1 & 1 \\ 3 & 0 & 1 & 0 & 1 \\ 4 & 0 & 1 & 1 & 0 \end{array}$$

Aquí hay que recordar que M_R^T resulta de convertir las columnas de M_R en filas.

6.3.4 Relación asimétrica

Una relación R de A en B es asimétrica si cuando (a, b) ∈ R entonces (b, a) ∉ R, además de que ningún elemento deberá estar relacionado consigo mismo; esto significa que la diagonal de la matriz de la relación deberá contener solamente ceros.

En relación con los pares simétricos de la siguiente matriz hay que observar que si uno de ellos vale 1, su simétrico debe valer 0. Por otro lado, la

diagonal debe tener solamente ceros, lo cual indica que ningún elemento está relacionado consigo mismo.

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 \\ 4 & 1 & 1 & 1 & 0 \end{array}$$

Los pares colocados simétricamente pueden ser pares de ceros, pero nunca pares de unos.

6.3.5 Relación antisimétrica

Una relación es antisimétrica cuando uno de los pares colocados simétricamente no está en la relación, lo cual significa que (a, b) $\notin$ R o bien (b, a) $\notin$ R. En este caso la diagonal de la matriz no es importante, ya que pueden estar o no relacionados los elementos con ellos mismos.

En la matriz de la relación siguiente, cuando menos uno de los pares simétricos de la relación es 0, lo cual significa que (a, b) $\notin$ R o bien (b, a) $\notin$ R. En la diagonal puede haber ceros o unos, y también puede haber pares de ceros colocados simétricamente y por lo tanto es una relación antisimétrica.

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 1 & 1 \\ 3 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 1 & 0 \end{array}$$

6.3.6 Relación transitiva

Una relación de A en B tiene la propiedad de ser transitiva si cuando aRb y bRc entonces existe el par aRc.

En la matriz de la siguiente relación se tiene (2, 3) y (3, 4), entonces existe (2, 4). También se tiene (3, 1) y (1, 3), entonces (3, 3). De esta forma se deben de revisar todos los posibles pares para ver si se cumple la transitividad.

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 1 & 1 \\ 3 & 1 & 0 & 1 & 1 \\ 4 & 0 & 0 & 1 & 0 \end{array}$$

Luego de llevar a cabo la tabulación correspondiente se concluye que la relación anterior no es transitiva, ya que debe tener los pares ordenados encontrados a continuación:

$$
\begin{aligned}
&(1, 3), (3, 1) \Rightarrow (1, 1)^* \\
&(2, 2), (2, 4) \Rightarrow (2, 4) \\
&(3, 1), (1, 3) \Rightarrow (3, 3) \\
&(4, 3), (3, 1) \Rightarrow (4, 1)^* \\
&(1, 3), (3, 3) \Rightarrow (1, 3) \\
&(2, 3), (3, 1) \Rightarrow (2, 1)^* \\
&(3, 3), (3, 1) \Rightarrow (3, 1) \\
&(4, 3), (3, 3) \Rightarrow (4, 3) \\
&(1, 3), (3, 4) \Rightarrow (1, 4)^* \\
&(2, 3), (3, 3) \Rightarrow (2, 3) \\
&(3, 3), (3, 3) \Rightarrow (3, 3) \\
&(4, 3), (3, 4) \Rightarrow (4, 4)^* \\
&(2, 2), (2, 2) \Rightarrow (2, 2) \\
&(2, 3), (3, 4) \Rightarrow (2, 4) \\
&(3, 3), (3, 4) \Rightarrow (3, 4) \\
&(2, 2), (2, 3) \Rightarrow (2, 3) \\
&(2, 4), (4, 3) \Rightarrow (2, 3) \\
&(3, 4), (4, 3) \Rightarrow (3, 3)
\end{aligned}
$$

Sin embargo, le faltan los 5 elementos marcados con asterisco (*), para que cumpla con la propiedad de transitividad, pero aunque sólo le faltara uno esto sería suficiente para que no fuera transitiva. También es común que se obtengan elementos repetidos, por ejemplo (2, 4) aparece dos veces, pero en este caso solamente se considera uno de ellos y los demás se descartan.

Por lo general las relaciones que se usan en la práctica son muy grandes, por lo que las dimensiones de la matriz también lo son y elaborar un algoritmo para saber si una relación es transitiva por medio de la tabulación requiere mucho tiempo, y el número de iteraciones que debe llevar a cabo la computadora es considerable. Se recomienda que en lugar de esto se desarrolle un algoritmo para multiplicar la matriz booleana M_R por ella misma, para obtener M_R^2. Si $M_R = M_R + M_R^2$ se dice que la relación R es transitiva.

$$
M_R^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 0 & 0 & 1 & 0 \\ 2 & 0 & 1 & 1 & 1 \\ 3 & 1 & 0 & 1 & 1 \\ 4 & 0 & 0 & 1 & 0 \end{array} \odot \begin{array}{|cccc|} 1 & 2 & 3 & 4 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 \end{array} = \begin{array}{|cccc|} 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{array}
$$

Los elementos de la matriz resultante se obtuvieron multiplicando las filas de la primera matriz, por cada una de las columnas de la segunda.

Como se muestra a continuación, para obtener la primera fila de la matriz resultante se multiplica la primera fila de la primera matriz por cada una de las columnas de la segunda:

$$0(0) + 0(0) + 1(1) + 0(0) = 0 + 0 + 1 + 0 = 1$$
$$0(0) + 0(1) + 1(0) + 0(0) = 0 + 0 + 0 + 0 = 0$$
$$0(1) + 0(1) + 1(1) + 0(1) = 0 + 0 + 1 + 0 = 1$$
$$0(0) + 0(1) + 1(1) + 0(0) = 0 + 0 + 1 + 0 = 1$$

Para obtener la segunda fila de la matriz resultante, se multiplica la segunda fila de la primera matriz por cada una de las columnas de la segunda matriz:

$$0(0) + 1(0) + 1(1) + 1(0) = 0 + 0 + 1 + 0 = 1$$
$$0(0) + 1(1) + 1(0) + 1(0) = 0 + 1 + 0 + 0 = 1$$
$$0(1) + 1(1) + 1(1) + 1(1) = 0 + 1 + 1 + 0 = 1$$
$$0(0) + 1(1) + 1(1) + 1(0) = 0 + 1 + 1 + 0 = 1$$

Y así sucesivamente hasta multiplicar todas las filas de la primera matriz por cada una de las columnas de la segunda.

Note cómo los pares ordenados obtenidos por medio de la multiplicación booleana son los mismos que se obtuvieron por medio de tabulación. En este caso se dice que R no es transitiva, ya que $M_R \neq M_R + M_R^2$. El procedimiento para multiplicar dos matrices booleanas es semejante a la multiplicación escalar de matrices. Hay que recordar que para multiplicar dos matrices es necesario que el número de columnas de la primera sea igual al número de filas de la segunda. En este caso no hay ningún problema ya que las matrices de las relaciones son de las mismas dimensiones, de forma que dicha condición se cumple. Se observó también que si la suma de los productos parciales es mayor que 1, se reduce a 1 ya que en álgebra booleana solamente existen ceros y unos. El símbolo $\odot$ significa que se trata de multiplicación de matrices boolenas.

Considerando que A = B, en la tabla 6.1 se resumen las propiedades de las diferentes relaciones.

Tabla 6.1 Propiedades de las relaciones.

Propiedad	Condición
Reflexiva	aRa, $\forall$ a $\in$ A
Irreflexiva	(a, a) $\notin$ R, $\forall$ a $\in$ A
Simétrica	Cuando (a, b) $\in$ R entonces (b, a) $\in$ R, o bien cuando (a, b) $\notin$ R entonces (b, a) $\notin$ R.
Asimétrica	Cuando (a, b) $\in$ R entonces (b, a) $\notin$ R. Además si a = b (a, a) $\notin$ R.
Antisimétrica	(a, b) $\notin$ R o bien (b, a) $\notin$ R. La diagonal no es importante en este caso.
Transitiva	Si (a, b) $\in$ R y (b, c) $\in$ R; entonces (a, c) $\in$ R.

Ejemplo 6.7. Sean los conjuntos A = B = {1, 2, 3, 4} y la relación

$$R = \{(1, 1), (1, 2), (2, 1), (2, 4), (3, 4), (4, 2), (4, 3), (4, 4)\}$$

Determinar si la relación es reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica o transitiva.

Solución. Las respuestas se pueden dar a partir de la matriz de la relación, ya que esto permite mayor claridad.

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 0 & 1 & 1 & 1 \end{array}$$

1) **La relación no es reflexiva** ya que debería tener solamente unos en la diagonal principal, esto es, todos los elementos del conjunto A deberían estar relacionados consigo mismos. Por ejemplo (2, 2) $\notin$ R. Cuando no se cumple con la propiedad, para demostrar esto es suficiente con exhibir un caso.
2) **La relación no es irreflexiva** ya que ningún elemento debería estar relacionado consigo mismo, lo cual significa que la diagonal principal deberá tener solamente ceros. A diferencia de esto se tiene que, por ejemplo (1, 1) $\in$ R.

3) **La relación sí es simétrica** ya que los pares de elementos colocados simétricamente alrededor de la diagonal principal son o bien ceros o unos [el simétrico de (2, 3) es (3, 2) y ambos deben ser ceros o bien unos, pero esto deberá cumplirse para todos los pares colocados simétricamente]. Una forma de saber si una relación es simétrica es por medio de su inversa (R^{-1}). Si $R = R^{-1}$ se dice que la relación es simétrica:

$$R = \{(1, 1), (1, 2), (2, 1), (2, 4), (3, 4), (4, 2), (4, 3), (4, 4)\}$$
$$R^{-1} = \{(1, 1), (2, 1), (1, 2), (4, 2), (4, 3), (2, 4), (3, 4), (4, 4)\}$$

Es más fácil manejar la información por medio de una matriz. En este caso la matriz de la relación deberá ser igual a su inversa ($M_R = M_R^{-1}$).

$$M_R^{-1} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 0 & 1 & 1 & 1 \end{array}$$

Como se observa que $M_R = M_R^{-1}$, se puede concluir que R es una relación simétrica.

4) **La relación no es asimétrica** ya que los pares de elementos colocados simétricamente alrededor de la diagonal deberían ser contrarios, esto es, si uno es cero su contrario debe ser uno. Además la diagonal principal deberá contener solamente ceros. En este caso se tiene por ejemplo que $(1, 2) \in R$ y $(2, 1) \in R$, pero si uno de ellos está contenido en la relación entonces su simétrico no debería estar en ella, sin embargo lo está y esto es suficiente para concluir que la relación R no es asimétrica. Además ningún elemento debería estar relacionado con él mismo, sin embargo no sucede eso ya que por ejemplo $(1, 1) \in R$.

5) **La relación no es antisimétrica** ya que al menos uno de los pares ordenados colocados simétricamente debería ser cero y en la matriz se tiene que $(1, 2) \in R$ y también que $(2, 1) \in R$, lo mismo ocurre con los pares (2, 4) y (4, 2) así como con (3, 4) y (4, 3). Conviene aclarar que no es necesario citar todos los casos para afirmar que la relación dada no es antisimétrica, ya que con un par de pares ordenados en donde no se cumpla la condición es suficiente para concluir que la relación no tiene cierta propiedad. Sin embargo, si se afirma que una relación tiene una propiedad entonces es necesario que se cumpla para todos los pares y no solamente para algunos.

6) **La relación no es transitiva** porque al menos existe un caso en donde no se cumple que si (a, b) ∈ R y (b, c) ∈ R, entonces (a, c) ∈ R. Un ejemplo de esto son los pares ordenados (2, 4) y (4, 2), ya que el par (2, 2) no pertenece a la relación. Esto mismo se pudo haber concluido si se observa que $M_R \neq M_R + (M_R)^2$:

$$M_R^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 0 & 1 & 1 & 1 \end{array} \odot \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 0 & 1 & 1 & 1 \end{array} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 0 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array}$$

$$M_R + M_R^2 = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 0 & 0 \\ 2 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 0 & 1 \\ 4 & 0 & 1 & 1 & 1 \end{array} + \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 0 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ 1 & 1 & 1 & 0 & 1 \\ 2 & 1 & 1 & 1 & 1 \\ 3 & 0 & 1 & 1 & 1 \\ 4 & 1 & 1 & 1 & 1 \end{array}$$

Como $M_R \neq M_R + M_R^2$ entonces la relación R no es transitiva.

6.4 Relaciones de equivalencia, clases de equivalencia y particiones

Una relación de equivalencia es aquella que tiene las tres propiedades: *reflexiva*, *simétrica y transitiva*. Por otro lado, una relación de equivalencia tiene clases de equivalencia y éstas forman particiones. Una partición es un subgrafo completo.

Las clases de equivalencia son conjuntos que contienen a todos los elementos b ∈ B y que están relacionados con a ∈ A. Los elementos del primer conjunto se encierran entre corchetes, de forma que una clase de equivalencia se puede indicar como

$$[a] = \{b \mid b \in B, aRb\}$$

Una partición es un conjunto de clases de equivalencia (conjunto de conjuntos) con las siguientes propiedades:

a) Deberán estar contenidos todos los elementos del conjunto A.

b) La intersección entre las clases de equivalencia deberá ser vacía.

Más formalmente se puede indicar como:

$\lambda = \{[a] \mid a \in A$, la intersección entre clases de equivalencia es vacía$\}$

Ejemplo 6.8. Establecer si la siguiente relación es de equivalencia y plantear el argumento correspondiente.

Sean $A = B = \{1, 2, 3, 4, 5\}$ y

$R = \{(1, 1), (1, 2), (1, 5), (2, 1), (2, 2), (2, 5), (3, 3), (3, 4), (4, 3), (4, 4), (5, 1), (5, 2), (5, 5)\}$

1) Por inspección de R se ve que se cumple que aRa $\forall$ a $\in$ A, esto es, todo elemento del conjunto A está relacionado con él mismo. Otra forma de ver esto es observar que la diagonal principal de la matriz de la relación sólo contiene unos:

$$M_R = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ 1 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 \\ 5 & 1 & 1 & 0 & 0 & 1 \end{array}$$

2) Es una relación simétrica porque para todos los pares simétricos de la relación se cumple que si (a, b) $\in$ R entonces (b, a) $\in$ R. Esto significa que $R = R^{-1}$:

$R = \{(1, 1), (1, 2), (1, 5), (2, 1), (2, 2), (2, 5), (3, 3), (3, 4), (4, 3), (4, 4), (5, 1), (5, 2), (5, 5)\}$

$R^{-1} = \{(1, 1), (2, 1), (5, 1), (1, 2), (2, 2), (5, 2), (3, 3), (4, 3), (3, 4), (4, 4), (1, 5), (2, 5), (5, 5)\}$

O bien por medio de matrices se cumple que $M_R = M_R^{-1}$:

$$M_R = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 \\ 5 & 1 & 1 & 0 & 0 & 1 \end{array} \qquad M_R^{-1} = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 \\ 5 & 1 & 1 & 0 & 0 & 1 \end{array}$$

3) También es una relación transitiva, ya que si (a, b) ∈ R y (b, c) ∈ R entonces (a, c) ∈ R, en todos los casos. Esto se puede observar fácilmente ya que $M_R = M_R + M_R^2$.

$$M_R^2 = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 \\ 5 & 1 & 1 & 0 & 0 & 1 \end{array} \odot \begin{array}{|ccccc|} 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array} = \begin{array}{|ccccc|} 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array}$$

$$M_R + M_R^2 = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 1 & 0 & 0 & 1 \\ 2 & 1 & 1 & 0 & 0 & 1 \\ 3 & 0 & 0 & 1 & 1 & 0 \\ 4 & 0 & 0 & 1 & 1 & 0 \\ 5 & 1 & 1 & 0 & 0 & 1 \end{array} + \begin{array}{|ccccc|} 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array} = \begin{array}{|ccccc|} 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{array}$$

Como se trata de una relación de equivalencia, entonces sus clases de equivalencia son las siguientes:

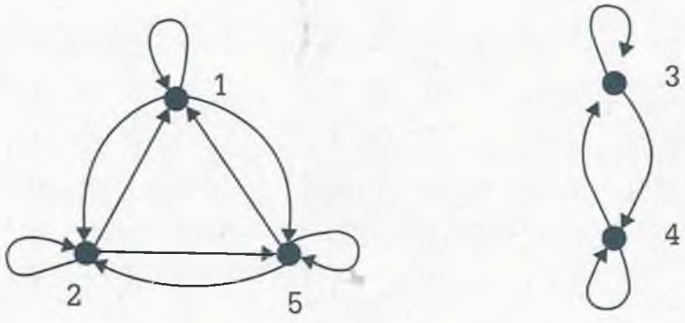
[1] = {1, 2,5}	Todos los elementos que están relacionados con 1.
[2] = {1, 2, 5}	Todos los elementos que están relacionados con 2.
[3] = {3, 4}	
[4] = {3, 4}	
[5] = {1, 2, 5}	

Hay que observar que en ningún caso la clase de equivalencia es vacía, ya que la propiedad reflexiva hace que cuando menos contenga un elemento (a ∈ [a]).

La partición es un conjunto de conjuntos en donde están contenidos todos los elementos de A, pero en donde la intersección de esos conjuntos es vacía. En este caso se tienen dos particiones:

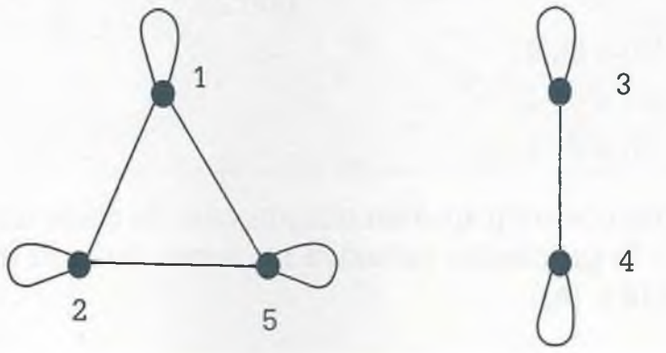
$$\lambda = \{[1], [3]\} = \{[1], [4]\} = \{[2], [3]\} = \{[2], [4]\} = \{[5], [3]\} = \{[5], [4]\}$$
$$\lambda = \{\{1, 2, 5\}, \{3, 4\}\}$$

Una característica importante de las particiones es que el grafo de la relación R está partido en subgrafos completos (de ahí el nombre de partición). En este caso está partido en dos:



Las relaciones de equivalencia son importantes porque es una propiedad que deben tener las redes en el área de computación, en donde la computadora 1 de una red puede enviar información a la computadora 2, pero además la computadora 2 puede enviar información o comunicarse con la computadora 1; ésta es la propiedad de simetría. Por otro lado, toda computadora tiene comunicación consigo misma, con lo cual se cumple la propiedad reflexiva. Así como si existe un camino de comunicación para ir de 1 a 2, (1, 2), y uno para ir de (2, 5), debe haber uno de (1, 5) con lo cual se cumple la propiedad de transitividad.

Debido a que en las redes se tienen grafos completos, ya no se pone la flecha o dirección de relación sino que se sustituyen por una arista sin cabeza de flecha. De esta forma la partición anterior se representa de la siguiente manera:

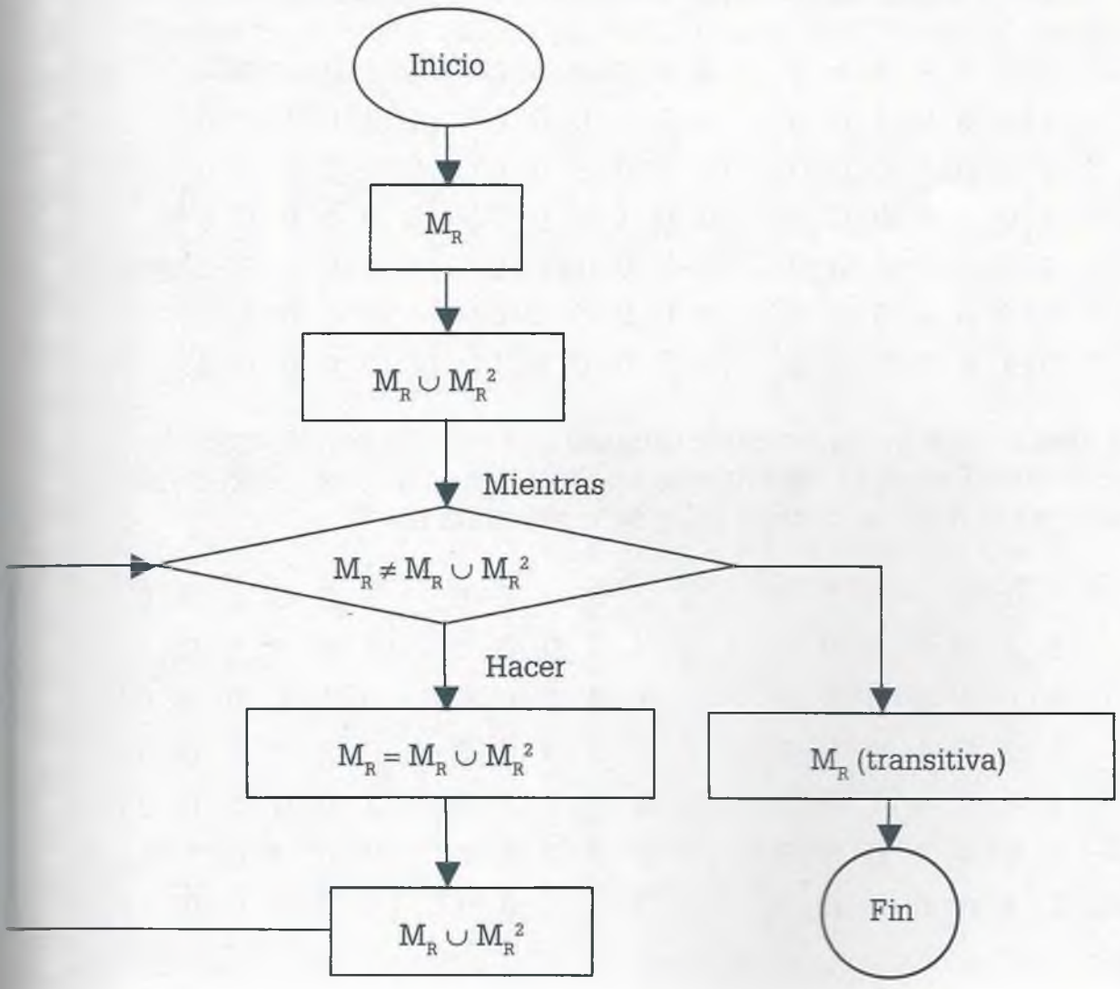


6.4.1 Cerraduras

No todas las relaciones son de equivalencia, pero es posible hacer que tengan esta propiedad agregando los pares ordenados necesarios mínimos para que sean reflexivas, simétricas y transitivas usando para ello las cerraduras. Los diferentes tipos de cerraduras que hay son los siguientes:

- **Cerradura reflexiva**. En este caso se agrega a la relación R la relación identidad para obtener una relación que sea reflexiva ($R \cup I$). La relación identidad (I) es una matriz cuadrada cuyos elementos de la diagonal son únicamente unos y los elementos restantes son ceros.
- **Cerradura simétrica**. A la relación R se le agrega la relación inversa R^{-1} para que la relación resultante tenga la propiedad de simetría, esto es, $R \cup R^{-1}$ o usando matrices $M_R \cup M_R^{-1}$.
- **Cerradura transitiva**. A la relación R se agrega la matriz que resulta de multiplicar la relación por ella misma: $M_R \cup M_R^2$.

Pero algunas veces no es suficiente con adicionar una sola vez M_R^2, sino que requiere considerar a ($M_R = M_R \cup M_R^2$), encontrar nuevamente M_R^2 y aplicar otra vez la cerradura ($M_R \cup M_R^2$) hasta que ($M_R = M_R \cup M_R^2$). El método se ilustra más claramente por medio del siguiente algoritmo.



Ejemplo 6.9. Sean A = B = {1, 2, 3, 4, 5, 6} y R = {(1, 1), (1, 4), (2, 3), (6, 1)}.

a) Establecer si R es una relación de equivalencia.

b) En caso de no ser relación de equivalencia, aplicar las cerraduras correspondientes para hacer que lo sea.

c) Determinar las clases de equivalencia.

d) Determinar la partición, si la tiene.

e) Dibujar el grafo no dirigido de la relación.

Solución

a) No es una relación de equivalencia ya que no es reflexiva, esto es, no cumple con la propiedad de que aRa $\forall$ a $\in$ A; por ejemplo (2, 2) $\notin$ R. Tampoco es simétrica, ya que no cumple con la condición de que si aRb entonces bRa; por ejemplo (2, 3) $\in$ R pero (3, 2) $\notin$ R. No es transitiva, ya que no se cumple que si aRb y bRc entonces aRc; por ejemplo (6, 1) $\in$ R y (1, 4) $\in$ R pero (6, 4) $\notin$ R.

b) Para hacer que sea reflexiva se le debe agregar la relación identidad (I), y esto se consigue mediante la operación ($M_R \cup I$).

$$M_R \cup I = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 2 & 0 & 0 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 0 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 0 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 0 \end{array} \cup \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Con esta operación R ya es reflexiva, puesto que cumple con la condición correspondiente. Para que esta misma relación tenga la propiedad de simetría se toma el resultado como M_R y se le adiciona M_R^{-1}.

$$M_R \cup M_R^{-1} = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 0 & 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \cup \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 2 & 0 & 1 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 1 \end{array}$$

Ahora la relación resultante tiene las propiedades reflexiva y simétrica. Tomando esta matriz como R y adicionando la relación R^2, $(M_R \cup M_R^2)$, se tiene:

$$M_R^2 = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \odot \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{array} = \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array}$$

$$M_R \cup M_R^2 = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 1 \end{array} \cup \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array} = \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array}$$

Siguiendo el algoritmo, mientras $M_R \neq M_R \cup M_R^2$ (cosa que es verdadera para nuestro caso) se debe seguir iterando. Considerar ahora la nueva matriz de la relación como $M_R = M_R \cup M_R^2$ y encontrar M_R^2 con esta nueva matriz como se muestra a continuación:

$$M_R^2 = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 1 & 0 & 1 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 1 & 0 & 1 \end{array} \odot \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array} = \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array}$$

Aplicar la cerradura nuevamente:

$$M_R \cup M_R^2 = \begin{array}{c|cccccc} & 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 1 & 0 & 0 & 0 \\ 3 & 0 & 1 & 1 & 0 & 0 & 0 \\ 4 & 1 & 0 & 0 & 1 & 0 & 1 \\ 5 & 0 & 0 & 0 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 1 & 0 & 1 \end{array} \cup \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array} = \begin{array}{cccccc} 1 & 2 & 3 & 4 & 5 & 6 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 \end{array}$$

Como ya no se cumple que $M_R \neq M_R \cup M_R^2$ por lo tanto este último resultado, que ahora es M_R, ya es transitivo.

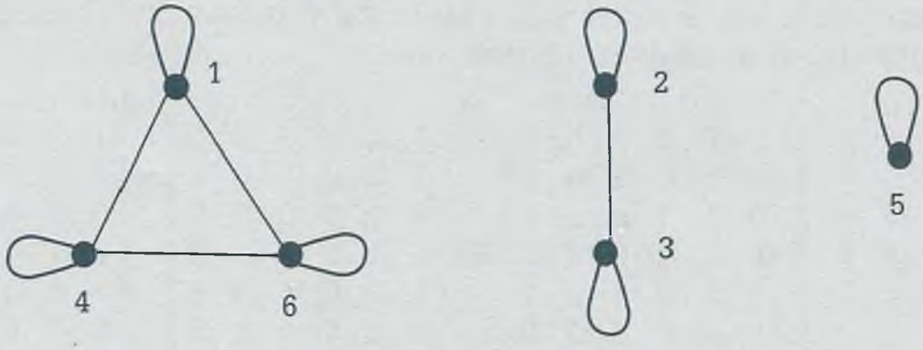
c) La relación resultante ya tiene la propiedad reflexiva, simétrica y transitiva, por lo que es una relación de equivalencia y sus clases de equivalencia son:

$$[1] = \{1, 4, 6\}$$
$$[2] = \{2, 3\}$$
$$[3] = \{2, 3\}$$
$$[4] = \{1, 4, 6\}$$
$$[5] = \{5\}$$
$$[6] = \{1, 4, 6\}$$

d) Su partición está integrada por las clases de equivalencia [1], [2] y [5], ya que ellas contienen todos los elementos del conjunto A y la intersección entre esas clases de equivalencia es vacía.

$$\lambda = \{[1], [2], [5]\} = \{\{1, 4, 6\}, \{2, 3\}, \{5\}\}$$

e) El grafo no dirigido de esta relación de equivalencia está dividido en tres partes, cada una de las cuales corresponde a una clase de equivalencia que integra la partición.



6.5 Operaciones entre relaciones

Así como se pueden realizar operaciones con números también es posible realizar operaciones entre relaciones. Las operaciones que se pueden llevar a cabo con relaciones son: unión, intersección, complemento, composición e inversa de una relación. Estas operaciones se pueden hacer usando matrices o bien con conjuntos.

- **Complemento de R.** Se indica como R′ y contiene todos aquellos pares ordenados que no forman parte de la relación R. Este conjunto incluye a todos los pares ordenados que están en el producto cartesiano $A \times B$

pero que no se encuentran en R. Cuando el complemento se obtiene por medio de matrices, se deben cambiar todos los unos por ceros y los ceros por unos.

- **Intersección.** Sean R y S relaciones de un conjunto A en B, entonces se puede obtener $R \cap S$. En términos de relaciones se puede ver que si $a(R \cap S)b$, entonces aRb y aSb. Si se toma a las relaciones como conjuntos, se sabe que la intersección de dos relaciones contiene a todos los pares ordenados comunes a las relaciones R y S. Si la manipulación es por medio de matrices, $M_{R \cap S}$ es el resultado de multiplicar elemento por elemento las matrices booleanas de R y S.

- **Unión.** La unión de dos relaciones $(R \cup S)$ significa que aRb o bien aSb. Los elementos que están en la unión de dos relaciones son todos los pares ordenados que están en R, que están en S, o que están en ambos. Por medio de matrices se lleva a cabo una suma de matrices booleanas entre M_R y M_S para obtener $M_{R \cup S}$.

- **Inversa.** También es posible obtener la inversa R^{-1} de una relación R. Cuando se trabaja con conjuntos se intercambia la posición de a y b, esto implica que si $(a, b) \in R$ entonces $(b, a) \in R^{-1}$. En el caso de matrices, la inversa de M_R es $M_R{}^{-1}$ que se puede obtener intercambiando filas por columnas en la matriz M_R.

- **Composición.** La composición de relaciones R y S $(R \circ S)$ equivale a la propiedad transitiva, esto significa que si $(a, b) \in R$ y $(b, c) \in S$, entonces $(a, c) \in (R \circ S)$. También es posible obtener la composición de dos relaciones por medio de una multiplicación booleana de las matrices de las relaciones $R \circ S = M_{R \circ S} = M_R \odot M_S$.

Ejemplo 6.10. Sean los conjuntos $A = B = C = D = \{1, 2, 3, 4\}$ y las relaciones R: $A \to B$, S: $B \to C$, T: $C \to D$, tales que

$R = \{(1, 1), (1, 3), (1, 4), (2, 1), (2, 2), (2, 4), (3, 2), (3, 3), (4, 1), (4, 3), (4, 4)\}$
$S = \{(1, 2), (1, 3), (2, 3), (2, 4), (3, 1), (3, 2), (3, 3), (4, 2), (4, 3)\}$
$T = \{(1, 3), (1, 4), (2, 1), (2, 3), (2, 4), (3, 1), (3, 2), (4, 2)\}$

Determinar $(S^{-1} \cap R)' \circ (T \cup R')$ por medio de conjuntos y verificar el resultado usando matrices booleanas.

Solución

Usando conjuntos se tiene que

$S^{-1} = \{(1, 3), (2, 1), (2, 3), (2, 4), (3, 1), (3, 2), (3, 3), (3, 4), (4, 2)\}$
$S^{-1} \cap R = \{(1, 3), (2, 1), (2, 4), (3, 2), (3, 3)\}$

Por otro lado, se sabe que el producto cartesiano $A \times B$ contiene todos los pares ordenados que resultan de relacionar todos los elementos del conjunto A con todos los elementos del conjunto B:

$$A \times B = \{(1, 1), (1, 2), (1, 3), (1, 4), (2, 1), (2, 2), (2, 3), (2, 4), (3, 1), (3, 2), (3, 3), (3, 4), (4, 1), (4, 2), (4, 3), (4, 4)\}$$

Tomando como referencia $A \times B$ se tiene que:

$$(S^{-1} \cap R)' = \{(1, 1), (1, 2), (1, 4), (2, 2), (2, 3), (3, 1), (3, 4), (4, 1), (4, 2), (4, 3), (4, 4)\}$$

$$R' = \{(1, 2), (2, 3), (3, 1), (3, 4), (4, 2)\}$$

$$(T \cup R') = \{(1, 2), (1, 3), (1, 4), (2, 1), (2, 3), (2, 4), (3, 1), (3, 2), (3, 4), (4, 2)\}$$

Para obtener la composición cuando se utilizan conjuntos se deben determinar todos los pares ordenados que se forman al combinar (a, b) con (b, c) para obtener (a, c):

$$(1, 1), (1, 2) \Rightarrow (1, 2)$$
$$(1, 2), (2, 4) \Rightarrow (1, 4)$$
$$(2, 3), (3, 1) \Rightarrow (2, 1)$$
$$(3, 1), (1, 4) \Rightarrow (3, 4)$$
$$(1, 1), (1, 3) \Rightarrow (1, 3)$$
$$(1, 4), (4, 2) \Rightarrow (1, 2)$$
$$(2, 3), (3, 2) \Rightarrow (2, 2)$$
$$(3, 4), (4, 2) \Rightarrow (3, 2)$$
$$(1, 1), (1, 4) \Rightarrow (1, 4)$$
$$(2, 2), (2, 1) \Rightarrow (2, 1)$$
$$(2, 3), (3, 4) \Rightarrow (2, 4)$$
$$(4, 1), (1, 2) \Rightarrow (4, 2)$$
$$(1, 2), (2, 1) \Rightarrow (1, 1)$$
$$(2, 2), (2, 3) \Rightarrow (2, 3)$$
$$(3, 1), (1, 2) \Rightarrow (3, 2)$$
$$(4, 1), (1, 3) \Rightarrow (4, 3)$$
$$(1, 2), (2, 3) \Rightarrow (1, 3)$$
$$(2, 2), (2, 4) \Rightarrow (2, 4)$$
$$(3, 1), (1, 3) \Rightarrow (3, 3)$$
$$(4, 1), (1, 4) \Rightarrow (4, 4)$$
$$(4, 2), (2, 1) \Rightarrow (4, 1)$$

$$(4, 2), (2, 4) \Rightarrow (4, 4)$$
$$(4, 3), (3, 2) \Rightarrow (4, 2)$$
$$(4, 4), (4, 2) \Rightarrow (4, 2)$$
$$(4, 2), (2, 3) \Rightarrow (4, 3)$$
$$(4, 3), (3, 1) \Rightarrow (4, 1)$$
$$(4, 3), (3, 4) \Rightarrow (4, 4)$$

Después de tomar el primer par de $(S^{-1} \cap R)'$ y el otro de $(T \cup R')$, y de llevar a cabo todas las combinaciones, se obtuvo como resultado de la composición el siguiente conjunto:

$(S^{-1} \cap R)' \circ (T \cup R') = \{(1, 1), (1, 2), (1, 3), (1, 4), (2, 1), (2, 2), (2, 3), (2, 4), (3, 2), (3, 3), (3, 4), (4, 1), (4, 2), (4, 3), (4, 4)\}$

Usando matrices se tiene lo siguiente.

Para verificar dicho resultado por medio de matrices, es necesario obtener las matrices de las relaciones R, S y T:

$$M_R = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 1 & 0 & 1 & 1 \\ 2 & 1 & 1 & 0 & 1 \\ 3 & 0 & 1 & 1 & 0 \\ 4 & 1 & 0 & 1 & 1 \end{array} \qquad M_S = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 1 & 1 & 0 \\ 2 & 0 & 0 & 1 & 1 \\ 3 & 1 & 1 & 1 & 0 \\ 4 & 0 & 1 & 1 & 0 \end{array}$$

$$M_T = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 1 & 1 \\ 2 & 1 & 0 & 1 & 1 \\ 3 & 1 & 1 & 0 & 0 \\ 4 & 0 & 1 & 0 & 0 \end{array} \qquad M_S^{-1} = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline 1 & 0 & 0 & 1 & 0 \\ 2 & 1 & 0 & 1 & 1 \\ 3 & 1 & 1 & 1 & 1 \\ 4 & 0 & 1 & 0 & 0 \end{array}$$

$$M_S^{-1}{}_{\cap R} = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 \end{vmatrix} \cap \begin{vmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \end{vmatrix} = \begin{vmatrix} 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{vmatrix}$$

$$M_{(S^{-1}\cap R)'} = \begin{vmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{vmatrix} \qquad M_{R'} = \begin{vmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{vmatrix}$$

$$M_{(T\cup R')} = \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{vmatrix}$$

$$M_{(S^{-1}\cap R)' \circ (T\cup R')} = \begin{vmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 \end{vmatrix} \odot \begin{vmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{vmatrix} = \begin{vmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{vmatrix}$$

Hay que observar que los resultados obtenidos son los mismos, tanto mediante conjuntos como por matrices.

6.6 Propiedades de las relaciones

Sean los conjuntos A, B, C y D, y las relaciones R: $A \to B$, S: $B \to C$, T: $C \to D$. Entonces se pueden establecer las siguientes propiedades:

- Si $R \subseteq S$, entonces $R^{-1} \subseteq S^{-1}$.
- Si $R \subseteq S$, entonces $R' \subseteq S'$.
- $(R \cap S)^{-1} = R^{-1} \cap S^{-1}$.
- $(R \cup S)^{-1} = R^{-1} \cup S^{-1}$.
- Si R es reflexiva, también lo es R^{-1}.
- R es reflexiva si y sólo si R′ es irreflexiva.
- Si R y S son reflexivas, entonces también lo son $R \cap S$ y $R \cup S$.
- R es simétrica si y sólo si $R = R^{-1}$.

- R es simétrica si y sólo si $R \cap R^{-1} = \varnothing$ cuando $R \neq A \times B$.
- Si R es simétrica, también lo son R′ y R^{-1}.
- Si R y S son simétricas, también lo son $R \cap S$ y $R \cup S$.
- R es antisimétrica si y sólo si $(R \cap R^{-1}) \subseteq I$.
- $(R \circ S) \circ T = R \circ (S \circ T)$.
- $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$.
- $(S \cup T) \circ R = (S \circ R) \cup (T \circ R)$.

Aunque se puede verificar que cada una de estas propiedades es válida para relaciones particulares, es recomendable plantear una demostración formal para establecer la validez general de éstas.

Ejemplo 6.11. Sean los conjuntos A = B = C = D y las relaciones R: A → B, S: B → C, T: C → D.

a) Demostrar que $(R \cap S)^{-1} = R^{-1} \cap S^{-1}$.
b) Demostrar que $(S \cup T) \circ R = (S \circ R) \cup (T \circ R)$.

Solución a

Si $(a, b) \in R$ y $(a, b) \in S$ entonces $(a, b) \in (R \cap S)$ y por tanto $(b, a) \in (R \cap S)^{-1}$.

Por otro lado, como $(a, b) \in R$ entonces $(b, a) \in R^{-1}$. Como $(a, b) \in S$, entonces $(b, a) \in S^{-1}$.

Si $(b, a) \in R^{-1}$ y $(b, a) \in S^{-1}$, entonces $(b, a) \in (R^{-1} \cap S^{-1})$.

Se concluye que $(R \cap S)^{-1} = R^{-1} \cap S^{-1}$.

Solución b

Si $(a, b) \in S$ o $(a, b) \in T$, entonces $(a, b) \in (S \cup T)$. Si además $(b, c) \in R$, entonces $(a, c) \in ((S \cup T) \circ R)$.

Por otro lado, como $(a, b) \in S$ y $(b, c) \in R$ entonces $(a, c) \in (S \circ R)$.

Como $(a, b) \in T$ y $(b, c) \in R$, entonces $(a, c) \in (T \circ R)$ y por lo tanto $(a, c) \in (S \circ R) \cup (T \circ R)$.

Se concluye que $(S \cup T) \circ R = (S \circ R) \cup (T \circ R)$.

6.7 Aplicaciones de las relaciones

Como ya se mencionó, las relaciones tienen muchas aplicaciones y en particular las que se presentan a continuación en el área de la computación.

6.7.1 Una lista enlazada es una relación

Sea A un vector de dimensión N que contiene nombres de personas, los cuales fueron colocados de acuerdo con el orden en que llegan, y sea P otro vector de las mismas dimensiones para guardar la dirección del siguiente nombre. Además se considera una variable X que guarda la posición en donde inicia la tabla de nombres.

a) Si el orden en que llegan los nombres es "María", "Juan", "Ana", "Pedro", "Jaime", ¿cuál es el valor de la variable X y cómo quedarían los vectores A y P?

b) ¿Cuál es el grafo dirigido de la relación formada por los arreglos A, P y la variable X?

c) Supóngase que se dan de alta los nombres "Benito" y "Luis" y se da de baja a "Juan". ¿Cómo quedaría la información en los arreglos y cuál es el grafo dirigido?

Solución de (a)

Considérese que la variable que indica el inicio de la lista es X = * y que los arreglos A y P están vacíos, como se muestra en la siguiente tabla:

X = * (* significa fin de lista)

	A		P
1		1	
2		2	
3		3	
4		4	
5		5	
.		.	
.		.	
N		N	

Llegar el primer nombre los arreglos quedan de la siguiente manera:

X = 1

	A	P
1	María	*
2		
3		
4		
5		
.		
.		
N		

La variable X = 1 indica que el primer nombre de la lista está en la posición 1 del arreglo A. El * en P indica que ya no hay más nombres y ahí termina la lista.

Cuando llega el segundo nombre los arreglos tienen la siguiente información:

X = 2

	A	P
1	María	*
2	Juan	1
3		
4		
5		
.		
.		
N		

Como el nombre de Juan se coloca alfabéticamente antes que María, ahora la variable que indica el inicio de la lista apunta a la posición de ese nombre X = 2. En esa misma posición pero para el arreglo P, se coloca el número 1 que indica que la posición del siguiente nombre a recorrer está en la posición número 1 del arreglo A y el * en P significa que ahí termina la lista.

Al agregar los nombres de Ana, Pedro y Jaime, los arreglos quedan como se muestra a continuación:

X = 3

	A
1	María
2	Juan
3	Ana
4	Pedro
5	Jaime
.	
.	
N	

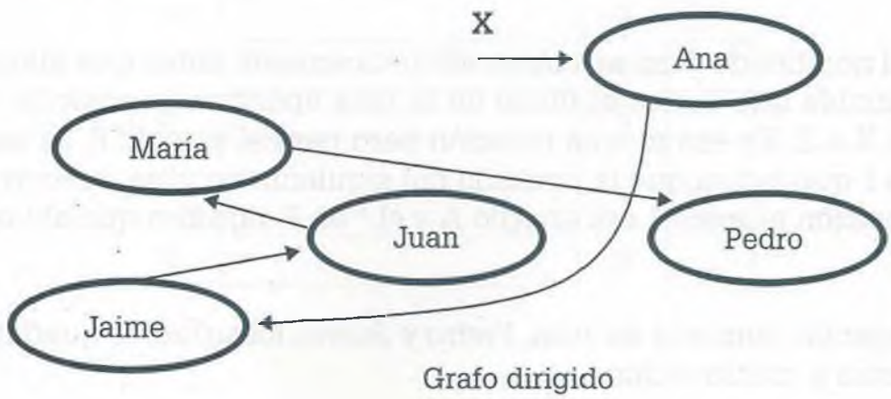
	P
1	4
2	1
3	5
4	*
5	2
.	
.	
N	

Hay que observar que al colocar la información de esta forma es posible recorrerla en orden alfabético, permitiendo con ello un acceso más rápido. El primer nombre está en la posición 3, como lo indica la variable X, el que sigue está en la posición 5, como lo indica el arreglo P, el que le sigue en la posición 2, y así sucesivamente hasta llegar al fin de lista indicado por *.

En el contexto de "Estructura de datos" esta forma de arreglar la información se conoce como *lista enlazada* y es utilizada en el ámbito de la computación con la finalidad de acomodar la información de forma que cuando se busque algo se encuentre de manera eficiente. Por ejemplo, si se busca el nombre Jaime se encontraría en el segundo paso ya que está inmediatamente después de Ana y no es necesario recorrer toda la lista. Si se buscara un nombre que no existe en el arreglo, como por ejemplo Carlos, y si se hiciera un programa que además de recorrer la información comparara alfabéticamente los nombres, en el segundo paso mandaría el mensaje de que "Carlos no está en la lista".

Esta lista enlazada también es una relación en la que los nodos son los nombres de las personas, y la flecha que relaciona los nodos es la información del arreglo P. Además de que se indica en dónde comienza el recorrido de la información.

Solución de (b)



Grafo dirigido

Solución de (c)

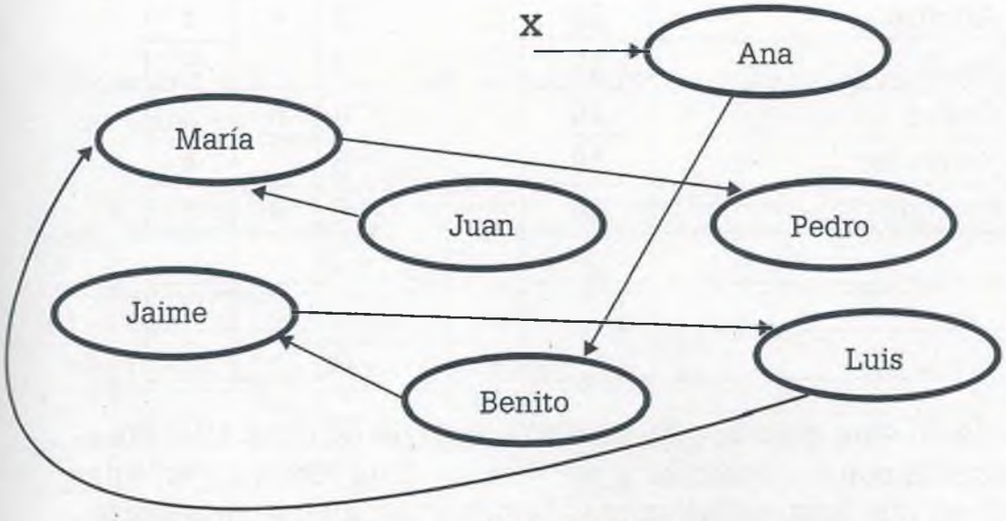
Después de dar de alta los nombres Benito y Luis, y de baja el de Juan, los arreglos quedan de la siguiente manera:

X = 3

	A		P
1	María	1	4
2	Juan	2	1
3	Ana	3	6
4	Pedro	4	*
5	Jaime	5	7
6	Benito	6	5
7	Luis	7	1
.		.	
.		.	
N		N	

En la tabla anterior se observa que cada vez que se da de alta un nuevo nombre, se llevan a cabo los ajustes en los apuntadores, y cuando se da de baja alguna persona simplemente se realiza la desconexión como ocurre con el caso del nombre Juan.

Por otro lado, el grafo queda de la siguiente manera:



Grafo dirigido

En este grafo dirigido se puede observar que el nodo Juan, al desconectarlo, ya no tiene ninguna flecha que apunte hacia él, y aunque él tiene una flecha que apunta a María ésta ya no importa ya que con ella no surte ningún efecto. En computación, para no desperdiciar espacio en memoria se puede guardar una lista de espacios disponibles y se pueden volver a ocupar cuando se necesiten, de forma que al lugar donde está Juan se le pondría una marca y posteriormente se usaría ese lugar para guardar otro nombre que se quisiera dar de alta.

Lo importante en este caso es observar que una lista enlazada es una relación. Pero aunque en los nodos solamente se representa un dato, realmente puede ser el registro de un archivo o bien una fila de una tabla en donde no solamente se tenga el nombre de la persona, sino otros datos como edad, salario, antigüedad y puesto, entre otros. De esta forma, al llegar al nombre de una persona, se puede acceder también a la información restante.

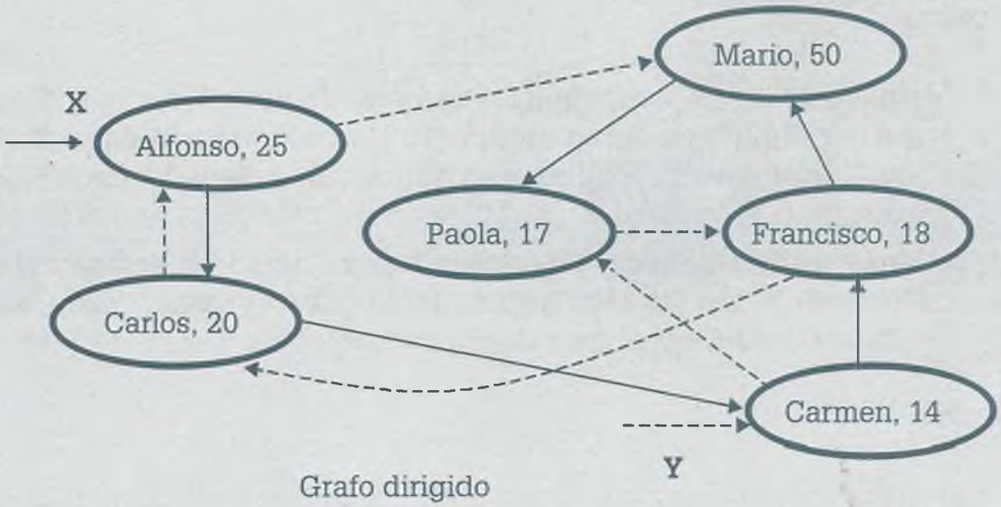
Ejemplo 6.12. Colocar en un arreglo A los nombres de las personas con sus respectivas edades, y en otro arreglo P los apuntadores correspondientes para recorrer dicha información alfabéticamente o por edades (en orden ascendente) si el orden en que llegan es el siguiente: Mario, 50; Alfonso, 25; Paola, 17; Carlos, 20; Francisco, 18; Carmen, 14. Usar como variables para inicio del recorrido a X para nombres, y a Y para edades.

X = 2 Y = 6

	A	
1	Mario	50
2	Alfonso	25
3	Paola	17
4	Carlos	20
5	Francisco	18
6	Carmen	14
.		
.		
N		

	P	
1	3	*
2	4	1
3	*	5
4	6	2
5	1	4
6	5	3
.		
.		
N		

Observar cómo en este caso la información se puede recorrer alfabéticamente, de acuerdo con los nombres y por edades. Esta forma de arreglar la información es una lista doblemente enlazada y también es una estructura de datos, así como también lo son las pilas y colas, mismas que se estudian en "Estructura de datos".



Grafo dirigido

En la figura, las flechas con líneas punteadas muestran el recorrido de la información para la edad y las flechas con líneas continuas para el nombre.

6.7.2 Las relaciones en las bases de datos

Un archivo en una base de datos también es una relación, y es posible llevar a cabo operaciones entre archivos aplicando las operaciones de relaciones. De hecho la aplicación de las relaciones en las bases de datos es muy común y así lo muestran los diferentes manejadores de bases de datos que existen en el mercado, los cuales tratan la información usando el álgebra relacional, esto es, las operaciones entre relaciones: unión, intersección, complementación, multiplicación booleana e inversa.

Ejemplo 6.13. Sean los archivos (relaciones) A y B que se indican a continuación:

Archivo A

Reg.	Nombre	Puesto	Salario	Antigüedad
1	Juan	Supervisor	4 000	5
2	Lorena	Secretaria	3 000	2
3	Jaime	Obrero	1 800	7
4	Alicia	Gerente	8 000	3
5	Alfredo	Obrero	2 100	9
6	Carlos	Supervisor	4 800	6
7	Alberto	Supervisor	2 400	2

Archivo B

Reg.	Nombre	Puesto	H. Semanales	H. Extras
1	Juan	Supervisor	40	0
2	Lorena	Secretaria	40	0
3	Jaime	Obrero	40	12
4	Alicia	Gerente	40	0
5	Alfredo	Obrero	40	8
6	Carlos	Supervisor	40	6
7	Alberto	Supervisor	40	4

Determinar

a) Una relación que contenga los campos Nombre, Puesto, H. Extras y Antigüedad, en este orden, para los trabajadores cuyo puesto sea "Supervisor" y que tengan una "antigüedad" mayor de 5 años.

b) Una relación que contenga los campos Puesto, H. Extras y Nombre, solamente para los trabajadores que hayan trabajado horas extras o bien su puesto sea "Gerente".

Solución

a) Supóngase que se coloca un 1 en las celdas que cumplan con la condición Puesto = "Supervisor" y "Antigüedad" ≥ 5, y un 0 en las demás celdas.

Archivo A

Reg.	Nombre	Puesto	Salario	Antigüedad
1	0	1	0	1
2	0	0	0	0
3	0	0	0	1
4	0	0	0	0
5	0	0	0	1
6	0	1	0	1
7	0	1	0	0

Archivo B

Reg.	Nombre	Puesto	H. Semanales	H. Extras
1	0	1	0	0
2	0	0	0	0
3	0	0	0	0
4	0	0	0	0
5	0	0	0	0
6	0	1	0	0
7	0	1	0	0

Posteriormente se realiza una unión entre estas dos relaciones.

A ∪ B

Reg.	Nombre	Puesto	Salario	Antigüedad	H. Semanales	H. Extras
1	0	1	0	1	0	0
2	0	0	0	0	0	0
3	0	0	0	1	0	0
4	0	0	0	0	0	0
5	0	0	0	1	0	0
6	0	1	0	1	0	0
7	0	1	0	0	0	0

Al llevar a cabo la intersección (Puesto = "Supervisor") ∩ (Antigüedad ≥ 5) se obtiene la siguiente relación con los campos que se indican y en el orden en que se solicitan:

Archivo C

Reg.	Nombre	Puesto	H. Extras	Antigüedad
1	0	1	0	1
2	0	1	0	1

Obsérvese cómo el número de registro no se conserva, ya que se trata de una relación nueva. Finalmente se sustituyen los ceros y unos por la información que corresponde para obtener la siguiente relación:

Archivo C

Reg.	Nombre	Puesto	H. Extras	Antigüedad
1	Juan	Supervisor	0	5
2	Carlos	Supervisor	6	6

Nótese cómo las operaciones se llevan a cabo en álgebra booleana y se pueden obtener relaciones de diferentes dimensiones con los campos en el orden deseado.

b) (H. Extras > 0) $\cup$ (Puesto = "Gerente"). Colocando un 1 en las celdas en donde se cumpla la condición y un 0 en donde no sea verdadera se tienen las siguientes relaciones:

Archivo A

Reg.	Nombre	Puesto	Salario	Antigüedad
1	0	0	0	0
2	0	0	0	0
3	0	0	0	0
4	0	1	0	0
5	0	0	0	0
6	0	0	0	0
7	0	0	0	0

Archivo B

Reg.	Nombre	Puesto	H. Semanales	H. Extras
1	0	0	0	0
2	0	0	0	0
3	0	0	0	1
4	0	1	0	0
5	0	0	0	1
6	0	0	0	1
7	0	0	0	1

Realizando la unión entre las dos relaciones se obtiene:

$A \cup B$

Reg.	Nombre	Puesto	Salario	Antigüedad	H. Semanales	H. Extras
1	0	0	0	0	0	0
2	0	0	0	0	0	0
3	0	0	0	0	0	1
4	0	1	0	0	0	0
5	0	0	0	0	0	1
6	0	0	0	0	0	1
7	0	0	0	0	0	1

Tomando solamente los campos solicitados (Puesto, H. Extras, Nombre) en el orden indicado, se tiene la siguiente relación:

Reg.	Puesto	H. Extras	Nombre
1	0	1	0
2	1	0	0
3	0	1	0
4	0	1	0
5	0	1	0

Finalmente se coloca la información correspondiente en cada una de las celdas para obtener:

Reg.	Puesto	H. Extras	Nombre
1	Obrero	12	Jaime
2	Gerente	0	Alicia
3	Obrero	8	Alfredo
4	Supervisor	6	Carlos
5	Supervisor	4	Alberto

Es importante mencionar que la colocación de los campos, así como las dimensiones del archivo resultante se obtienen con rutinas propias de los manejadores de bases de datos, pero lo que aquí interesa mostrar es que las operaciones entre relaciones son fundamentales en el campo de bases de datos.

6.8 Funciones

Los lenguajes de programación tradicionales se fundamentan en el concepto de función, y lo mismo ocurre en el caso de los visuales ya que por ejemplo para dibujar una ventana es necesario proporcionar los parámetros que no son otra cosa más que las propiedades del dibujo que al final se obtiene como resultado. Además de las funciones comunes (seno, coseno, raíz cuadrada, logaritmo natural, valor absoluto, etc.), los lenguajes de programación permiten diseñar funciones que pueden servir para realizar operaciones más complejas, y que se pueden usar en otros programas.

Definición. Una función f es una relación que asigna a cada elemento x de un conjunto A un único elemento b de un conjunto B. Sean A y B conjuntos no vacíos. Una función f de A en B se escribe como

$$f: A \rightarrow B$$

Se puede decir que todas las funciones son relaciones, pero no todas las relaciones son funciones. Para que una relación sea considerada como una función, deberá cumplir con las siguientes condiciones:

1) El Dom(f) = A. Esto es, el conjunto de los primeros elementos de todos los pares ordenados de la relación es el dominio de la función y también es igual al conjunto A.
2) Si hay dos pares ordenados (a, b) y (a, c) que pertenecen a f, entonces b = c. Esto significa que cada elemento del dominio deberá estar relacionado sólo con un elemento del codominio.

Ejemplo 6.14. Sean los conjuntos A = {1, 2, 3, 4} y B = {a, b, c}. Determinar si las siguientes relaciones son funciones:

a) R = {(1, b), (2, c), (3, a), (4, b)}
b) R = {(1, a), (2, c), (1, b), (3, a), (4, c)}
c) R = {(1, c), (2, c), (3, c), (4, c)}
d) R = {(1, b), (2, c), (4, a)}

Solución

a) Sí es una función, ya que cumple con las dos propiedades de la definición.

b) No es una función, ya que no satisface la segunda condición. Específicamente, existen al menos dos pares ordenados (1, a) y (1, b), cuyo primer elemento es 1 y $a \neq b$.

c) Sí es una función ya que satisface las condiciones de la definición planteada.

d) No es una función, ya que $Dom(f) \neq A$.

La propiedad 2 de la definición de función afirma que si $(a, b) \in f$, entonces b está determinada de manera única por a. Por lo tanto, se puede escribir:

$$b = f(a)$$

Esto significa que b está en función de a. También se dice que a es la variable independiente y b la variable dependiente. En computación al elemento a se le llama argumento de la función y a b se le llama valor de la función para el argumento a. A b también se le llama imagen de a bajo f, lo cual se ilustra en la siguiente figura:

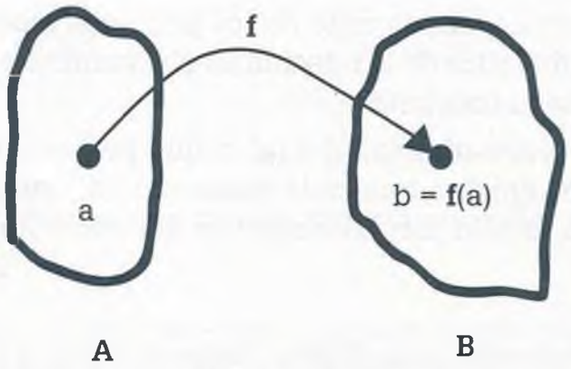


Imagen de una función

De esta forma, dados un valor del parámetro o variable independiente y la función, es posible determinar el valor de la variable dependiente. Por ejemplo, en la expresión $y = \cos(x)$ se tiene que la variable independiente o parámetro es x, la función es cos y la variable dependiente es y. En el caso de $y = x^2 + 3x - 1$ la variable independiente es x, la función es $x^2 + 3x - 1$ y la variable dependiente es y. En general los lenguajes de programación permiten crear esta función de la siguiente manera:

```
Función trinomio (x: real): real;
  inicio
    trinomio: = x * x + 3 * x – 1;
  fin.
```

En este caso el nombre de la función es "trinomio", el parámetro x que se encuentra dentro del paréntesis se define como del tipo real y el valor obtenido de la función también es del tipo real. De esta manera, cuando se llama a la función y = trinomio (0.8) esto significa que el programa va a calcular $y = (0.8)^2 + 3(0.8) - 1$.

Una función puede estar definida para dos o más parámetros o variables independientes, además de incluir la condición de que para éstos siempre se obtenga como resultado un solo valor. Un ejemplo de función de dos variables independientes a partir de las cuales se obtiene un solo valor es la función y = mod(a,b) cuyo valor es el resto que se obtiene al dividir a entre b, de forma que en particular para el caso y = mod(14, 3) se obtiene el valor 2, que es el resto de dividir 14 entre 3. Otro ejemplo es el de la función para obtener el mayor de dos números enteros:

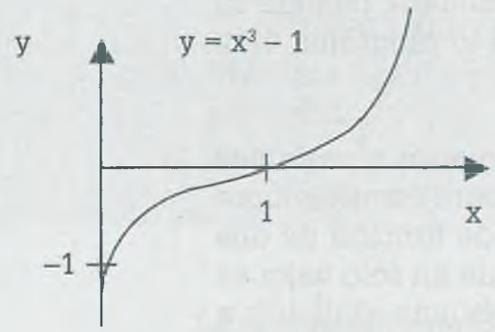
```
Función mayor (a, b: entero): entero;
inicio
  si a > b entonces
    mayor = a
  sino
    mayor = b;
fin.
```

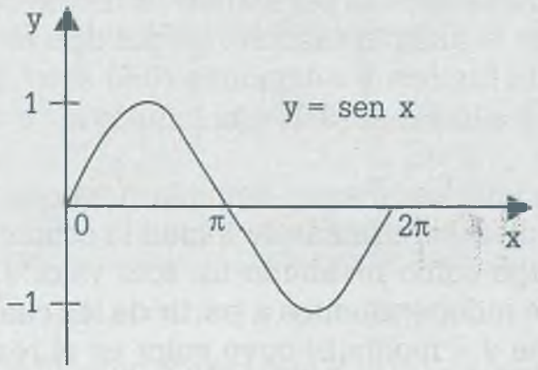
En este caso la función "mayor" recibe dos números enteros, a y b, y obtiene como resultado el mayor de ellos de forma que en el caso de y = mayor(5, 3) se obtiene y = 5.

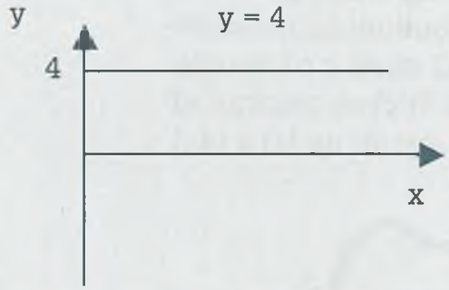
Dado que el objetivo es exponer las funciones desde un punto de vista general, éstas se denotarán como f, g, h, etcétera.

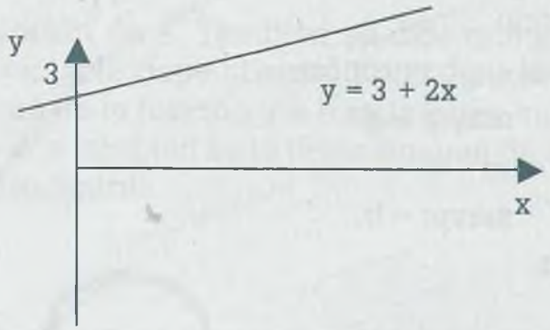
Como las funciones son relaciones, es posible representarlas de diferentes maneras: como conjuntos, matrices, ecuaciones, algoritmos y gráficas.

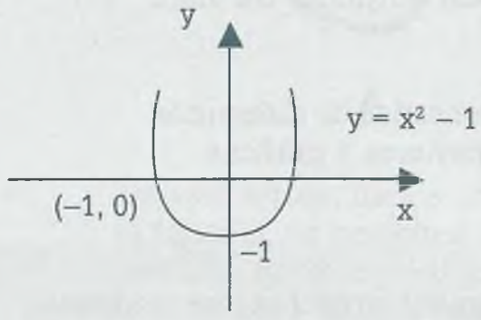
Ejemplo 6.15. Determinar cuáles de las siguientes gráficas corresponden a una función y cuáles a una relación. En todos los casos se tiene que A es el conjunto de los números reales.

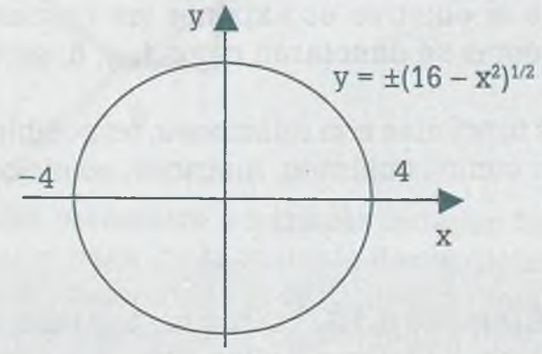






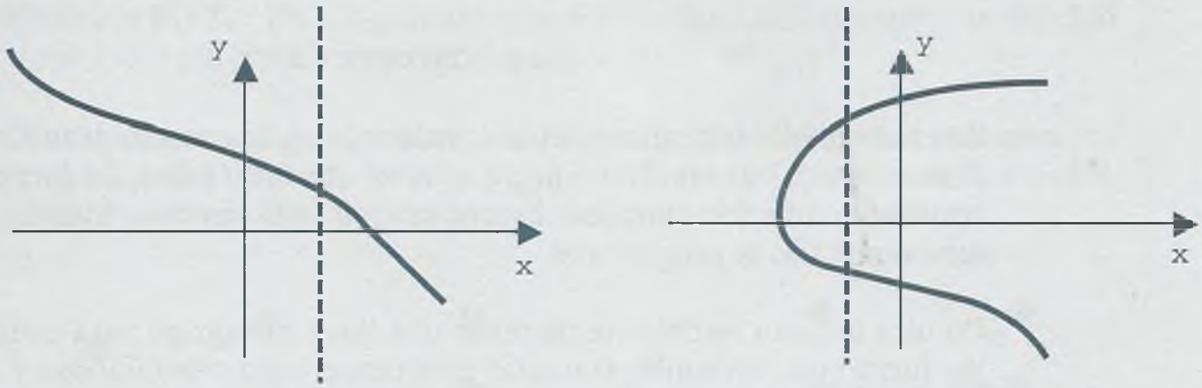






Las primeras cinco gráficas corresponden a una función ya que cumplen con las dos condiciones de la definición, sin embargo la sexta gráfica no es la de una función ya que de su expresión matemática se ve que a un mismo valor de x le corresponden dos de y, por ejemplo $f(4) = f(-4) = 0$, por lo que no cumple con la segunda condición de la definición de función.

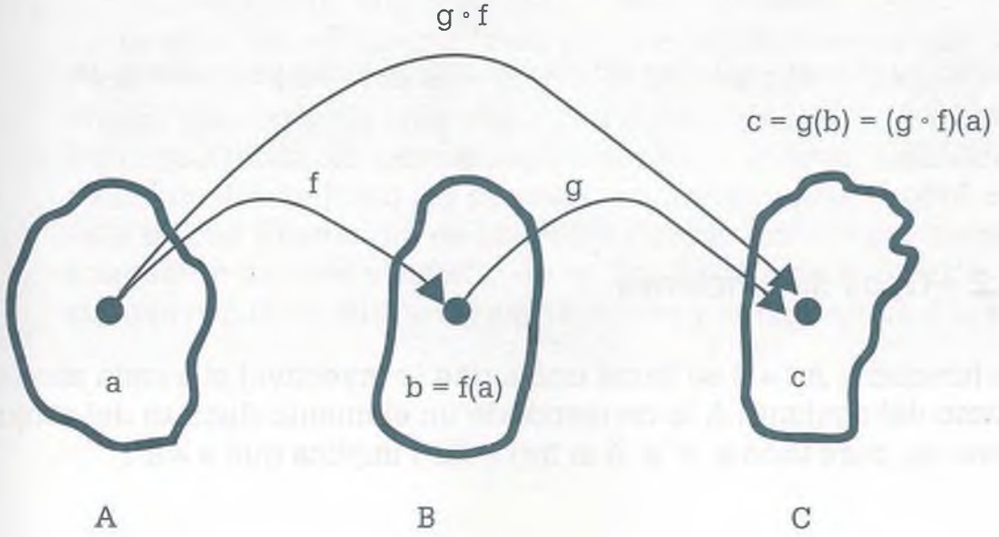
Un criterio que se utiliza para determinar si una gráfica es una función consiste en trazar una recta perpendicular al eje de las abscisas en cualquier parte de éste y ver cuántos puntos de la gráfica intersecta la recta; si sólo intersecta un punto entonces la gráfica corresponde a una función, y si en alguna parte intersecta más de un punto entonces la gráfica no corresponde a una función. En las siguientes dos gráficas se ilustra la aplicación de este criterio.



La primera gráfica corresponde a una función, mientras que la segunda no.

6.8.1 Composición de funciones

Si f: A → B y g: B → C son funciones, entonces la combinación g ∘ f llamada composición también es una función. En la siguiente figura se ilustra el concepto de composición de funciones.



Hay que observar que $(a, c) \in (g \circ f)$ si y sólo si $(a, b) \in f$ y $(b, c) \in g$ para alguna $b \in B$, de forma que

$$c = g(b) = g(f(a)) = (g \circ f)(a)$$

La composición de funciones es de gran utilidad en el campo de la computación, ya que permite la aplicación de varias funciones en una misma línea de código, dando por resultado programas más compactos. Como ejemplo de esto considérese la siguiente línea de código:

```
w: = abs (sqrt (cos (x – 3 * y)))
```

Se observa que esta instrucción contiene la aplicación de tres funciones. coseno (cos), raíz cuadrada (sqrt) y valor absoluto (abs), de forma que al resultado obtenido al aplicar la función cos se le aplica la función sqrt y a este resultado la función abs.

De otra manera se debería de tener una línea de código para cada una de las funciones, haciendo con esto programas más voluminosos y algunas veces más complejos. Todas las funciones tienen un nombre seguido de argumentos de la función encerrados entre paréntesis: en este caso (x – 3 * y) es el argumento de la función cos, cos(x – 3 * y) es el argumento de la función sqrt y sqrt(cos(x – 3 * y)) es el argumento de la función abs.

Ejemplo 6.16. Sean $A = B = C = R$ y las funciones $f: A \to B$, $g: B \to C$ definidas respectivamente por $f(a) = a^2 - 1$, $g(b) = b + 2$. Entonces:

a) $(g \circ f)(3) = g(f(3)) = g(3^2 - 1) = g(8) = 8 + 2 = 10$
b) $(f \circ g)(-1) = f(g(-1)) = f(-1 + 2) = f(1) = (1^2 - 1) = 0$
c) $(g \circ f)(x - 1) = g(f(x - 1)) = g(x^2 - 2x) = x^2 - 2x + 2$
d) $(g \circ g \circ f)(1) = g(g(f(1)) = g(g(1^2 - 1)) = g(g(0)) = g(2) = 4$

En este ejemplo se observa que las funciones compuestas se evalúan de dentro hacia fuera.

6.8.2 Tipos de funciones

Una función $f: A \to B$ se llama uno a uno (o inyectiva) si a cada elemento distinto del conjunto A le corresponde un elemento distinto del conjunto B, esto es, para todo $a, a' \in A$ si $f(a) = f(a')$ implica que $a = a'$.

Una función f: A → B se llama sobre (o suprayectiva) si el conjunto de los segundos elementos de los pares ordenados de la función es igual al conjunto B, dicho de otra forma, es sobre si Cod(f) = B.

Cuando una función es uno a uno y sobre (o biyectiva), se dice que f tiene una correspondencia uno a uno.

Ejemplo 6.17. En la siguiente figura se ilustran los conceptos de función inyectiva, suprayectiva y biyectiva.

Considérese que los puntos de la izquierda son los elementos del conjunto A, que los puntos de la derecha son los elementos del conjunto B, y que la relación entre ambos conjuntos es la indicada por las flechas.

Función inyectiva | Función suprayectiva | Función biyectiva

En esta figura hay que observar que todos los elementos del conjunto A están relacionados en todos los casos que se muestran, independientemente del tipo de función de que se trate; si no fuera de esta manera no se trataría de funciones ya que no se estaría cumpliendo la primera condición de la definición de funciones.

En relación con el tipo de funciones en esta figura hay que observar que en la función inyectiva cada elemento del conjunto A está relacionado sólo con un elemento del conjunto B; en el caso de la función suprayectiva se tiene que todos los elementos del conjunto B están relacionados con un elemento del conjunto A y por esta razón se considera sobre, sin embargo, esta misma función no es inyectiva porque uno de sus elementos no es imagen de un solo elemento de A. Por último, en la función biyectiva se cumple con la condición de ser inyectiva y suprayectiva a la vez.

Ejemplo 6.18. En cada inciso determinar si la función dada es inyectiva, suprayectiva o biyectiva:

a) $A = \{a, b, c, d\}$
$B = \{1, 2, 3\}$
$f = \{(a, 2), (b, 3), (c, 2), (d, 1)\}$

b) $A = \{1, 2, 3, 4, 5\} = B$
$f = \{(1, 3), (2, 5), (4, 3), (3, 5), (5, 1)\}$

c) $A = B = R$. Aquí R es el conjunto de los números reales.
$f(a) = |a|$

d) $A = B = \{1, 2, 3, 4, 5\}$ y sea $f = \{(1, 2), (2, 5), (3, 1), (4, 4), (5, 3)\}$.

Solución de (a)

- No es inyectiva ya que no se cumple que para $a \neq a'$, entonces $f(a) \neq f(a')$. Ejemplos de esto son los pares ordenados (a, 2) y (c, 2).
- Sí es suprayectiva ya que $Cod(f) = B = \{1, 2, 3\}$.
- No es biyectiva porque debería ser inyectiva y suprayectiva a la vez.

Solución de (b)

- No es inyectiva ya que no se cumple que para $a \neq a'$ debe ser $f(a) \neq f(a')$, como lo muestran los pares (1, 3) y (4, 3).
- No es suprayectiva ya que $Cod(f) \neq B$. Faltan los elementos 2 y 4.
- Obviamente tampoco es biyectiva.

Solución de (c)

- No es inyectiva ya que por ejemplo para $a = -1.7$ y $a' = 1.7$ se tiene que $f(a) = f(a')$ y por lo tanto no se cumple que para $a \neq a'$ debe ser $f(a) \neq f(a')$.
- No es suprayectiva ya que $cod(f) \neq B$. Faltan los negativos en el codominio.
- No es biyectiva.

Solución de (d)

- Es inyectiva ya que cada elemento del dominio tiene exclusivamente una imagen del codominio.
- Es suprayectiva, ya que $Cod(f) = B$.
- Como es inyectiva y suprayectiva a la vez, por lo tanto también es biyectiva.

6.8.3 Funciones invertibles

Una función f: A → B es invertible si su relación inversa f^{-1} es una función. Específicamente:

a) f^{-1} es una función si y sólo si f es inyectiva y suprayectiva, es decir, es una biyección.

b) Si el inciso (a) se cumple, entonces la función f^{-1} también es una función biyectiva y $(f^{-1})^{-1} = f$.

Ejemplo 6.19. Sea f: A → B con A = {1, 2, 3} y B = {rojo, verde, azul}. Entonces f = {(1, verde), (2, azul), (3, rojo)}. Obtener f^{-1}.

Solución

Como la función es inyectiva y suprayectiva, entonces es biyectiva y se puede obtener su inversa:

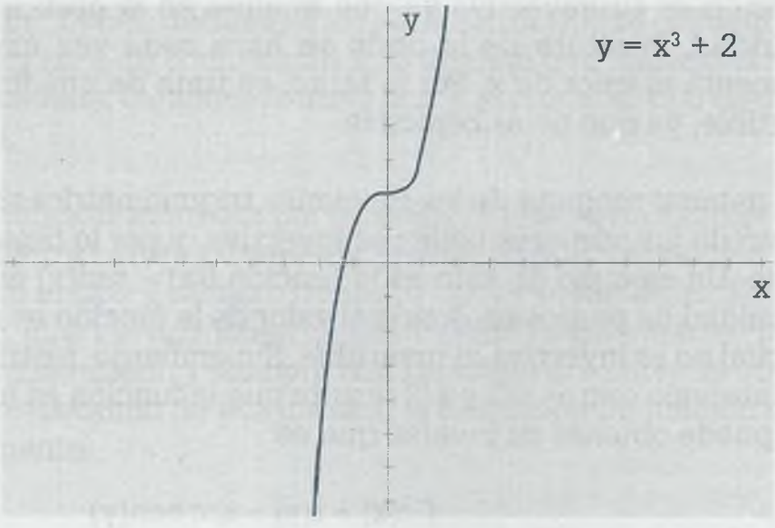
$$f^{-1} = \{(\text{verde}, 1), (\text{azul}, 2), (\text{rojo}, 3)\}$$

Como se ve, esta función f^{-1} también es biyectiva.

Ejemplo 6.20. Sean A = B = R y $f(a) = a^3 + 2$. Obtener la inversa de la función.

Solución

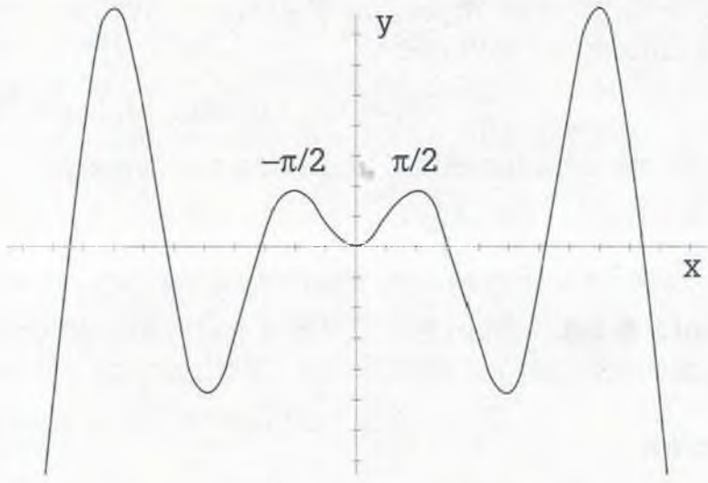
Lo primero que hay que hacer es determinar si la función dada es biyectiva y para esto se observa su gráfica.



A partir de la gráfica se ve que se trata de una función inyectiva, ya que a cada valor $x \in R$ le corresponde un único valor de $y \in R$. También se ve que es suprayectiva, ya que $Cod(f) = B = R$. Por lo tanto se trata de una función biyectiva e invertible, y su expresión es el resultado de despejar la variable independiente en la función inicial:

$$f(a) = a^3 + 2$$
$$b = a^3 + 2$$
$$a = (b - 2)^{1/3}$$
$$f^{-1}(a) = f(b) = (b - 2)^{1/3}$$

Sea $A = B = R$ y sea la función $y = x \operatorname{sen}(x)$ cuya gráfica es la siguiente:



A partir de la gráfica se puede observar que $y = x \operatorname{sen}(x)$ no es inyectiva, ya que para infinidad de valores $x \in R$ se tiene que $y = 0$. Por otro lado, la función es suprayectiva ya que aunque en la gráfica no se observa con claridad, la altura de la onda se hace cada vez mayor a medida que aumenta el valor de x. Por lo tanto, se trata de una función que no es invertible, ya que no es biyectiva.

En general ninguna de las funciones trigonométricas definidas en el conjunto de los números reales es biyectiva, y por lo tanto ninguna es invertible. Un ejemplo de esto es la función $f(x) = \operatorname{sen}(x)$ en la cual existe una infinidad de puntos en donde el valor de la función es el mismo, razón por la cual no es inyectiva ni invertible. Sin embargo, restringiendo el dominio al intervalo entre $-\pi/2$ y $\pi/2$ resulta que la función es inyectiva y por tanto se puede obtener su inversa que es:

$$f^{-1}(x) = f(y) = \text{arc sen}(y)$$

Por tanto, para determinar si una función es invertible se debe tener presente su dominio y codominio.

6.9 Aplicación de las funciones

Cada uno de los lenguajes conocidos como Java, C++, Pascal y Basic, tienen funciones integradas al lenguaje como sin(x), abs(x), sqrt(x), mod(x, y) y exp(x), entre otras. Además, en cada uno de los lenguajes es posible crear funciones con características especiales que las funciones estándar no tienen, pero que se consideran necesarias ya sea porque se usan con mucha frecuencia, por que permiten dar claridad al código o porque hacen más compactos los programas. Por ejemplo, si las funciones $y = x^3 - 1$, potencia = a^b se usan frecuentemente en un programa, se pueden crear de la siguiente manera:

```
Función y(x: real): real          Función potencia(a, b: entero): entero
Inicio                            Inicio
  y: = x^3 − 1                      potencia: = a^b
  retornar y                        retornar potencia
fin                               fin
```

Una vez creadas las funciones, simplemente se les pasa el valor o valores por medio de los parámetros que están entre paréntesis y de esa manera se obtiene el resultado. Por ejemplo:

```
Imprimir y(2)          El resultado que imprime es 7(2^3 − 1)
r: = potencia(2, 3)    El resultado obtenido es r = 8(2^3)
```

Algunos lenguajes usan la palabra "Función" y otros no, pero a todas las funciones se les da un nombre (en los casos anteriores es *y* y *potencia*), ya que el nombre de la función actúa como variable en la cual se almacena el resultado de la función. Deben declararse los parámetros que se encuentran dentro del paréntesis (x, a, b) como algún tipo de dato que maneje el lenguaje (entero, real, cadena, carácter, flotante, etc.), así como el tipo de dato de la propia función.

Cuando se llama la función se invoca por su nombre y se envía el valor del parámetro (o parámetros) entre paréntesis, cuidando que los valores enviados coincidan con el tipo y número de datos de los parámetros. Es importante mencionar que las funciones pueden recibir varios datos, pero solo proporcionan un resultado y siempre con los mismos valores se obtiene el mismo valor, respetando de esa manera la definición de función que se expuso anteriormente.

6.10 Resumen

- **Relación.** Una relación R se forma al unir elementos de diferentes conjuntos que cumplen con cierta propiedad o característica. Los elementos que se unen pueden ser de dos, tres o más conjuntos.

Como ejemplo supóngase que se tienen los conjuntos A, B y C, además de que elementos del conjunto A están relacionados con elementos del conjunto B y elementos del conjunto C porque cumplen con ciertas propiedades, de forma que se puede tener una relación R integrada por tripletas. Los conjuntos son

$A = \{1, 2, 3, 4, 5, 6, 7\}$

$B = \{a, e, m, p, u\}$

$C = \{\text{verde, azul, café, amarillo}\}$

Considérese que R está formada por tripletas que contienen un elemento de A que es divisible entre 3, uno de B que es una letra vocal y uno de C que es un color básico:

R = {(3,a,azul),(3,a,amarillo),(3,e,azul),(3,e,amarillo),(3,u,azul),(3,u,amarillo),
(6,a,azul),(6,a,amarillo),(6,e,azul),(6,e,amarillo),(6,u,azul),(6,u,amarillo)}

Esta relación es una relación trinaria porque está conformada por elementos de tres conjuntos distintos.

Las relaciones más comunes en ciencias de la computación son las relaciones binarias, que están integradas por pares de elementos de dos conjuntos.

Sean A y B dos conjuntos no vacíos. Una relación binaria R es un conjunto de pares ordenados, en donde el primer elemento a está relacionado con el segundo elemento b por medio de cierta propiedad o característica, la cual se indica como aRb mientras que para la relación se tiene que

$$\mathbf{R} = \{(a,b) \mid a \in A \text{ y } b \in B\}$$

- **Producto cartesiano ($A \times B$).** Es la combinación de todos los elementos del conjunto A con todos los elementos del conjunto B.

- **Dominio de R (Dom(R)).** Conjunto de todos los primeros elementos de los pares encontrados en una relación.

- **Codominio de R (Cod(R)).** Está conformado por los segundos elementos de los pares de la relación R.

- **Matriz de una relación (M_R).** Si A y B son dos conjuntos finitos y si R es una relación de A en B, es posible representar a R como una matriz M_R donde un elemento de la matriz es

$$1 \text{ si } (a,b) \in R$$
$$0 \text{ si } (a,b) \notin R$$

- **Grafo de una relación (G_R).** Se puede representar una relación por medio de una gráfica integrada por nodos y flechas, y a dicha gráfica se le conoce como "grafo dirigido" de R, en donde los elementos de los conjuntos A y B se representan como nodos y la relación que existe entre dichos elementos se indica por medio de una flecha que va del elemento del conjunto A al elemento del conjunto B con el que está relacionado.

- **Grafos dirigidos y no dirigidos.** En un grafo dirigido se representa la relación entre un elemento del conjunto A y un elemento del conjunto B por medio de una flecha que va de a hacia b. Sin embargo en un grafo no dirigido la relación es en ambos sentidos, esto significa que aRb y que bRa, por esa razón la relación se representa por una sola línea sin cabeza de flecha.

- **Propiedades de las relaciones.** En una relación es común que los elementos de A también sean elementos de B, es decir que A = B. Por ejemplo en una red de computadoras A y B tienen los mismos elementos porque relacionan computadoras, en una red carretera A y B tienen los mismos elementos porque relacionan ciudades, o en una red de agua potable A y B tienen los mismos elementos porque relacionan válvulas. Cuando ocurre que A = B, las relaciones pueden tener una, varias o ninguna de las siguientes propiedades:

Propiedad	Condición
Reflexiva	aRa; $\forall a \in A$. Esto es: todos los elementos de A están relacionados consigo mismos.
Irreflexiva	$(a,a) \notin R \ \forall a \in A$. Esto es: ningún elemento de A está relacionado con él mismo.
Simétrica	Cuando $(a,b) \in R$ entonces $(b,a) \in R$, o bien cuando $(a,b) \notin R$ entonces $(b,a) \notin R$. Esto es: los elementos simétricos de la relación son iguales.
Asimétrica	Cuando $(a,b) \in R$ entonces $(b,a) \notin R$, además si $a = b$ entonces $(a,a) \notin R$. Esto es: en ningún caso los dos pares simétricos están en la relación.
Antisimétrica	$(a,b) \notin R$ o bien $(b,a) \notin R$. Esto es: cuando $a \neq b$ en ningún caso los dos pares simétricos están en la relación.
Transitiva	*Si* $(a,b) \in R$ y $(b,c) \in R$, *entonces* $(a,c) \in R$. *Esto es:* cuando aRb y bRc entonces aRc.

- **Relación de equivalencia.** Es aquella que es reflexiva, simétrica y transitiva. Si la relación es la comunicación en una red de computadoras, dicha red debe ser una relación de equivalencia, porque toda computadora se puede comunicar con ella misma (reflexiva), si la computadora X se puede comunicar con la computadora W entonces la computadora W se puede comunicar con la X (simétrica). Si la computadora X se puede comunicar con la W y la computadora W se puede comunicar con la Z, entonces la computadora X se puede comunicar con la computadora Z (transitiva).

- **Clases de equivalencia [a].** Son conjuntos que contienen a todos los elementos $b \in B$ que están relacionados con $a \in A$.

$$[a] = \{b \mid b \in B, aRb\}$$

- **Partición (λ).** Es un conjunto de clases de equivalencia con las siguientes propiedades:

  a) Deberán estar contenidos todos los elementos del conjunto A.

  b) La intersección entre las clases de equivalencia debe ser vacía.

  $\lambda = \{[a] \mid a \in A$; la intersección entre clases de equivalencia es vacía$\}$

- **Cerraduras.** No todas las relaciones son de equivalencia, pero es posible hacer que tengan esta propiedad agregando los pares ordenados necesarios mínimos usando para ello las cerraduras: reflexiva ($R \cup I$), simétrica ($R \cup R^{-1}$) y transitiva ($R \cup R^2$).

- **Operación entre relaciones.** De la misma manera como se realizan operaciones entre conjuntos, también se pueden llevar a cabo las siguientes operaciones entre relaciones:

  a) Complemento de R (R′). Son a todos los pares ordenados que están en el producto cartesiano $A \times B$ pero que no están en R.

  b) Intersección de R y S ($R \cap S$). Sean R y S relaciones de un conjunto A en B, entonces se puede formar $R \cap S$. En términos de relación se puede ver que si $a(R \cap S)b$, entonces aRb y aSb. Por medio de matrices $M_{R \cap S}$ es el resultado de multiplicar elemento por elemento las matrices booleanas de R y S.

  c) Unión de R y S ($R \cup S$). La unión de dos relaciones ($R \cup S$) significa que aRb o bien aSb. Por medio de matrices se lleva a cabo una suma de matrices booleanas entre M_R y M_S para obtener $M_{R \cup S}$.

  d) Inversa de R (R^{-1}). La inversa de R se encuentra intercambiando la posición de a y b, esto implica que si $(a,b) \in R$ entonces $(b,a) \in R^{-1}$. En el caso de matrices la inversa se encuentra intercambiando filas por columnas (M_R^{-1}).

e) **Composición de R y S (R ∘ S).** La composición de relaciones R y S equivale a la propiedad transitiva, esto significa que si (a,b)∈ R y (b,c)∈ S, entonces (a,c)∈ (R ∘ S). Es posible también encontrar la composición de dos relaciones por medio de una multiplicación booleana de las matrices de las relaciones

$$(R \circ S = M_{R \circ S} = M_R \cdot M_S).$$

- **Una función f.** Asigna a cada elemento x de un conjunto A un único elemento b de un conjunto B. Sean A y B conjuntos no vacíos. Una función f de A en B se escribe f: A → B.

Se puede decir que todas las funciones son relaciones, pero no todas las relaciones son funciones. Para que una relación sea considerada como una función, deberá cumplir con las siguientes condiciones:

a) Dom(f) = A, esto es, el conjunto de los primeros elementos de todos los pares ordenados de la relación es el dominio de la función y también es igual al conjunto A.

b) Los elementos del dominio solamente deberán estar relacionados con un elemento del codominio.

- **Función inyectiva (o uno a uno).** Una función f: A → B se llama inyectiva, si a cada elemento distinto del conjunto A corresponde un elemento distinto del conjunto B.

- **Función suprayectiva (o sobre).** Una función f: A → B se llama suprayectiva, si el conjunto de los segundos elementos de los pares ordenados de la función es igual al conjunto B.

- **Función biyectiva (o correspondencia uno a uno).** Cuando una función f es inyectiva y suprayectiva a la vez, se dice que es biyectiva.

- **Función invertible.** Una función f: A → B es invertible si su relación inversa f^{-1} es también una función. Por otro lado, una relación es invertible si se trata de una función biyectiva.

- **Aplicación de las relaciones.** Las relaciones se pueden aplicar en bases de datos si un archivo se considera como una relación (o en otro contexto, una base de datos). También se aplican en estructuras de datos ya que una relación es una lista enlazada, una pila y también un árbol. Otra aplicación es en teoría de grafos partiendo de que una relación también es un grafo. En programación también se aplican ya que una función es una relación con ciertas características.

6.11 Problemas

6.1 Sean $A = B = C = \{1, 2, 3, 4, 5\}$, $R: A \to B$ tal que

$$R = \{(1, 1), (1, 3), (1, 5), (2, 2), (2, 4), (3, 1), (3, 3), (3, 5), (4, 2), (4, 4), (5, 1), (5, 3), (5, 5)\}$$

y $S: B \to C$ tal que bSc si y sólo si $a \geq b$ y b es impar.

a) Obtener los pares ordenados de la relación S.

b) Determinar M_R y M_S.

c) ¿Cuál es el grafo dirigido de R y cuál el de S?

d) Explicar si las relaciones R y S tienen algunas de las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica y transitiva.

e) Las relaciones R y S ¿son de equivalencia? Si no lo son, hacer que lo sean aplicando las cerraduras correspondientes. Encontrar después las clases de equivalencia y la partición.

f) Determinar $M_{(R' \cap S)}{}^{-1}$.

g) La relación obtenida en el inciso (f) ¿es de equivalencia?

6.2 Sean $A = B = C = \{1, 2, 3, 4\}$, $R: A \to B$ tal que aRb si y sólo si $a = b$ y $S: B \to C$ tal que bSc si y sólo si b es par y c es múltiplo de 4.

a) Determinar los pares ordenados de R y S.

b) Determinar el producto cartesiano $A \times B$.

c) Obtener el dominio y codominio de R y S.

d) ¿Cuáles son los grafos dirigidos de R y S?

e) ¿Cuáles son las matrices de R y S?

f) Obtener $(R' \cap S) \circ S^{-1}$.

g) La relación obtenida en el inciso (f) ¿es una relación de equivalencia? En caso de no ser así, hacer que lo sea aplicando las cerraduras: reflexiva, simétrica y transitiva.

h) Obtener las clases de equivalencia y partición de la relación.

i) ¿Cuál es el grafo dirigido y qué diferencia existe con el grafo no dirigido?

6.3 Sean A = B = C = {1, 2, 3, 4, 5}, R: A → B y S: B → C, donde:

R = {(1, 1), (1, 4), (1, 5), (2, 1), (2, 2), (2, 5), (3, 2), (3, 4), (4, 1), (4, 3), (4, 4), (4, 5), (5, 1), (5, 4), (5, 5)}

S = {(2, 1), (2, 3), (2, 4), (3, 1), (3, 3), (3, 4), (3, 5), (4, 3), (4, 4), (5, 1), (5, 2), (5, 3), (5, 5)}

a) Obtener $M_{(S^{-1} \cap R)^{\circ} S'}$.

b) Explicar si tiene o no las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica o transitiva.

c) ¿Cuál es su grafo dirigido?

6.4 Sean A = B = C = D = {1, 2, 3, 4, 5}, R: A → B en donde aRb si y sólo si a es par y b es primo, S: B → C en donde bSc si y sólo si c es divisible entre 3 y T: C → D donde T = {(1, 1), (3, 4)}.

a) Obtener los pares ordenados de las relaciones R y S.

b) Calcular los productos cartesianos A × B y C × D. ¿Qué diferencia existe entre estos dos productos cartesianos en este caso? Si los conjuntos A, B, C y D tuvieran diferentes elementos ¿ocurriría lo mismo?

c) Explicar si las relaciones R, S y T tienen una o varias de las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica, transitiva.

d) Alguna de las relaciones R, S o T es una relación de equivalencia (explique). En caso de ser así, obtener las clases de equivalencia y partición.

e) Representar las relaciones R, S y T como matriz y como grafo dirigido.

f) Establecer si las relaciones R, S y T cumplen con todo lo necesario para ser consideradas una función (justificar su respuesta).

g) Obtener $(R \cup T^{-1}) \cap (S^{-1} \cap R') \circ T)^{-1}$.

h) La relación obtenida en el inciso (g) ¿es una relación de equivalencia? En caso de no ser así, hacer que lo sea aplicando las cerraduras correspondientes.

i) Obtener las clases de equivalencia y partición de la relación obtenida en el inciso (h).

j) Obtener el grafo dirigido y no dirigido de la relación obtenida en el inciso (h).

6.5 Sean $A = B = Z^+$, donde aRb si y sólo si $|a - b| \leq 3$.

a) Explicar si R tiene o no alguna de las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica y transitiva.

b) ¿R es una relación de equivalencia? Si no lo es, ¿qué propiedad o propiedades le hacen falta para serlo?

6.6 Sean $A = B = C = D = \{1, 2, 3, 4, 5\}$, $R: A \to B$, $S: B \to C$ y $T: C \to D$, donde $R = \varnothing$, $S = B \times C$ y $T = \{(2, 3)\}$.

a) Explique si las relaciones R, S y T tienen las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica y transitiva.

b) ¿Alguna de las relaciones R, S y/o T son relaciones de equivalencia? Explique su respuesta.

6.7 Sean $A = B = C = D = \{1, 2, 3, 4\}$, $R: A \to B$, $S: B \to C$ y $T: C \to D$, donde:

$R = \{(1, 1), (1, 2), (1, 4), (2, 2), (2, 3), (2, 4), (3, 1), (3, 2), (3, 4), (4, 1), (4, 3), (4, 4)\}$

$S = \{(1, 2), (2, 2), (2, 3), (4, 3)\}$

$T = \{(3, 1), (3, 3), (3, 4), (4, 2)\}$

a) Obtener $M_{(T^{-1} \cup S) \cap (R \circ T)}$.

b) Si la relación resultante en el inciso (a) no es una relación de equivalencia, hacer que lo sea aplicando las cerraduras reflexiva, simétrica y transitiva.

c) Obtener las clases de equivalencia y partición.

d) ¿Cuál es el grafo dirigido y no dirigido de la relación que resulta en el inciso (b)?

6.8 Sean $A = B = C$, $R: A \to B$ y $S: B \to C$. Demostrar que:

a) R es asimétrica si y sólo si $R \cap R^{-1} = \varnothing$.

b) $(R \cap S)^2 \subseteq (R^2 \cap S^2)$.

6.9 Sean $A = B = C$, $R: A \to B$ y $S: B \to C$. Demostrar que:

a) Si $R \subseteq S$ entonces $R^{-1} \subseteq S^{-1}$.

b) R es antisimétrica si y sólo si $(R \cap R^{-1}) \subseteq I$.

6.10 Resolver cada uno de los siguientes incisos:

a) Sean $A = B = \{1, 2, 3, 4, 5\}$ y $R: A \to B$, donde

$R = \{(1, 2), (1, 4), (2, 2), (2, 5), (3, 1), (3, 5), (4, 3), (4, 4), (4, 5), (5, 1)\}$

- Explicar si tiene alguna o varias de las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica y/o transitiva.
- Si la relación anterior no es de equivalencia, hacer que lo sea aplicando las cerraduras correspondientes.
- Obtener las clases de equivalencia y la partición (si ésta existe).

b) Sean $A = B = C = \{a, b, c, d, e\}$, $R: A \to B$ y $S: A \to B$, donde:

$R = \{(a, c), (a, d), (a, e), (b, a), (b, b), (b, d), (c, a), (c, d), (d, c), (e, a), (e, b), (e, d), (e, e)\}$

$S = \{(a, a), (a, c), (a, e), (b, b), (b, c), (b, d), (c,b), (c, c), (c, d), (c, e), (d, a), (d, b), (d, e)\}$

- Obtener $((R \cap S^{-1}) \circ (S' \cup R^{-1}))'$. Hacerlo por conjuntos y ratificar dicho resultado por medio de matrices.
- Explicar si la relación obtenida tiene alguna o varias de las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica y/o transitiva.
- Cuál es el grafo dirigido de la relación resultante y establecer si se trata de una relación de equivalencia. Si no es así ¿cuál es el menor número de pares ordenados que le faltan para que lo sea?

c) Sean $A = B = C = \{1, 2, 3, 4, 5\}$, $R: A \to B$ y $S: A \to B$, donde

$R = \{(1, 3), (1, 5), (2, 1), (2, 2), (2, 4), (2, 5), (3, 2), (3, 3), (3, 5), (4, 1), (5, 3), (5, 5)\}$

$S = \{(b, c) \mid b \in B, c \in C, bRc$ si y sólo si b es primo y c es par$\}$

- Obtener $(R' \circ (S^{-1} \cup R^{-1}))$ por medio de matrices y de conjuntos.
- Explicar si la relación resultante tiene alguna o varias de las siguientes propiedades: reflexiva, irreflexiva, simétrica, asimétrica, antisimétrica y/o transitiva.
- Cuál es el grafo dirigido de la relación resultante y establecer si se trata de una relación de equivalencia.

d) Sean $A = B = C = \{1, 2, 3, 4, 5\}$, $R: A \to B$ y $S: A \to B$, donde

$R = \{(1, 1), (1, 4), (2, 3), (2, 4), (2, 5), (3, 2), (3, 3), (3, 5), (4, 5), (5, 1), (5, 2), (5, 5)\}$

$S = \{(1, 1), (1, 5), (2, 2), (2, 4), (2, 5), (5, 1), (5, 3), (5, 5)\}$.

Demostrar que:

- $(R \cap S)^{-1} = R^{-1} \cap S^{-1}$.
- $(R \cup S)^{-1} = R^{-1} \cup S^{-1}$.
- $(R \circ S) \circ T = R \circ (S \circ T)$.
- $(R \circ S)^{-1} = S^{-1} \circ R^{-1}$.
- $(S \cup T) \circ R = (S \circ R) \cup (T \circ R)$.

6.11 Guardar nombres de personas en un arreglo A, la posición del siguiente nombre en un arreglo P y el inicio del recorrido de la información en la variable X. El orden en que llegan los nombres es "Lorena", "Miriam", "Gustavo", "Alicia", "Fernando", "Juan".

a) ¿Cuál es el contenido de los arreglos A y P así como el de la variable X, si se desea recorrer los nombres alfabéticamente en forma ascendente?

b) Si se da de baja a "Fernando" y se dan de alta "José" y "Alejandra" ¿de qué manera quedarán los arreglos?

c) ¿Cuál es el grafo dirigido de los arreglos A y P, así como el de la variable X, en este momento?

d) ¿Cómo quedarían los arreglos si en lugar de acomodar la información para consultarla en forma ascendente, se desea hacer dicha consulta en forma descendente?

6.12 Se desea guardar los salarios en un arreglo A, la posición del siguiente salario en un arreglo P y el inicio del recorrido de la información en la variable X. El orden en que llegan los datos es 3 000, 4 000, 2 000, 5 000, 3 000, 3 500.

a) ¿Cuál es el contenido de los arreglos A y P, así como el de la variable X, si se desea recorrer la información de manera ascendente?

b) ¿Cuál es el grafo dirigido en este momento?

c) Si se da de baja el salario cuyo monto es 3 000 y se dan de alta 2 500 y 4 300, ¿cómo quedarían los arreglos?

d) ¿Cuál de los dos 3 000 fue dado de baja y por qué?

e) ¿Cuál es el grafo dirigido en este momento?

f) ¿Cómo quedaría la información si al arreglo P se le adiciona otra columna para recorrer la información en forma descendente, y se considera a la variable Y para indicar el inicio de esta lista?

6.13 Colocar en un arreglo A los nombres de personas con su respectiva edad, y en otro arreglo P los apuntadores correspondientes para recorrer dicha información alfabéticamente o por edades. El orden en que llega la información es "Francisco", 30; "Fausto", 18; "Arturo", 22; "Alberto", 32; "Lidia", 23; "Berta", 20. Usar X como variable para iniciar el recorrido de los nombres, y Y para indicar el inicio del recorrido de las edades.

a) ¿Cuál es el contenido de los arreglos A y P, así como el de las variables X, Y, si se desea recorrer los nombres alfabéticamente (ascendente) y las edades en forma descendente?

b) ¿Cuál es el grafo dirigido en este momento?

c) Si se da de alta a "Pedro", 13 y se da de baja a "Fausto", 18, ¿de qué forma quedarían los arreglos?

6.14 Colocar en un arreglo A el nombre de personas con su respectiva antigüedad, y en otro arreglo P los apuntadores correspondientes para recorrer dicha información alfabéticamente y por antigüedad. El orden en que llega la información es "Sandra", 3; "Carmen", 5; "Pablo", 1; "Alfonso", 2; "Santiago", 4; "Elena", 1. Usar X como variable para iniciar el recorrido de los nombres, y Y para inicio del recorrido de la antigüedad.

a) ¿Cuál es el contenido de los arreglos A y P, así como el de las variables X y Y, si se desea recorrer los nombres alfabéticamente (ascendente) y la antigüedad en forma ascendente?

b) ¿Cuál es el grafo dirigido en este momento?

c) Si se da de alta "Alfredo", 2, "Fermín", 4, y de baja "Elena", 1, ¿de qué forma quedarían los arreglos?

d) ¿Cuál es el grafo dirigido en este momento?

e) Si ahora se agrega una columna a P y se usa la variable Z para recorrer también los nombres en forma descendente, ¿cómo quedarían los arreglos?

6.15 Considérense los archivos A y B que se muestran a continuación:

Relación A

Reg.	Código	Nombre	Departamento
1	3 427	José	Mantenimiento
2	6 072	Pedro	Producción
3	8 611	Alicia	R. Humanos
4	7 512	Fernando	Producción
5	5 825	Carlos	Producción
6	7 020	Carmen	Contabilidad

Relación B

Reg.	Código	Puesto	Salario
1	3 427	Supervisor	4 300
2	6 072	Obrero	3 000
3	8 611	Secretaria	2 800
4	7 512	Obrero	3 200
5	5 825	Supervisor	5 000
6	7 020	Secretaria	3 000

Obtener:

a) Una relación con los campos Nombre, Puesto y Salario. Para las personas cuyo Departamento = "Producción" y Salario ≥ 3 100.

b) Una relación con los campos Código, Puesto y Departamento para las personas cuyo Puesto = "Obrero" o bien (Puesto = "Secretaria" y Departamento = "Contabilidad").

c) Una relación que contenga los campos Nombre, Departamento y Salario de todos los trabajadores que no pertenezcan al Departamento de producción.

6.16 Sean las relaciones A y B que se muestran a continuación:

Relación A

Reg.	Código	Producto	Precio
1	2 010	Galletas	3.00
2	3 040	Detergente	12.00
3	5 513	Mermelada	15.00
4	1 728	Aceite	16.00
5	6 724	Arroz	16.00
6	3 319	Frijol	18.00
7	6 502	Azúcar	20.00
8	3 045	Detergente	14.00
9	1 717	Aceite	18.00

Relación B

Reg.	Código	Presentación	Contenido
1	2 010	Paquete	250 gr.
2	3 040	Bolsa	600 gr.
3	5 513	Frasco	250 ml
4	1 728	Frasco	1 000 ml
5	6 724	Bolsa	1 000 gr.
6	3 319	Bolsa	1 000 gr.
7	6 502	Bolsa	2 000 gr.
8	3 045	Bolsa	600 gr.
9	1 717	Frasco	1 000 ml

Obtener:

a) Una relación con los campos Código, Producto, Presentación y Precio. Con Producto = "Detergente" o Presentación = "Bolsa".

b) Una relación con los campos Producto, Contenido y Precio. Para Precio ≥ 15.00 y (Contenido < 1 000 gr. o Contenido < 500 ml).

c) Una relación con los campos Código, Producto, Presentación, Contenido y Precio. Para Código = 5 513.

d) Relación con los campos Código, Producto y Presentación. Para Producto = "Detergente".

6.17 Sean los conjuntos A = {1, 2, 3, 4} y B = {a, b, c, d}. Para cada uno de los siguientes incisos:

- Indicar si la relación dada también es una función.
- Determinar Dom(R) y Cod(R).
- En caso de ser función, verificar si es inyectiva, suprayectiva y/o biyectiva.
- En caso de ser una función invertible, ¿cuál es f^{-1}?

a) R = {(1, a), (2, b), (3, d), (4, c)}

b) R = {(1, a), (2, a), (3, a), (4, a)}

c) R = {(1, b), (1, c), (2, a), (2, d)}

d) R = {(1, c), (2, c), (3, d), (4, d)}

6.18 Sean A = {1, 2, 3} y B = {a, b, c, d}. En relación con cada uno de los incisos:

- Indicar si la relación también es una función.
- Determinar Dom(R) y Cod(R).
- En caso de ser función, determinar si es inyectiva, suprayectiva y/o biyectiva.
- En caso de ser una función invertible, ¿cuál es f^{-1}?

a) R = {(1, a), (2, a), (3, a)}

b) R = {(1, a), (2, b), (2, c), (3, d)}

c) R = {(1, c), (2, b), (3, c), (3, a)}

d) R = {(2, a), (2, b), (2, c), (d, d)}

6.19 Sean A = B = R; $f(x) = -4x^3 - 2$; $g(y) = 3y^2 - 1$; $h(z) = 5z + 3$.

a) Demostrar que en cada caso realmente se trata de una función, y para esto utilizar la gráfica correspondiente.

b) Establecer si son funciones invertibles, y si es así obtener la inversa.

c) Determinar el valor de la composición $f \circ h \circ g \circ g(2)$ y $g \circ f \circ h \circ f(-x)$.

6.20 En cada uno de los siguientes incisos:

- Demostrar que se trata de una función utilizando un bosquejo de su gráfica.
- Establecer si se trata de una función invertible, y en caso de ser así obtener su inversa.

a) $f(x) = \dfrac{(x^2 - 8)^4}{\frac{5}{3}} + \dfrac{1}{7}$

para $A = \{x \mid x \in R;\ x > 2\}$; $B = \{x \mid x \in R;\ x > 1\}$.

b) $f(x) = \dfrac{\sqrt{x^3 - 1}}{7} + \dfrac{9}{5}$

para $A = \{x \mid x \in R;\ x > 1\}$; $B = \{x \mid x \in R;\ x > 2\}$.

c) $f(x) = 3\text{tg}(x)$ para $A = \{x \mid x \in R;\ -1 < x < 1\}$; $B = R$.

d) $f(x) = \text{sen}(x)$

para $A = \{x \mid x \in R;\ \pi < x < 3\pi\}$; $B = \{x \mid x \in R;\ -1 < x < 1\}$.

e) $f(x) = 3(-1)^{n+1}$ para $A = N$; $B = \{x \mid x \in Z;\ -3 \leq x \leq 3\}$

6.21 Sean $A = B = C = D = R$, $f: A \to B$, $g: B \to C$ y $h: C \to D$ definidas por

$f(a) = 3a + a^3$, $g(b) = b^5$, $h(c) = c + 1$. Obtener:

a) $g \circ f(2)$

b) $f \circ g(x - 1)$

c) $g \circ f \circ h\,(x)$

d) $f \circ g \circ h\,(-x)$

6.22 Sean $A = B = C = D = R$, $f: A \to B$, $g: B \to C$, y $h: C \to D$ definidas por

$f(a) = -a^3$, $g(b) = b^2 - 1$, $h(c) = 3c + 1$. Obtener:

a) $f \circ g \circ f(-1)$
b) $f \circ g \circ g(1 - x)$
c) $g \circ g \circ f \circ h(-x)$
d) $f \circ f \circ g \circ h(2x)$

6.23 Diseñar un algoritmo en donde al dar dos números enteros positivos, el algoritmo encuentre el mínimo común múltiplo de a y b usando para ello una función.

6.24 Diseñar un algoritmo que contenga una función para:

a) Obtener el mayor de 3 números enteros positivos mayor (x, y, z).
b) Obtener el máximo común divisor de dos números enteros positivos mcd(a, b).
c) Dado un número entero positivo, determinar si éste es primo o no.
d) En algunos lenguajes se utilizan algoritmos para elevar una cantidad a cualquier potencia. Si $x^n = e^{nLn(x)}$, donde n, $x \in R$, e = 2.7182, es la base de los logaritmos y Ln(x) es el logaritmo natural de x. Dados dos número reales x y n encontrar con la función anterior x^n.

6.25 Diseñar algoritmos para determinar si una relación es:

a) Reflexiva.
b) Irreflexiva.
c) Simétrica.
d) Asimétrica.
e) Antisimétrica.
f) Transitiva.

6.26 Diseñar algoritmos para llevar a cabo las siguientes operaciones entre relaciones:

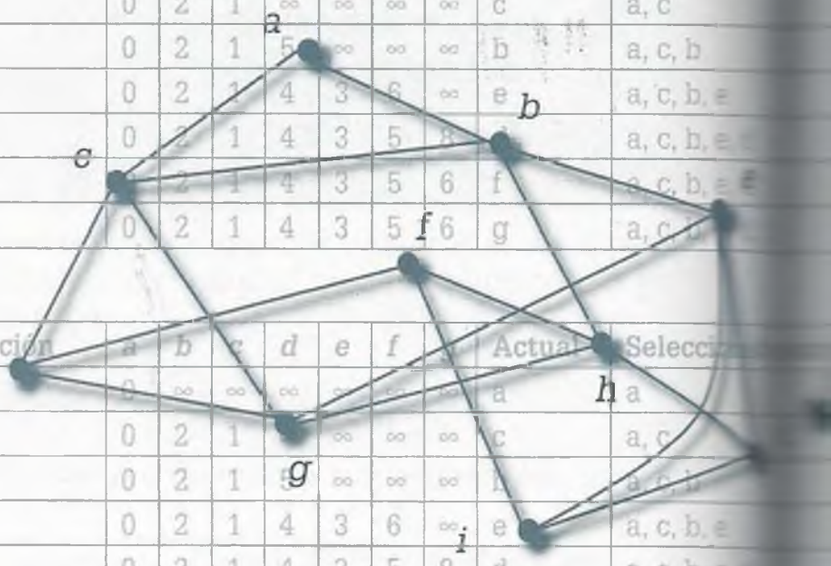
a) Unión.
b) Intersección.
c) Complementación.
d) Inversa.
e) Composición.

CAPÍTULO

VII

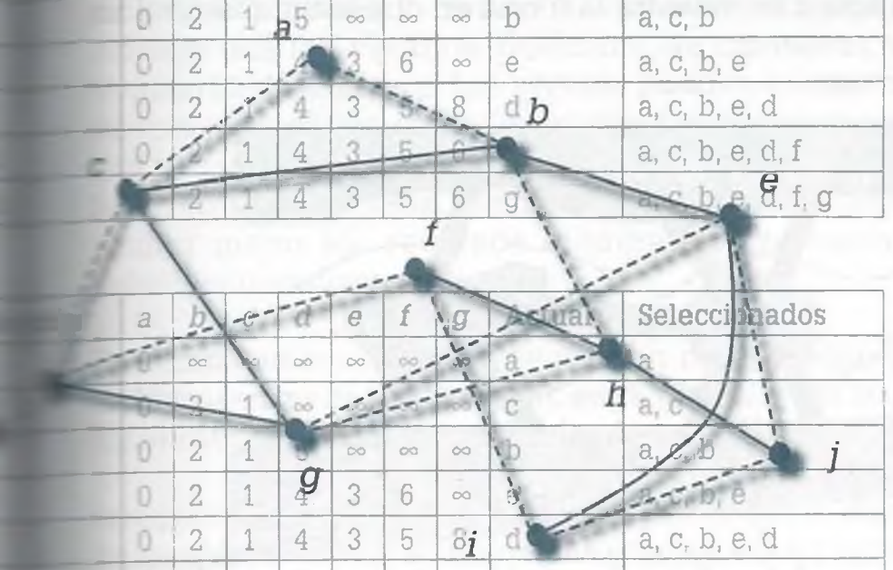
Grafos

Iteración	a	b	c	d	e	f	g	Actual	Selección
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞	c	a, c
2	0	2	1	5	∞	∞	∞	b	a, c, b
3	0	2	1	4	3	6	∞	e	a, c, b, e
4	0	2	1	4	3	5	8	d	a, c, b, e, d
5	0	2	1	4	3	5	6	f	a, c, b, e, d, f
6	0	2	1	4	3	5	6	g	a, c, b, e, d, f, g



Iteración	a	b	c	d	e	f	g	Actual	Selección
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞	c	a, c
2	0	2	1	5	∞	∞	∞	b	a, c, b
3	0	2	1	4	3	6	∞	e	a, c, b, e
4	0	2	1	4	3	5	8	d	a, c, b, e, d
5	0	2	1	4	3	5	6	f	a, c, b, e, d, f
6	0	2	1	4	3	5	6	g	a, c, b, e, d, f, g

	a	b	c	d	e	f	g	Actual	Seleccionados
	0	∞	∞	∞	∞	∞	∞	a	a
	0	2	1	∞	∞	∞	∞	c	a, c
	0	2	1	5	∞	∞	∞	b	a, c, b
	0	2	1	4	3	6	∞	e	a, c, b, e
	0	2	1	4	3	5	8	d	a, c, b, e, d
	0	2	1	4	3	5	6	f	a, c, b, e, d, f
	0	2	1	4	3	5	6	g	a, c, b, e, d, f, g



	a	b	c	d	e	f	g	Actual	Seleccionados
	0	∞	∞	∞	∞	∞	∞	a	a
	0	2	1	∞	∞	∞	∞	c	a, c
	0	2	1	5	∞	∞	∞	b	a, c, b
	0	2	1	4	3	6	∞	e	a, c, b, e
	0	2	1	4	3	5	8	d	a, c, b, e, d
	0	2	1	4	3	5	6	f	a, c, b, e, d, f
	0	2	1	4	3	5	6	g	a, c, b, e, d, f, g

Si simplemente hace girar la rueda, es álgebra; pero si contiene una idea, es topología.

Solomon Lefschetz

Objetivos

- Aprender la simbología y nomenclatura propia de la teoría de grafos y distinguir las características de éstos.
- Aplicar la teoría de grafos a la solución de problemas que se pueden resolver con esta metología.
- Saber qué condiciones debe cumplir un grafo que se considera tiene circuitos importantes como Euler y Hamilton.
- Usar la técnica de coloración de grafos para iluminar un mapa con la cantidad mínima de colores sin que se junten dos colores iguales.

7.1 Introducción

Leonhard Euler
(1707-1783)

Fue un matemático y físico suizo que es considerado uno de los más grandes matemáticos de la historia, ya que realizó importantes descubrimientos en áreas tan diversas como el cálculo, la teoría de grafos, la geometría, el álgebra, la teoría de números, el cálculo de variaciones, la mecánica, la hidrodinámica, la óptica y la astronomía, y también introdujo gran parte de la moderna terminología del análisis matemático.

Hacia 1738 Euler quedó prácticamente ciego y a pesar de este problema su productividad intelectual no disminuyó gracias a su gran capacidad de cálculo mental y a su memoria fotográfica. A partir de 1741 vivió en Berlín y aquí escribió más de 380 artículos y publicó dos de sus principales obras: la *Introductio in analysin infinitorum*, un texto acerca de las funciones matemáticas publicado en 1748, y la *Institutiones calculi differentialis*, que se publicó en 1755 y que trata acerca del cálculo diferencial.

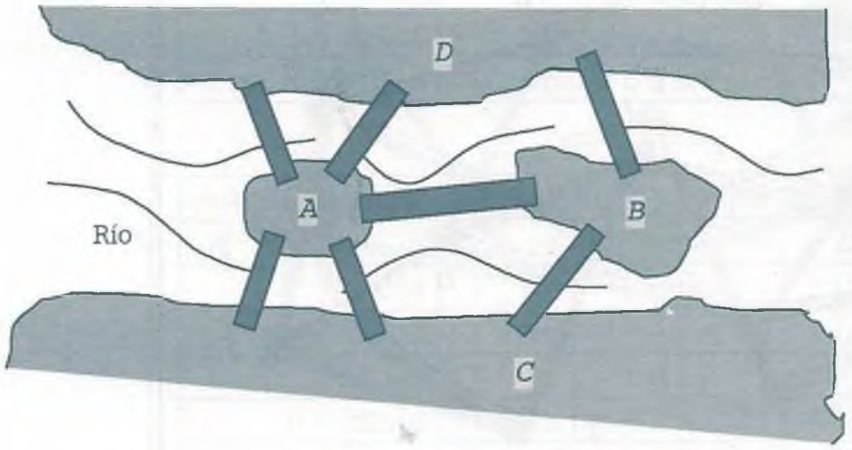
En 1735 Euler resolvió el problema conocido como el *problema de los puentes de Königsberg*, y con esta solución estableció lo que se considera como el primer teorema de la teoría de grafos así como el nacimiento de la topología.

Topología

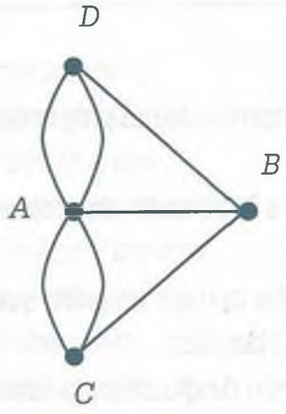
En términos coloquiales, la topología es un área de la matemática que estudia las propiedades de los objetos que no cambian cuando éstos se deforman o se estiran. Un ejemplo de este tipo de deformación es la que se puede hacer sobre una taza de goma para obtener una dona:

(continúa)

Uno de los primeros resultados de la teoría de grafos fue el que obtuvo Leonhard Euler en el siglo XVIII al resolver el problema de los puentes de Königsberg. Este problema consiste en recorrer 7 puentes que conectan 4 porciones de tierra, bajo la condición de pasar por cada puente una sola vez. En la siguiente figura se muestra la forma en que están distribuidos los puentes:



Euler representó este problema por medio de una figura como la siguiente



y la llamó "grafo". A las porciones de tierra representadas por un punto las llamó "vértices", a los puentes representados por líneas les dio el nombre de "aristas" y al número de líneas que salen o entran a un vértice le llamó "orden del vértice", el cual más tarde se llamó valencia.

Después de analizar el problema, Euler llegó a la conclusión de que es imposible obtener un itinerario que salga de un vértice y regrese a él pasando por todas las aristas solamente una vez. Según Euler, si el vértice donde se inicia y termina el recorrido es el mismo, entonces dicho vértice debe ser de valencia par ya que por un puente se sale y por otro diferente se debe de regresar. Lo mismo ocurre con todos los demás vértices, ya

que debe estar contemplada la entrada y la salida, por lo tanto Euler estableció que "en un grafo, únicamente se puede establecer un ciclo que pase por todas las aristas sólo una vez si todos los vértices tienen valencia par".

Los grafos son representaciones de las redes, y por medio de ellos se puede expresar en forma visual y sencilla la relación entre elementos de distinto tipo, por ejemplo se pueden usar para representar la estructura de una empresa en lo que se conoce como "organigrama", o bien para modelar una red eléctrica, telefónica, de carreteras, de agua potable, de alcantarillado, etcétera. Los vértices pueden ser postes, transformadores, teléfonos, ciudades, centrales telefónicas, válvulas, registros, y las aristas que tienen relación entre esos vértices pueden ser cables, tubos y carreteras, entre otras cosas. Por medio de la teoría de grafos, se pueden aprovechar mejor los recursos eliminando conexiones redundantes y reduciendo costos y distancias.

En computación los grafos se utilizan para mostrar las relaciones entre archivos (en las bases de datos), entre registros (en la estructura de datos), entre computadoras y entre redes como lo hace la red internet.

Los grafos son relaciones, como las que se expusieron en el capítulo anterior, que resultan ser muy útiles gracias a la forma en la que se les puede representar ya que es más claro ver la relación entre dos elementos en un grafo que en una matriz o en un conjunto.

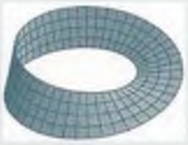
En este capítulo se exponen los conceptos fundamentales de los grafos y se presentan algunas aplicaciones de ellos en el área de la computación, sin embargo la teoría expuesta aquí se puede transportar a cualquier sistema en donde la intercomunicación es fundamental.

(continuación)

Bajo estas condiciones una taza de café y una dona son topológicamente equivalentes ya que deformando y estirando cualquiera de ellas se obtiene la otra como resultado de la transformación.

El nacimiento de la topología se suele ubicar en el año 1735, cuando Euler resolvió el problema de los puentes de Königsberg, solución que no sólo exhibe un enfoque totalmente topológico, sino que además aporta el primer invariante de la topología algebraica.

Hacia finales del siglo XVIII el punto de partida del desarrollo sistemático de la topología fue la definición rigurosa de conceptos fundamentales del análisis como función, continuidad, diferenciabilidad, así como el estudio de las nuevas geometrías no euclideanas y de objetos geométricos como la banda de Möbius (1858) que se muestra a continuación

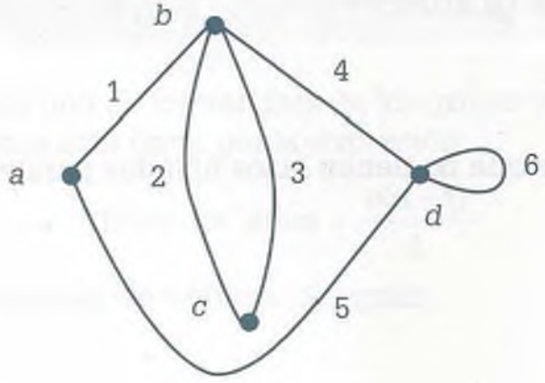


la cual fue descubierta en forma independiente por los matemáticos alemanes August Ferdinand Möbius y Johann Benedict Listing en 1858. Como se puede ver, esta banda se caracteriza por ser una superficie con un solo lado y un solo contorno además de ser un objeto no orientable.

7.2 Partes de un grafo

Un grafo (G) es un diagrama que consta de un conjunto de vértices (V) y un conjunto de lados (L).

Considérese el siguiente grafo:



A partir de esta figura se definen los siguientes elementos:

- ***Vértices (nodos)***

Se indican por medio de un pequeño círculo y se les asigna un número o letra. En el grafo anterior los vértices son V = {*a*, *b*, *c*, *d*}.

- ***Lados (ramas o aristas)***

Son las líneas que unen un vértice con otro y se les asigna una letra, un número o una combinación de ambos. En el grafo anterior los lados son L = {1, 2, 3, 4, 5, 6}.

- ***Lados paralelos***

Son aquellas aristas que tienen relación con un mismo par de vértices. En el grafo anterior los lados paralelos son: P = {2, 3}.

- ***Lazo***

Es aquella arista que sale de un vértice y regresa al mismo vértice. En el grafo anterior se tiene el lazo: A = {6}.

- ***Valencia de un vértice***

Es el número de lados que salen o entran a un vértice. En el grafo anterior las valencias de los vértices son:

Valencia (*a*) = 2
Valencia (*b*) = 4
Valencia (*c*) = 2
Valencia (*d*) = 3

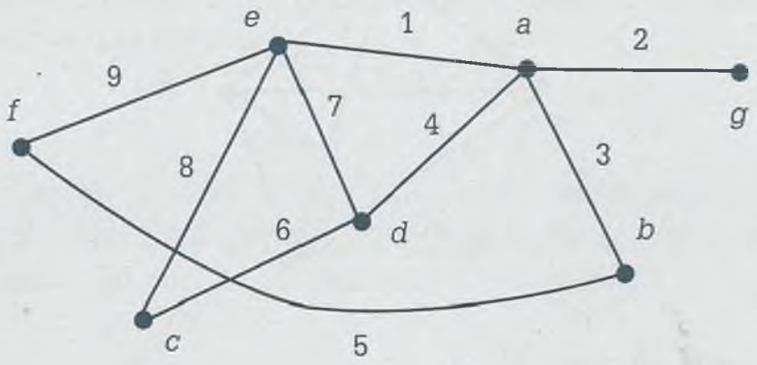
Hay que observar cómo en el caso del vértice *d* el lazo sólo se considera una vez, entrada o salida pero no ambos.

7.3 Tipos de grafos

- ***Grafos simples***

Son aquellos grafos que no tienen lazos ni lados paralelos.

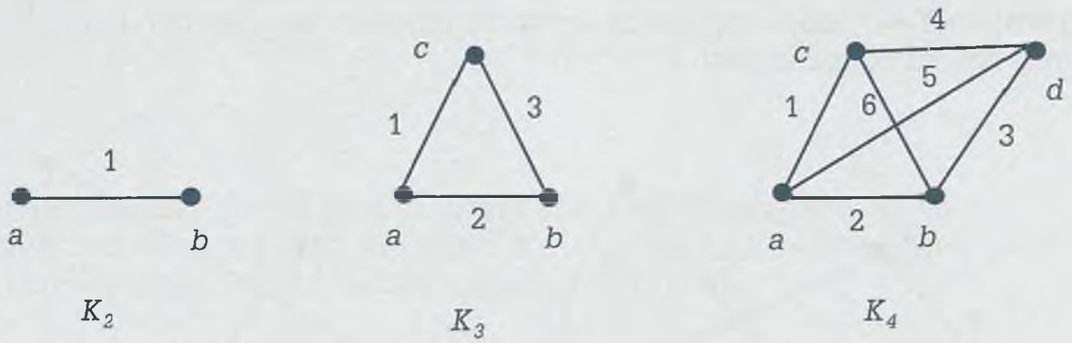
Ejemplo 7.1. En la siguiente figura se muestra un ejemplo de grafo simple.



- ***Grafo completo de n vértices (K_n)***

Es el grafo en donde cada vértice está relacionado con todos los demás, sin lazos ni lados paralelos. Se indica como K_n, en donde n es el número de vértices del grafo.

Ejemplo 7.2. En la siguiente figura se muestran tres ejemplos de grafos completos.

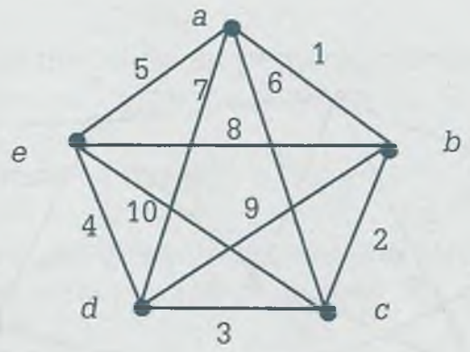


La valencia en cada uno de los vértices de los grafos completos es $(n-1)$, y el número de lados está dado por la expresión

$$\text{Núm. de lados} = \frac{n(n-1)}{2}$$

en donde n es el número de vértices del grafo.

Ejemplo 7.3. El grafo K_5 de la siguiente figura tiene



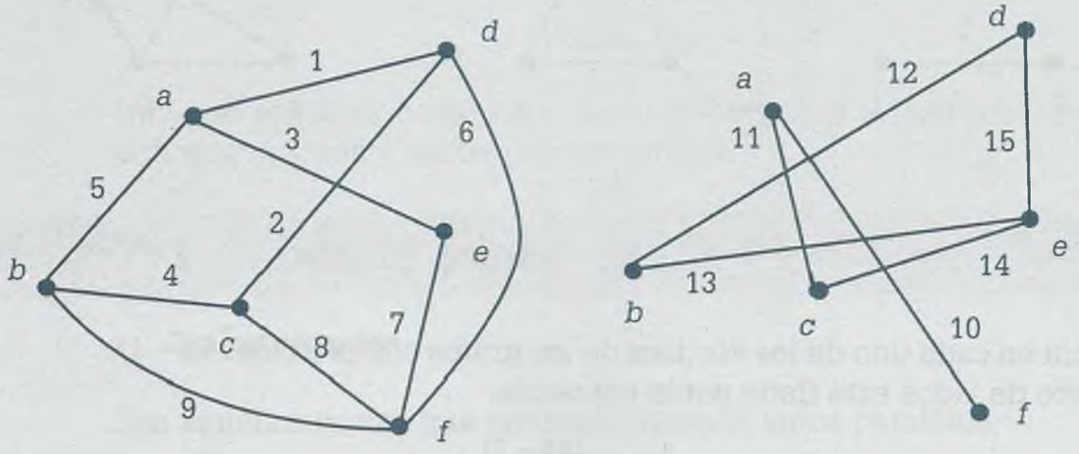
Valencia de cada vértice = (5 – 1) = 4

$$\text{Núm. de lados} = \frac{5(5-1)}{2} = 10$$

- ***Complemento de un grafo (G′)***

Es el grafo que le falta al grafo G, de forma que entre ambos forman un grafo completo de *n* vértices. Este grafo no tiene lazos ni ramas paralelas.

Ejemplo 7.4. En la siguiente figura se muestra un ejemplo de grafo G junto con su complemento G′:

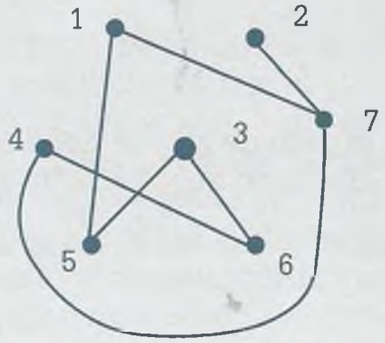


G G′

- ***Grafo bipartido***

Es el grafo que está compuesto por dos conjuntos de vértices, $A = \{a_1, a_2, a_3, \ldots, a_n\}$ y $B = \{b_1, b_2, \ldots, b_m\}$, en donde los elementos del conjunto A se relacionan con los del conjunto B, pero entre los vértices de un mismo conjunto no existe arista que los una.

Ejemplo 7.5. Sean los conjuntos de vértices $A = \{1, 2, 3, 4\}$ y $B = \{5, 6, 7\}$, con los cuales se forman los siguientes grafos:



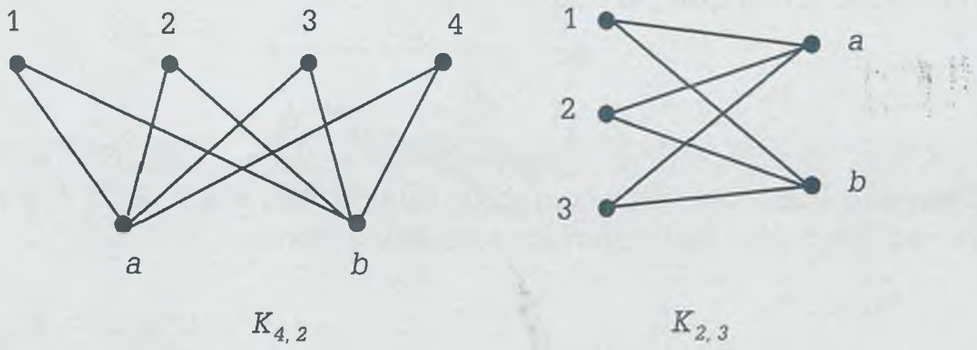
Estos dos grafos son bipartidos, ya que los elementos del conjunto A están relacionados con los del conjunto B, pero entre los elementos de un mismo conjunto no hay relación alguna.

Una forma muy sencilla de saber si un grafo es bipartido es aplicar el hecho de que nunca tiene un ciclo de longitud impar, además de que debe de cumplir con la característica mencionada anteriormente.

- ***Grafo bipartido completo ($K_{n, m}$)***

Es el grafo que está compuesto por dos conjuntos de vértices, uno de ellos $A = \{a_1, a_2, a_3, \ldots, a_n\}$ y otro $B = \{b_1, b_2, \ldots, b_m\}$, y en el que cada vértice de A está unido con todos los vértices de B, pero entre los vértices de un mismo conjunto no existe arista que los una. El grafo bipartido completo se indica como $K_{n, m}$.

Ejemplo 7.6. En la siguiente figura se muestran dos grafos bipartidos completos:



$K_{4,2}$ $K_{2,3}$

En el caso de $K_{4,2}$ se tiene que $A = \{1, 2, 3, 4\}$ y $B = \{a, b\}$, mientras que en $K_{2,3}$ se tiene que $A = \{a, b\}$ y $B = \{1, 2, 3\}$.

7.4 Representación matricial

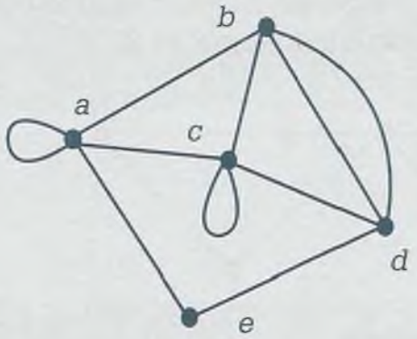
El uso de matrices para representar sistemas de ecuaciones, relaciones o grafos permite una rápida y clara manipulación de la información, así como el determinar algunas propiedades de los grafos que de otra manera serían más difíciles de obtener. Además de esto se tiene que en la computadora es más fácil el manejo de matrices, ya que se pueden tratar como arreglos o listas doblemente ligadas.

A continuación se describen las representaciones matriciales de los grafos.

- ***Matriz de adyacencia (M_a)***

Es una matriz cuadrada en la cual los vértices del grafo se indican como filas y como columnas: el orden de los vértices es el mismo que guardan las filas y las columnas de la matriz. Se coloca un 1 como elemento de la matriz cuando existe una relación entre uno y otro vértice, o bien un 0 cuando no exista relación alguna.

Ejemplo 7.7. A continuación se muestra un grafo y su matriz de adyacencia correspondiente:



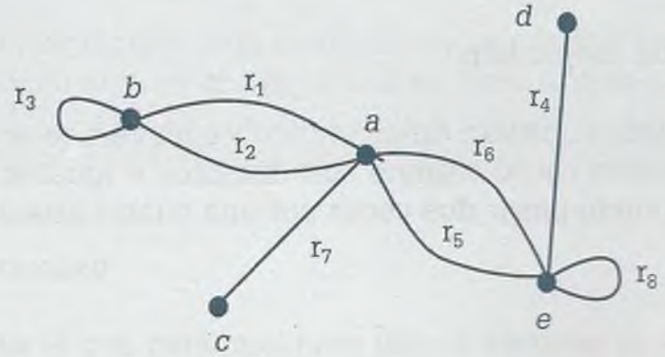
$$M_a = \begin{array}{c|ccccc} & a & b & c & d & e \\ \hline a & 1 & 1 & 1 & 0 & 1 \\ b & 1 & 0 & 1 & 1 & 0 \\ c & 1 & 1 & 1 & 1 & 0 \\ d & 0 & 1 & 1 & 0 & 1 \\ e & 1 & 0 & 0 & 1 & 0 \end{array}$$

Algo que se puede observar en la matriz de adyacencia es que no se pueden representar en ella los lados paralelos, como ocurre con el par de aristas que unen los nodos *b* y *d*. En esta matriz también la mayoría de las aristas están repetidas, como ocurre con la arista que une a los vértices *b* y *c* que tiene un 1 en la línea *b* columna *c*, pero que también tiene un 1 en la fila *c* columna *b*. Por último, los lazos, a diferencia de las aristas normales solamente se representan una sola vez. Se puede concluir que la matriz de adyacencia es buena para llevar a cabo operaciones con relaciones, pero que no permite registrar en ella toda la información del grafo.

- ***Matriz de incidencia (M_i)***

En esta matriz se colocan los vértices del grafo como filas y las aristas como columnas.

Ejemplo 7.8. Considérese el siguiente grafo junto con su matriz de incidencia correspondiente:



$$M_i = \begin{array}{c|cccccccc|c} & r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 & \\ \hline a & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 5 \\ b & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 3 \\ c & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ d & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ e & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 4 \\ \hline & 2 & 2 & 1 & 2 & 2 & 2 & 2 & 1 & \end{array}$$

En esta matriz sí es posible representar lados paralelos, como ocurre con r_1, r_2, r_5 y r_6. Al sumar los elementos de cada una de las filas se obtiene la valencia de los vértices, y al sumar las columnas es posible distinguir cuando se trata de un lazo ya que su suma es 1, como ocurre con r_3 y r_8. Cuando no se trata de lazos, el resultado de la suma es 2.

7.5 Caminos y circuitos

En un grafo se puede recorrer la información de diferente manera, lo cual implica seguir distintas rutas para llegar de un nodo del grafo a otro. A continuación se definen varios conceptos relacionados con el recorrido de un grafo, y en el ejemplo 7.9 se ilustran éstos.

- ***Camino***

Es una sucesión de lados que van de un vértice *x* a un vértice *w* (dichos lados se pueden repetir).

- ***Circuito (ciclo)***

Es un camino del vértice *w* al vértice *w*, esto es, un camino que regresa al mismo vértice de donde salió.

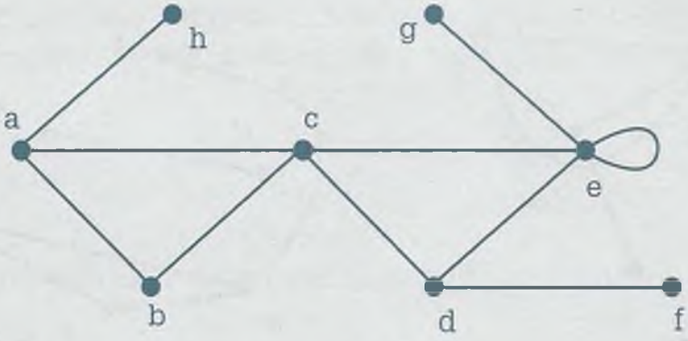
- ***Circuito simple de longitud n***

Es aquel camino del vértice *w* al vértice *w* que solamente tiene un ciclo en la ruta que sigue.

- ***Camino simple de longitud n***

Es una sucesión de lados que van de un vértice *x* a un vértice *w*, en donde los lados que componen dicho camino son distintos e iguales a *n*. Esto significa que no se puede pasar dos veces por una misma arista.

Ejemplo 7.9. Con relación al grafo



se tienen los recorridos que muestra la tabla con sus correspondientes características.

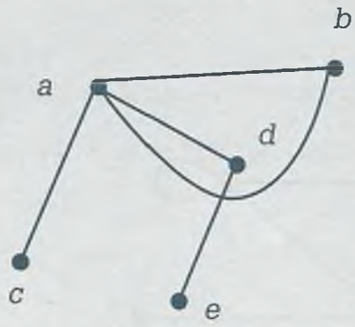
Recorrido	Camino	Camino simple de longitud n	Circuito	Circuito simple de longitud n
{*a, b, c, e, d, f*}	*	* L = 5		
{*a, h, a, b, c*}	*			
{*c, e, e, d, c, b*}	*	* L = 5		
{d, e, g, e, e, d}	*		*	
{*e, e*}	*		*	* L = 1
{*h, a, b, c, a, h*}	*		*	
{*c, d, e, c*}	*		*	* L = 3
{a, *b, c, d, e, c*}	*			
{*a, h, a*}	*		*	* L = 2
{*b, a, c, d, f*}	*	* L = 4		

Observar que todo recorrido es un camino y que la longitud del camino o del circuito es el número de vértices que se tocan menos 1.

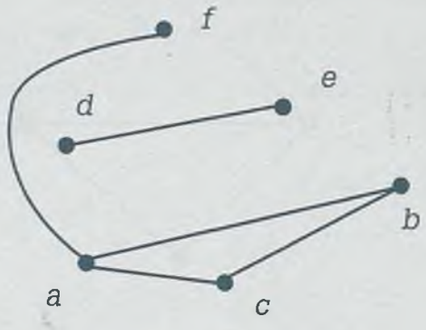
- ***Grafo conexo***

Es aquél en el que para cualquier par de vértices *w*, *x*, distintos entre sí, existe un trayecto para ir de *w* a *x*.

Ejemplo 7.10. Aquí se muestra un grafo conexo y uno no conexo.



Grafo conexo



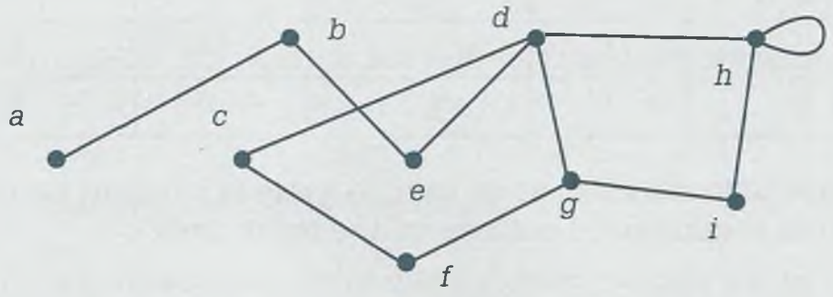
Grafo no conexo

En el grafo conexo (conectado) siempre existe un camino para ir de un vértice a otro, sin embargo en el grafo no conexo existen vértices que no están conectados y, por lo tanto, no se puede acceder a ellos. Así, en el grafo no conexo del ejemplo 7.10 no se puede tener un camino para ir del vértice *b* al *e*.

- ***Camino de Euler***

Es aquel camino que recorre todos los vértices pasando por todas las ramas solamente una vez.

Ejemplo 7.11. Considérese el siguiente grafo



Un camino de Euler es {*a, b, e, d, c, f, g, d, h, h, i, g*} o bien {*g, i, h, h, d, g, f, c, d, e, b, a*}.

Una característica importante de los grafos que tienen camino de Euler es que siempre comienza y termina en vértices que tienen valencia impar, por esta razón es imposible que en el grafo del ejemplo 7.11 un camino de Euler pueda comenzar en el vértice *f*. Por otro lado, si un grafo tiene más de dos vértices con valencia impar, entonces no puede tener un camino de Euler ya que es requisito que tenga dos y solamente dos vértices de valencia impar.

- ***Circuito de Euler***

Es aquel ciclo que recorre todos los vértices pasando por todos los lados solamente una vez.

Un grafo tiene un circuito de Euler si y sólo si es conexo y todos sus vértices tienen valencia par.

El siguiente algoritmo de Fleury permite determinar un circuito de Euler:

1) Verificar que el grafo sea conexo y que todos los vértices tengan valencia par. Si no cumple con estas condiciones entonces el grafo no tiene circuito de Euler y finalizar.

2) Si cumple con la condición anterior, seleccionar un vértice arbitrario para iniciar el recorrido.

3) Escoger una arista a partir del vértice actual. Esa arista seleccionada no debe ser "lado puente", a menos que no exista otra alternativa.

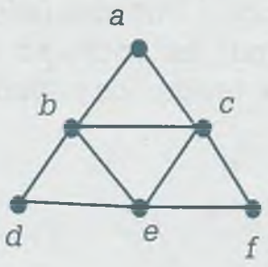
   Lado puente es aquella arista que si se elimina, los grafos pierden la propiedad de ser conexos.

4) Desconectar los vértices que están unidos por la arista seleccionada.

5) Si todos los vértices del grafo ya están desconectados, ya se tiene el circuito de Euler y finalizar. De otra manera continuar con el paso 3.

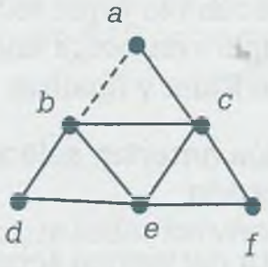
La *firma del diablo* es un juego que consiste en dibujar una figura sin levantar el lápiz del papel, partiendo de un punto y regresando nuevamente a él sin pasar dos veces por una misma arista. Este problema se puede resolver por medio del *circuito de Euler*.

Ejemplo 7.12. Determinar un circuito de Euler en el siguiente grafo.



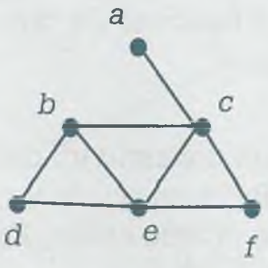
Solución

1) Se puede observar que, efectivamente, se trata de un grafo conexo y que todos sus vértices tienen valencia par.
2) Considérese que se inicia el recorrido en el vértice *a*.
3) Hay que escoger una arista a partir del vértice actual, y esa arista seleccionada no debe ser "lado puente" a menos que no exista otra alternativa.



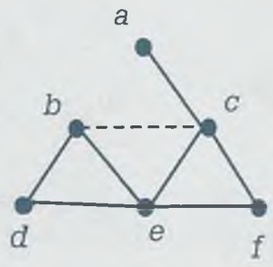
Se puede seleccionar cualquiera de las dos aristas (*a, b*) o (*a, c*), ya que ninguna es puente. Supóngase que en este caso se escoge (*a, b*), indicada con línea punteada.

4) Regístrese como parte del circuito de Euler dicha arista. Circuito de Euler: (*a, b*).
5) Desconéctense los vértices que están unidos por la arista seleccionada. Después de eliminar la arista se tiene el siguiente grafo

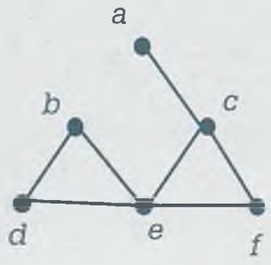


Como en este caso todavía no están desconectados todos los vértices del grafo, se continúa desde el paso 3, obteniéndose las siguientes posiciones:

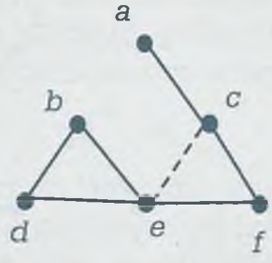
Como se ve en el siguiente grafo, del vértice actual *b* se puede seleccionar cualquiera de las aristas (*b*, *c*), (*b*, *e*) o (*b*, *d*), ya que ninguna es puente. Supóngase que se escoge (*b*, *c*) indicada con línea punteada. Circuito de Euler: (*a*, *b*, *c*).



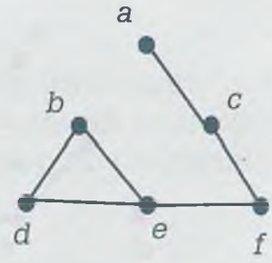
Eliminando la arista seleccionada se obtiene el siguiente grafo:



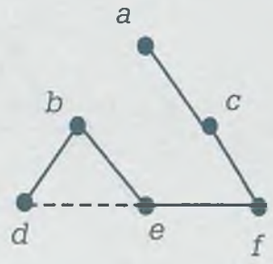
Del vértice actual *c* se puede seleccionar cualquiera de las aristas (*c*, *e*) o (*c*, *f*), ya que ninguna es puente. Supóngase que se escoge (*c*, *e*) como se indica. Circuito de Euler: (*a*, *b*, *c*, *e*).



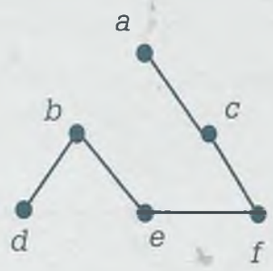
Eliminando la arista seleccionada se obtiene el siguiente grafo:



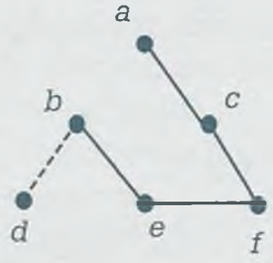
Del vértice actual *e* se puede seleccionar cualquiera de las aristas (*e*, *b*) o (*e*, *d*), pero no (*e*, *f*) porque se trata de un lado puente. Supóngase que se escoge (*e*, *d*) como se indica en el siguiente grafo. Circuito de Euler: (*a*, *b*, *c*, *e*, *d*).



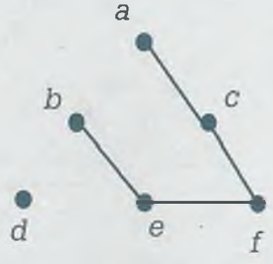
Eliminando la arista seleccionada se obtiene el siguiente grafo:



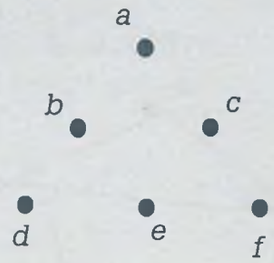
En el vértice actual *d* solamente está el lado puente (*d*, *b*), pero como ya no existe otra arista se selecciona ésta. Circuito de Euler: (*a*, *b*, *c*, *e*, *d*, *b*)



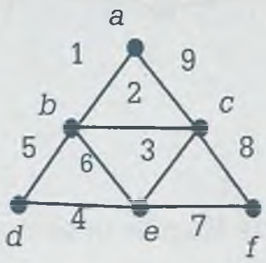
Eliminado la arista seleccionada se obtiene el siguiente grafo:



Se continúa así hasta que todos los vértices queden desconectados. Finalmente se obtiene el circuito de Euler: (*a*, *b*, *c*, *e*, *d*, *b*, *e*, *f*, *c*, *a*).



Como se mencionó, con el circuito de Euler se puede dar solución al juego de *La firma del diablo* al reconstruir el grafo, partiendo del nodo *a*, como lo muestra la numeración de las aristas:



- ***Circuito de Hamilton***

Se trata de un problema similar al del circuito de Euler, con la diferencia de que en lugar de pasar por todos los lados del grafo solamente una vez, en el circuito de Hamilton se pasa por cada vértice solamente una vez.

El problema surgió en el siglo XIX cuando Hamilton inventó un juego en donde estaban colocados nombres de ciudades en las esquinas de un dodecaedro. El juego consiste en iniciar en cualquier ciudad, viajar a lo largo de las aristas y visitar cada una de las ciudades exactamente una vez y regresar al punto de partida.

Respecto de un grafo se sabe que tiene un circuito de Euler si es conexo y todos sus vértices tienen valencia par, sin embargo no hay forma de saber con anticipación si un grafo tiene o no un circuito de Hamilton.

William Rowan Hamilton
(1805-1865)

Fue un matemático, físico y astrónomo irlandés que hizo importantes contribuciones al desarrollo de la óptica, la dinámica y el álgebra. Sin lugar a dudas su investigación mejor conocida es su descubrimiento de los cuaterniones, sin embargo su trabajo también ha sido muy importante en el desarrollo de la mecánica cuántica.

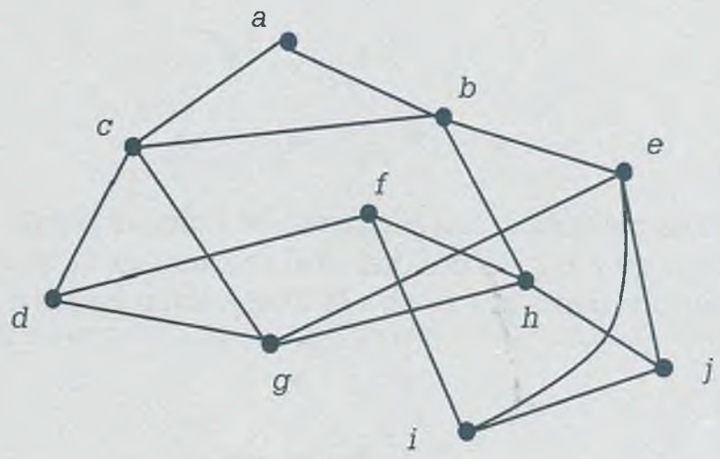
En relación con los cuaterniones se tiene que éstos se definen como el conjunto de números de la forma

$$H = \{a+bi+cj+dk \mid a, b, c, d \in R;\ i=j=k=ijk=-1\}$$

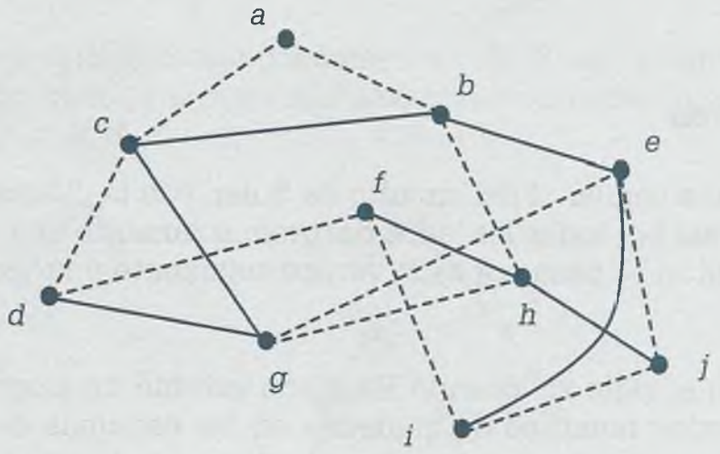
Usando la definición de suma y producto entre matrices con elementos complejos se definen la suma y el producto entre cuaterniones, y la estructura algebraica que se obtiene es la de un campo con producto no conmutativo, esto es, la de un anillo con división o campo asimétrico.

Además de su importancia intrínseca y de su utilidad en la teoría de números, los cuaterniones se aplican en electromagnetismo, teoría de la relatividad y mecánica cuántica.

Ejemplo 7.13. Determinar, si es posible, un circuito de Hamilton en el siguiente grafo.



Solución. El circuito de Hamilton es como se indica en el grafo.



Como se ve, el circuito de Hamilton es {*a*, *b*, *h*, *g*, *e*, *j*, *i*, *f*, *d*, *c*, *a*}, la línea punteada.

En general los algoritmos para obtener un circuito de Hamilton en un grafo tardan un tiempo exponencial, y están en función del número de vértices y aristas.

7.6 Isomorfismo

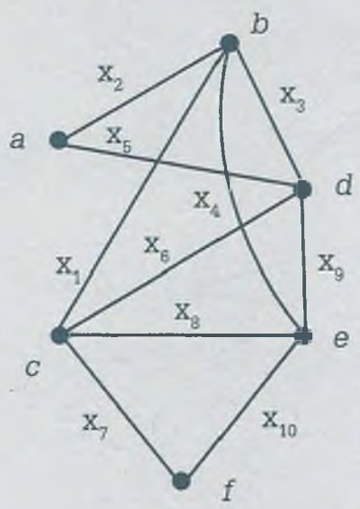
Se dice que dos grafos G_1 y G_2 son isomorfos cuando teniendo apariencia diferente realmente son iguales, porque coinciden en:

- El número de lados.
- El número de vértices.
- El conjunto de valencias.
- Ser o no conexos.
- El número de circuitos de longitud *n*.
- Tener o no circuito de Euler.

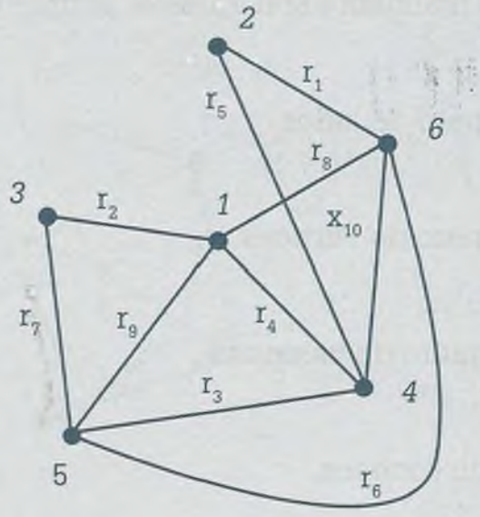
Esto implica que todos los vértices de G_1 tienen un vértice equivalente en G_2, y que todas las aristas del grafo G_1 tienen una arista equivalente en G_2. La consecuencia de esto es que con las propiedades de un vértice en G_1 como argumentos, y por medio de una función biyectiva *f*, se puede obtener un vértice en G_2 con las mismas propiedades. También teniendo las propiedades de un vértice en G_2 como argumentos, y por medio de una función biyectiva *g*, se puede obtener un vértice en G_1 con las mismas propiedades.

Por otro lado, se sabe que dos grafos G_1 y G_2 son isomorfos si y sólo si para alguna ordenación de vértices y sus aristas, sus matrices de incidencia son iguales.

Ejemplo 7.14. Determinar si los grafos siguientes G_1 y G_2 son isomorfos, haciendo coincidir sus matrices de incidencia, manteniendo una matriz estática y realizando intercambios de filas y/o columnas en la otra matriz.



G_1



G_2

Solución. Se tiene que las matrices de incidencia son:

$$M_{G_1} = \begin{array}{c|cccccccccc} & x_1 & x_2 & x_3 & x_4 & x_5 & x_6 & x_7 & x_8 & x_9 & x_{10} \\ \hline a & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ b & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ c & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ d & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 \\ e & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ f & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{array}$$

$$M_{G_2} = \begin{array}{c|cccccccccc} & r_1 & r_2 & r_3 & r_4 & r_5 & r_6 & r_7 & r_8 & r_9 & r_{10} \\ \hline 1 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 2 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 4 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 5 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 6 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{array}$$

Considerando estática la matriz de G_1 y realizando intercambios en la de G_2 se tiene:

M_{G_2} =

	r_1	r_2	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_{10}
2	1	0	0	0	1	0	0	0	0	0
1	0	1	0	1	0	0	0	1	1	0
4	0	0	1	1	1	0	0	0	0	1
6	1	0	0	0	0	1	0	1	0	1
5	0	0	1	0	0	1	1	0	1	0
3	0	1	0	0	0	0	1	0	0	0

=

	r_1	r_{10}	r_3	r_4	r_5	r_6	r_7	r_8	r_9	r_2
2	1	0	0	0	1	0	0	0	0	0
1	0	0	0	1	0	0	0	1	1	1
4	0	1	1	1	1	0	0	0	0	0
6	1	1	0	0	0	1	0	1	0	0
5	0	0	1	0	0	1	1	0	1	0
3	0	0	0	0	0	0	1	0	0	1

M_{G_2} =

	r_1	r_{10}	r_4	r_3	r_5	r_8	r_2	r_6	r_9	r_7
2	1	0	0	0	1	0	0	0	0	0
1	0	0	1	0	0	1	1	0	1	0
4	0	1	1	1	1	0	0	0	0	0
6	1	1	0	0	0	1	0	1	0	0
5	0	0	0	1	0	0	0	1	1	1
3	0	0	0	0	0	0	1	0	0	1

=

	r_1	r_4	r_{10}	r_3	r_5	r_8	r_7	r_6	r_9	r_7
2	1	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	1	1	0	1	0
4	0	1	1	1	1	0	0	0	0	0
6	1	0	1	0	0	1	0	1	0	0
5	0	0	0	1	0	0	0	1	1	1
3	0	0	0	0	0	0	1	0	0	1

M_{G_2} =

	r_5	r_4	r_{10}	r_3	r_1	r_8	r_2	r_6	r_9	r_7
2	1	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	1	1	0	1	0
4	1	1	1	1	0	0	0	0	0	0
6	0	0	1	0	1	1	0	1	0	0
5	0	0	0	1	0	0	0	1	1	1
3	0	0	0	0	0	0	1	0	0	1

=

	r_5	r_4	r_{10}	r_3	r_1	r_8	r_2	r_9	r_6	r_7
2	1	0	0	0	1	0	0	0	0	0
1	0	1	0	0	0	1	1	1	0	0
4	1	1	1	1	0	0	0	0	0	0
6	0	0	1	0	1	1	0	0	1	0
5	0	0	0	1	0	0	0	1	1	1
3	0	0	0	0	0	0	1	0	0	1

M_{G_2} =

	r_5	r_4	r_{10}	r_3	r_1	r_8	r_2	r_9	r_6	r_7
2	1	0	0	0	1	0	0	0	0	0
4	1	1	1	1	0	0	0	0	0	0
1	0	1	0	0	0	1	1	1	0	0
6	0	0	1	0	1	1	0	0	1	0
5	0	0	0	1	0	0	0	1	1	1
3	0	0	0	0	0	0	1	0	0	1

=

	r_4	r_5	r_{10}	r_3	r_1	r_8	r_2	r_9	r_6	r_7
2	0	1	0	0	1	0	0	0	0	0
4	1	1	0	0	0	1	1	1	0	0
1	1	0	1	1	0	0	0	0	0	0
6	0	0	1	0	1	1	0	0	1	0
5	0	0	0	1	0	0	0	1	1	1
3	0	0	0	0	0	0	1	0	0	1

Con lo anterior es posible comprobar que efectivamente G_1 y G_2 son isomorfos.

Es muy complicado demostrar que dos grafos son isomorfos haciendo intercambios de filas y/o columnas hasta que las matrices de incidencia coincidan, y lo es aún más a medida que aumenta el número de vértices y aristas, independientemente de si esto se hace en forma manual o en la computadora.

Debido a lo anterior, en lugar de hacer coincidir las matrices lo que se hace es una comparación de las propiedades principales de los grafos de forma que si coinciden en todas ellas se concluye que son isomorfos, pero si existe alguna diferencia (ya sea en el número de vértices, en el conjunto de valencias, o si uno de ellos es conexo y el otro no, o si uno tiene circuito de Euler y el otro no, o bien si uno de ellos tiene más circuitos de longitud *n* que el otro), esto es causa suficiente para determinar que los grafos no son isomorfos.

En la siguiente tabla se muestran las propiedades en donde debe haber coincidencia entre los grafos G_1 y G_2 anteriores para que se consideren isomorfos.

Propiedad	G_1	G_2	Observación
Núm. de vértices	6	6	
Núm. de lados	10	10	
Valencias	2, 4, 4, 4, 4, 2	4, 2, 2, 4, 4, 4	Coinciden en el mismo número de vértices de valencias 2 y 4.
Conexo	Sí	Sí	Ya que para cualquier par de vértices se puede encontrar un camino.
Camino de Euler	No	No	Ya que todos los vértices tienen valencia par.
Circuito de Euler	Sí	Sí	Ya que todos los vértices tienen valencia par y se trata de grafos conexos.
Circuitos de longitud *n*. (En este caso de longitud 3.)	6 a, b, d, a b, e, c, b b, d, c, b b, d, e, b c, d, e, c c, e, f, c	6 1, 3, 5, 1 1, 6, 4, 1 1, 4, 5, 1 1, 5, 6, 1 2, 4, 6, 2 4, 5, 6, 4	En lugar de tener longitud 3, se pudo ver cuántos circuitos tienen longitud 4. Pero en cualquier caso debe coincidir.

Es posible observar que en el caso de valencias, la coincidencia debe ser en el número de nodos de cierta valencia, y cuando se trata del número de circuitos de longitud *n* deben coincidir en el número y la longitud. Por ejemplo, si uno de ellos tiene un circuito de longitud 5 y el otro no, eso es causa suficiente para concluir que no son isomorfos. Las propiedades de "Camino de Euler" y "Circuito de Euler" son relativamente fáciles de determinar. Sin embargo, no es fácil determinar si tienen o no un circuito de Hamilton. Si es posible determinar de una forma relativamente fácil si dos grafos tienen o no circuito de Hamilton sería recomendable hacerlo, pero en caso contrario, con que sean iguales en las características que contiene la tabla anterior es suficiente.

7.7 Grafos planos

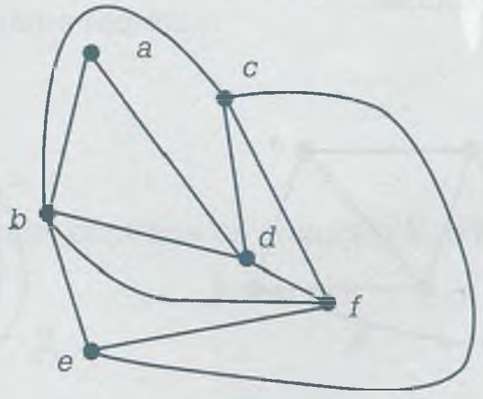
Un grafo plano es aquel que se puede dibujar en un solo plano y cuyas aristas no se cruzan entre sí.

Por otro lado, la ecuación de Euler

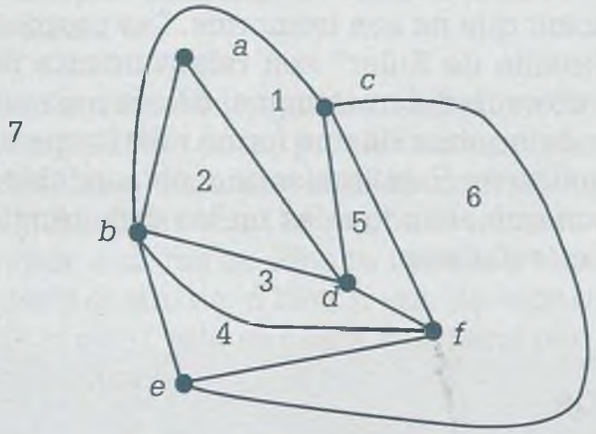
$$A = L - V + 2$$

en donde A = número de áreas, L = número de lados y V = número de vértices, es válida para un grafo plano y conexo.

Ejemplo 7.15. El siguiente grafo es un ejemplo de grafo plano y conexo:



Se consideran áreas a todas aquellas secciones cerradas {1, 2, 3, 4, 5, 6} y a la sección que rodea el grafo {7}, como se muestra en la siguiente figura.

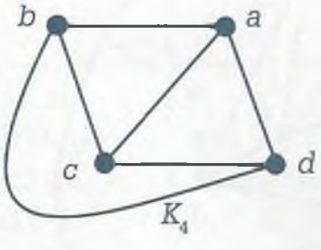


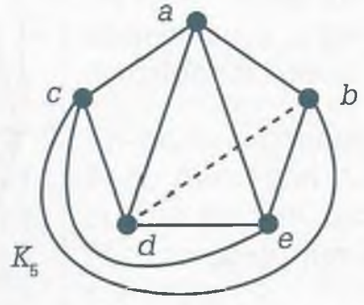
De aquí se tienen los valores A = 7, L = 11 y V = 6, los cuales satisfacen la fórmula de Euler: 7 = 11 – 6 + 2.

Otra propiedad importante de un grafo plano es que cada lado es frontera máximo de dos áreas. Así en el grafo del ejemplo 7.15 se tiene que el lado *c-f* es frontera de las áreas 5 y 6, y el lado *b-c* lo es de las áreas 1 y 7.

De acuerdo con lo anterior, si se tiene un grafo en el que la igualdad A = L – V + 2 no se cumple o bien uno de los lados es frontera de más de dos áreas, entonces con esto es más que suficiente para establecer que el grafo considerado no es plano.

Existen grafos importantes que se vieron anteriormente, por ejemplo K_4 es un grafo plano ya que se puede dibujar sin que sus aristas se crucen, pero K_5 es un grafo no plano ya que no hay forma en que por lo menos un par de aristas se crucen.

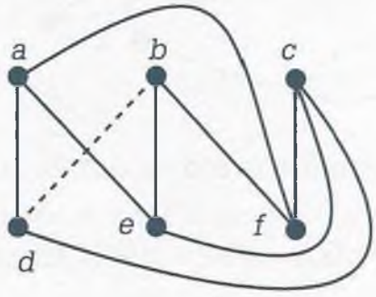




En el grafo K_4 es fácil observar que cada uno de los lados es frontera de máximo dos áreas y que la ecuación de Euler se cumple ya que A = 4, L = 6 y V = 4.

En relación con K_5 se observa que si el grafo se pudiera dibujar en forma plana, la línea punteada sería frontera de las áreas (*b, d, e, b*), (*a, b, d, a*) y (*a, b, d, c, a*). Incluso es complicado delimitar las áreas.

Otro grafo importante que no es plano, es el grafo bipartido completo $K_{3,3}$ que se muestra a continuación:



Aquí se observa que no es posible dibujar en forma plana un $K_{3,3}$.

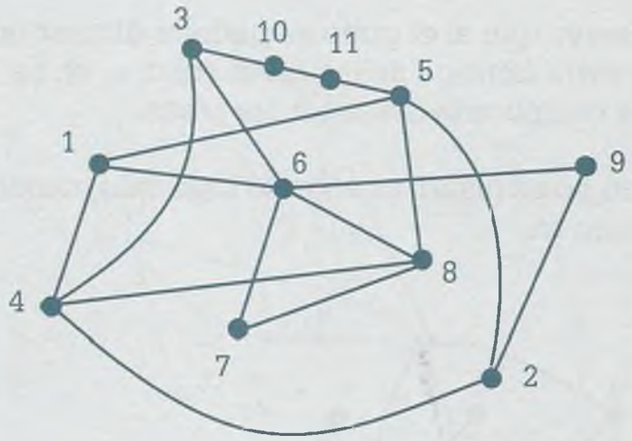
Entre más grande y complejo sea el grafo, es más difícil identificar las áreas y también es más complicado verificar si efectivamente cada uno de sus lados es frontera de máximo dos áreas adyacentes, por lo tanto, prácticamente es imposible determinar si la ecuación de Euler se cumple y en consecuencia es difícil determinar si el grafo es plano o no.

En lugar de determinar si la ecuación de Euler se cumple y si cada uno de los lados es frontera de máximo dos áreas, se utilizan $K_{3,3}$ y K_5 como patrones para demostrar que otros grafos más complejos no son planos. Esto es posible eliminando lados hasta encontrar oculto un grafo no plano $K_{3,3}$ o K_5 como lo establece el siguiente teorema:

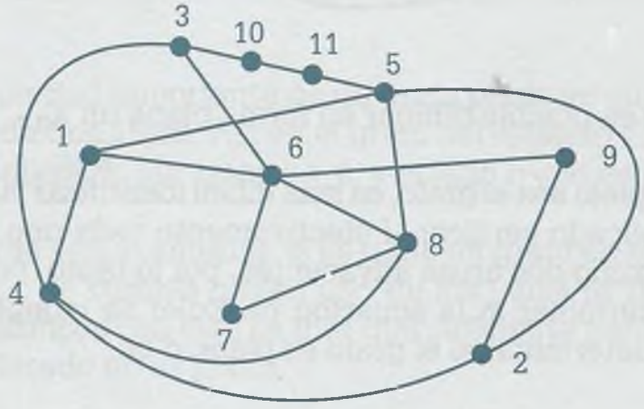
Teorema de Kuratowski

Un grafo G es plano si y sólo si no contiene un subgrafo K_5 o $K_{3,3}$.

Ejemplo 7.16. Si se desea dibujar en forma plana el siguiente grafo



una buena aproximación es el siguiente grafo en donde se han movido de lugar los lados 3-4, 5-2 y 4-8:

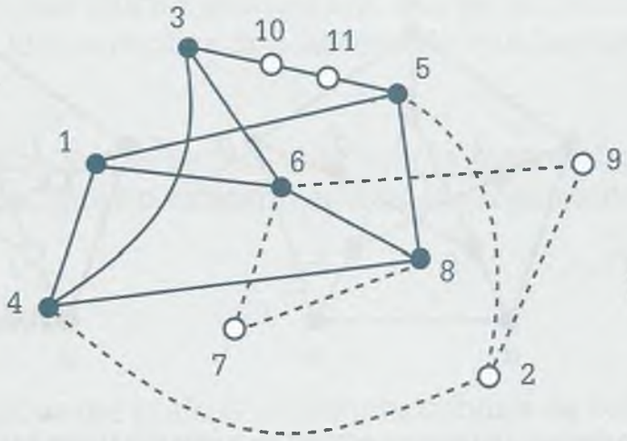


Sin embargo, el grafo aún no es plano ya que se cruzan dos pares de lados.

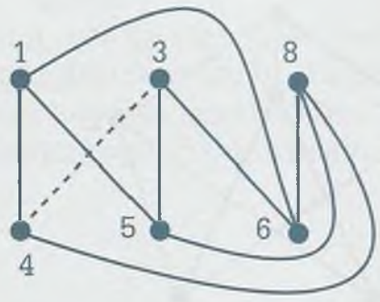
Es posible que cambiando de lugar los vértices (algo que también es permitido) se pueda lograr que sólo se crucen un par de lados, pero nunca se podrá dibujar en forma plana ya que el grafo contiene en su interior un grafo $K_{3,3}$ que se sabe que no es plano. Esto se puede observar si se eliminan los lados punteados 5-2, 6-9, 7-8, 2-4, 2-9 y 6-7, y los vértices 2, 7 y 9 de acuerdo con la siguiente figura.

Cuando se busca si G contiene un grafo no plano en su interior como $K_{3,3}$ y K_5 se eliminan los vértices de valencia dos, como fue el caso de los nodos 10 y 11 pero no necesariamente se elimina la arista o bien pueden eliminarse las aristas, como ocurre con los vértices 9 y 7. También es válido

eliminar todas las aristas y los vértices que permitan ver claramente que el grafo G contiene en su interior un grafo $K_{3,3}$ y K_5 y que por lo tanto no es plano como se muestra en la siguiente figura



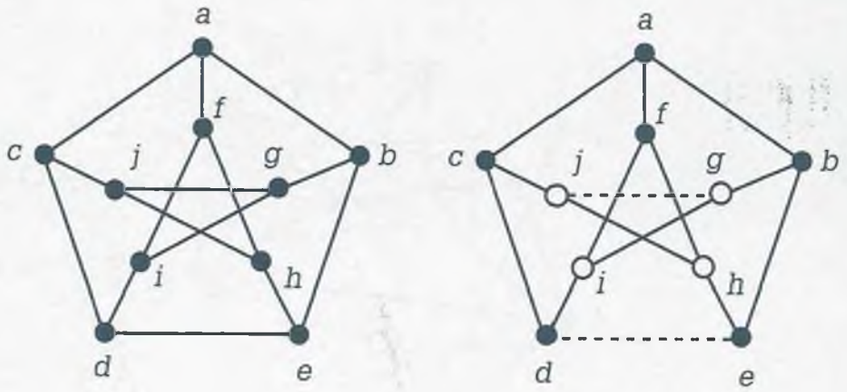
Dibujando el grafo con los vértices y las aristas que se conservan, se puede observar claramente que se trata de un grafo $K_{3,3}$, como se muestra a continuación:



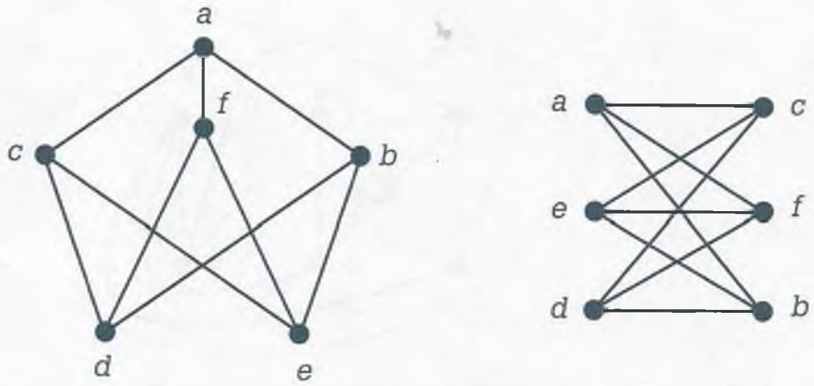
Es obvio que si un grafo tan pequeño como $K_{3,3}$ no es plano, menos lo va a ser uno más grande que contenga dentro de él a un $K_{3,3}$.

Hay que tener presente que para determinar si un grafo tiene en su interior un grafo K_5, éste deberá tener cinco o más vértices y cada uno de ellos una valencia de cuatro o más. Así mismo, para que un grafo pueda tener en su interior un $K_{3,3}$ como mínimo debe contar con seis vértices y cada uno de ellos al menos debe tener valencia tres, además de que en el K_5 todos los vértices están relacionados entre ellos y en el $K_{3,3}$ hay dos conjuntos de vértices en donde los de un conjunto están relacionados con los del otro y entre elementos de un mismo conjunto no existe lado que los una.

El siguiente grafo se conoce como de Petersen y se trata de un grafo no plano porque contiene dentro de él un subgrafo $K_{3,3}$ como se muestra a continuación



Eliminando las aristas punteadas y los vértices blancos se tiene el siguiente subgrafo $K_{3,3}$ que se sabe que no es plano, por lo tanto el grafo de Petersen tampoco es plano:



7.8 Coloración de grafos

Sea G(V,A) un grafo y sea C un conjunto de colores. La coloración de los vértices V del grafo usando un color del conjunto C se encuentra dada por la función

$$f: V \rightarrow C \text{ tal que } \forall\ v_1, v_2 \in V \text{ adyacentes}$$

$$f(v_1) \neq f(v_2)$$

Esto significa que cuando se lleva a cabo la coloración cada par de vértices adyacentes v_1 y v_2 del grafo deberán estar iluminados con un color diferente.

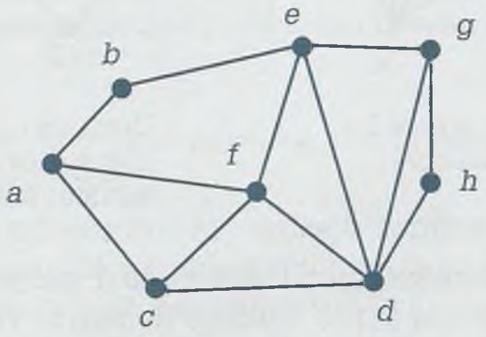
A cada vértice del conjunto V, la función f le asocia un color del conjunto C. Es importante mencionar que no es necesario que se utilicen todos los colores del conjunto C, lo que implica que se trata de una función que no es suprayectiva.

En la coloración de grafos lo que se busca es usar la menor cantidad de colores posible, cuidando que no existan vértices adyacentes del mismo color.

7.8.1 Número cromático

Se llama número cromático del grafo G al número mínimo de colores con que se puede colorear un grafo, cuidando que los vértices adyacentes no tengan el mismo color. El número cromático se indica de la siguiente como $X(G)$.

Ejemplo 7.17. Considérese que se desea iluminar el siguiente grafo G y que se dispone para ello del conjunto de colores $C = \{1, 2, 3, 4, 5\}$, teniendo en cuenta que vértices adyacentes no deben tener el mismo color y que se debe usar el menor número de colores posible para encontrar el número cromático.

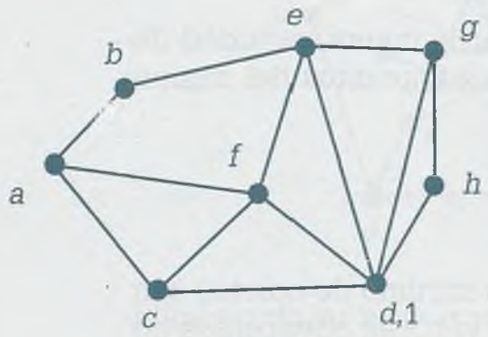


G

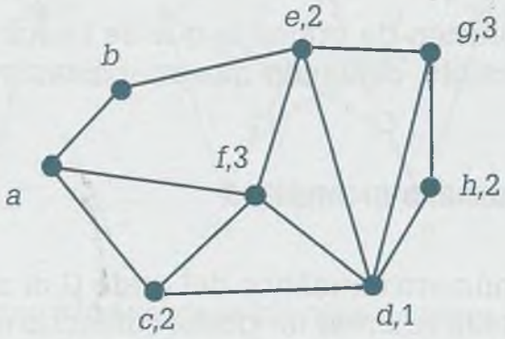
Solución

Se selecciona el vértice que tenga mayor valencia y se ilumina de un color de los disponibles en el conjunto C, después se colorean los vértices adyacentes *a él con un color diferente*, verificando que no se presenten

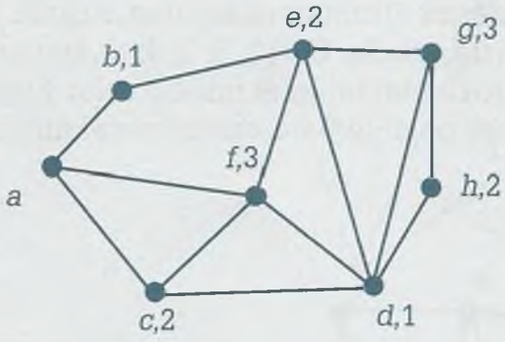
vértices adyacentes iluminados con un mismo color. Luego se toma el vértice que tenga mayor valencia de este grupo de vértices coloreados en segunda opción y se colorean sus vértices adyacentes, cuidando que no existan vértices adyacentes del mismo color y usando el menor número de colores posibles para iluminar el grafo. Lo que se obtiene son los siguientes estados del grafo:



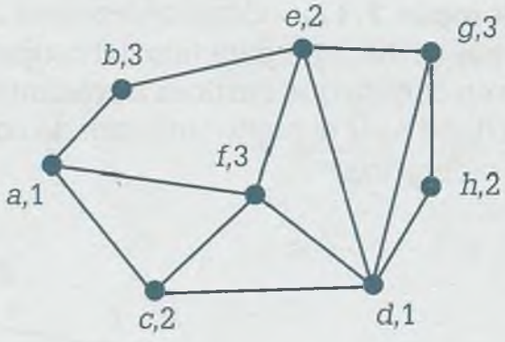
Después de colorear el vértice d con el color 1



Después de colorear los vértices adyacentes a d



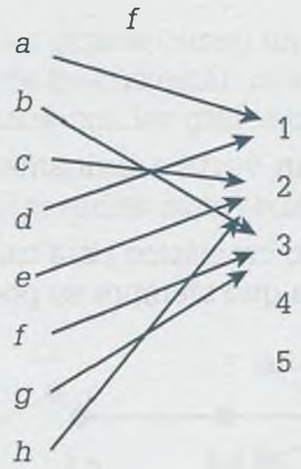
Después de colorear los vértices adyacentes faltantes a e



Después de hacer los ajustes necesarios y colorear los vértices faltantes

En este ejemplo primero se iluminó el vértice d, por ser el que tiene valencia mayor (en caso de dos o más vértices de mayor valencia se selecciona cualquiera de ellos). Después se colorearon los vértices adyacentes a él cuidando que no existan vértices con el mismo color, por ejemplo si el vértice c ya está iluminado con el color 2, el vértice f no puede iluminarse también con el color 2 o el color 1 porque es adyacente al vértice c y al vértice d. Nuevamente se seleccionó un vértice de mayor valencia que aún no tenía todos sus vértices adyacentes iluminados, en este caso el vértice e, y se coloreó el vértice adyacente faltante b con el color 1. Finalmente fue necesario realizar los ajustes necesarios usando la menor cantidad de

colores para lo cual fue necesario cambiar el color del vértice b al color 3 para que el vértice a fuera iluminado con el color 1. Se encontró que el número máximo de colores utilizado para iluminar el grafo G son 3, por lo tanto su número cromático es X(G)=3. Hay que observar que no fue necesario utilizar todos los colores del conjunto C de forma que la coloración usando la función f: V → C se puede representar de la siguiente manera



Aquí se observa que f: V → C no es una función suprayectiva, considerando que $\forall\ v_1, v_2 \in V$ adyacente $f(v1) \neq f(v_2)$.

La coloración de grafos es un problema NP-Computable y esto significa que no hay procedimientos eficientes para llevar a cabo esta tarea, sin embargo existen métodos aproximados que pueden dar buenos resultados. Uno de ellos es el siguiente:

1) Seleccionar el vértice de mayor valencia v e iluminarlo con un color cualquiera del conjunto C.
2) Colorear los vértices adyacentes al vértice v verificando que no existan vértices adyacentes del mismo color. En caso de ser necesario llevar a cabo intercambio de colores con la finalidad de usar la menor cantidad de ellos. Si ya están coloreados todos los vértices del grafo finalizar, en caso contrario continuar con el paso 3.
3) Seleccionar el vértice v de mayor valencia que ya esté coloreado y que todavía tenga vértices adyacentes sin colorear. Regresar al paso 2.

Se recomienda colorear del mismo color tantos vértices como sea posible e iluminar al mismo tiempo los vértices que compartan nodos vecinos.

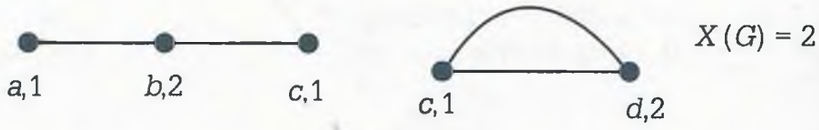
7.8.2 Características del número cromático

El número cromático posee las siguientes siete características fundamentales:

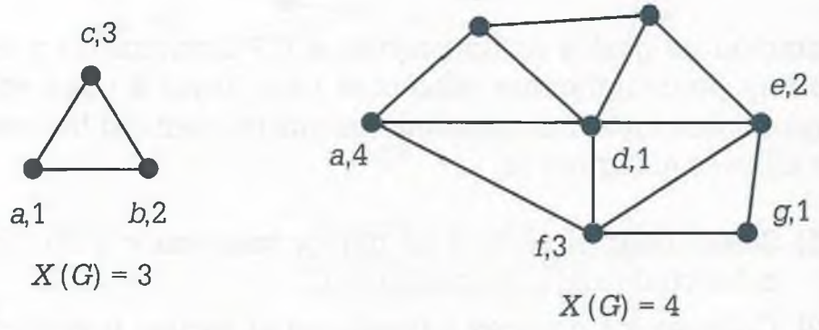
a) Un grafo G tiene número cromático X(G)=1 si y sólo si no tiene aristas:

ya que un vértice únicamente puede iluminarse de un solo color.

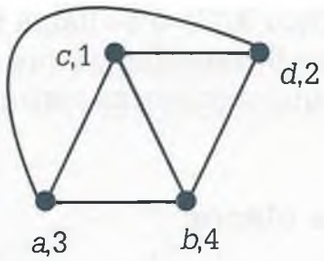
b) El número cromático para un camino o un ciclo de longitud 2 es X(G)=2 ya que siempre se podrán alternar los colores:



c) Si el grafo G tiene un ciclo de longitud impar entonces X(G) ≥ 3:



d) El número cromático del grafo completo K_n es $X(K_n)=n$, considerando que la característica de este tipo de grafo es que todos los vértices son adyacentes entre sí. Por jemplo K_4 tiene número cromático 4, porque es el menor número de colores con el que se puede iluminar, cuidando que vértices adyacentes no tengan el mismo color.

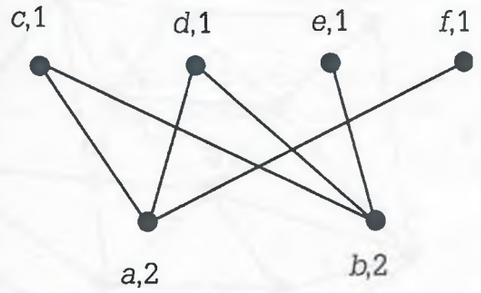


K_4

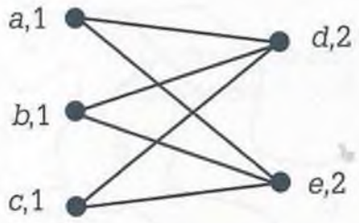
$X(K_4) = 4$

e) En general la mayoría de los grafos tienen un número cromático $X(G) \leq n$ porque se entiende que no están relacionados todos los vértices entre sí, como ocurre con los grafos completos de n vértices K_n.

f) Los grafos bipartidos o bipartidos completos ($K_{n,m}$), tienen un número cromático X(G)=2.

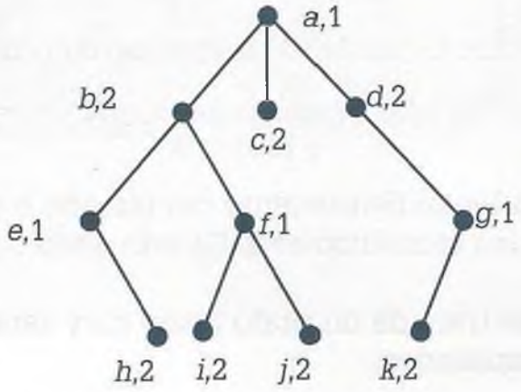


Grafo bipartido



Bipartido completo $K_{2,3}$

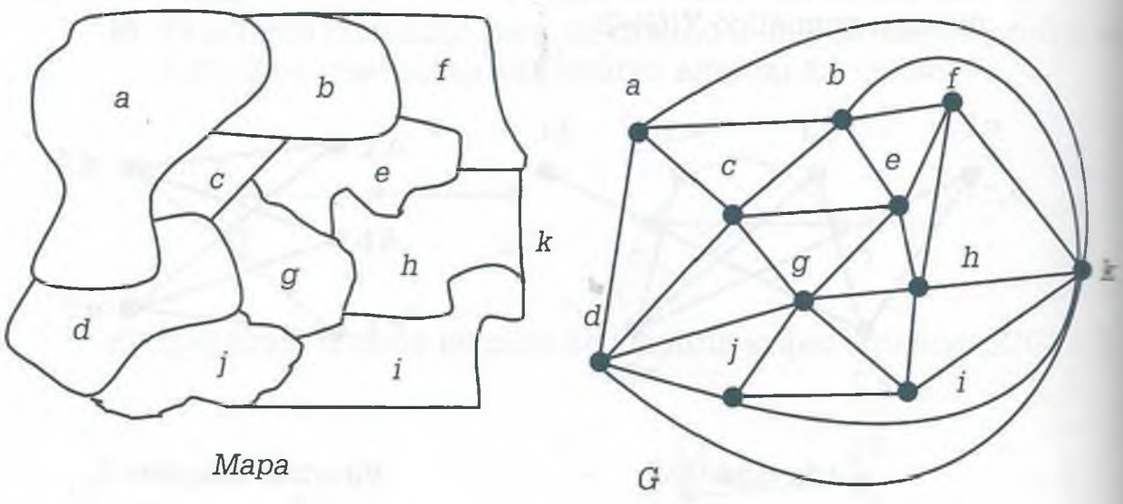
g) Todos árboles de cualquier orden tienen número cromático X(G) = 2 o bien se dice que son 2-colorables.



Un grafo G con número cromático X(G)=5 se llama 5-colorable y se infiere que si es 5-colorable es también 6-colorable ya que si un grafo G se puede iluminar con 5 colores es obvio que también se podrá colorear con 6, aunque su número cromático es 5.

7.8.3 Coloración de grafos planos

Todo mapa puede ser representado por un grafo plano, en donde cada parte del mapa representa un vértice. En el grafo también se debe incluir la parte que rodea al mapa como un vértice adicional, ya que es adyacente a varias partes del mapa. Dos partes del mapa que son vecinas se representan como vértices adyacentes en el grafo. A continuación se muestra un pequeño mapa con su correspondiente grafo plano.



Mapa *G*

En relación con el coloreado de un grafo se tiene el siguiente teorema:

Teorema de los cuatro colores. (Appel y Haken)

Cualquier grafo plano G puede ser coloreado con cuatro colores diferentes.

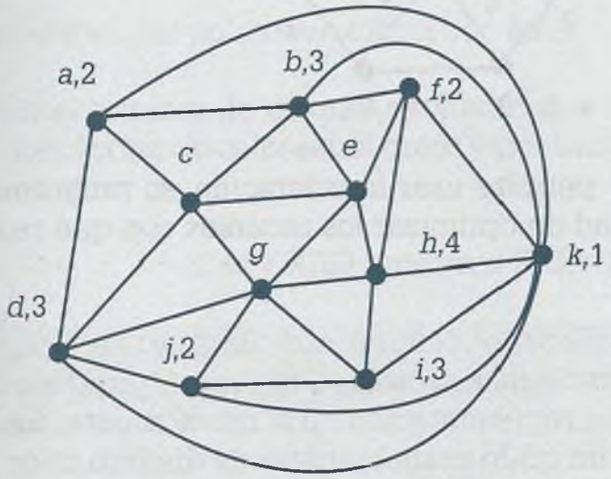
Esto significa que independientemente del tamaño o complicación de un grafo plano, su número cromático es $X(G) \leq 4$.

Será menor de 4 si se trata de un grafo plano muy sencillo y máximo será de 4 para grafos complicados.

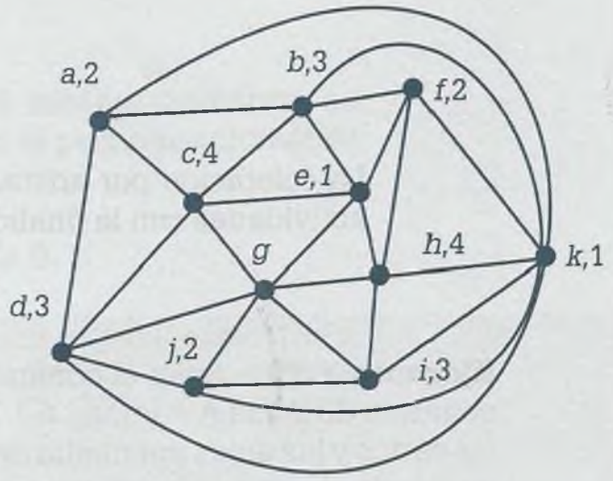
Problema de los cuatro colores

Este problema surgió a mediados del siglo XIX cuando Francis Guthrie (1831-1899), luego de colorear el mapa de Inglaterra con 4 colores, planteó la cuestión de si todos los grafos planos se podrían colorear con solamente 4 colores. El problema permaneció sin ser resuelto por más de cien años hasta que en 1976 Kenneth Appel y Wolfgang Haken demostraron con ayuda de una computadora que efectivamente todo grafo plano tiene un número cromático menor o igual a 4.

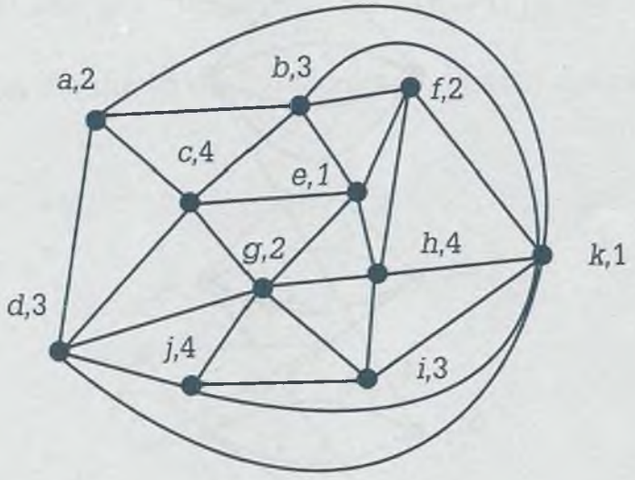
Ejemplo 7.18. Para colorear y obtener el número cromático del grafo plano G que se obtuvo a partir del mapa anterior, se procede de la siguiente manera:



Después de colorear el vértice k y sus vértices adyacentes.



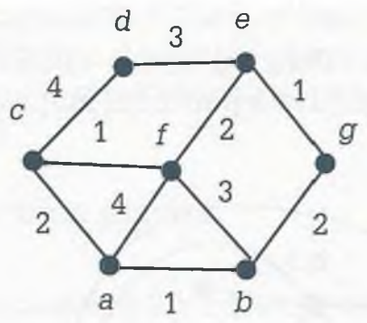
Después de colorear los vértices adyacentes restantes de b.



Cambiando de color j y coloreando el vértice g

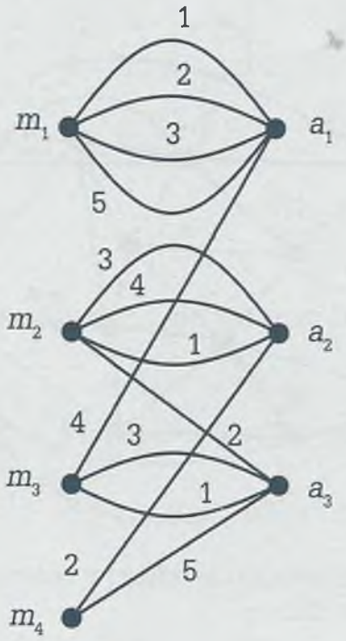
Por tanto se tiene que X(G) = 4.

También es posible la coloración de los grafos iluminando las aristas en lugar de los vértices. El siguiente grafo tiene iluminadas sus aristas con los colores {1, 2, 3} verificando que dos aristas de un mismo color no incidan sobre un mismo vértice.



La coloración por aristas permite usar la coloración en programación de actividades con la finalidad de optimizar los recursos con que se dispone.

Ejemplo 7.19. Sean el conjunto de maestros $M = \{m_1, m_2, m_3, m_4\}$ y el conjunto de aulas $A = \{a_1, a_2, a_3\}$. La representación de la relación entre los maestros y las aulas por medio de un grafo usando aristas de distinto color, en donde el color de la arista indica la hora a la cual se imparte la clase, es la siguiente:



Aquí se tiene que

1: Clase de 8:00-9:00.
2: Clase de 9:00-10:00
3: Clase de 10:00-11:00
4: Clase de 11:00-12:00
5: Clase de 12:00-13:00

7.8.4 Polinomio cromático

Al número de formas en que se puede colorear el grafo G usando w colores se le llama polinomio cromático de G y se denota como P(G,w).

Las propiedades del polinomio cromático son:

a) Si el número de colores es menor que el número cromático, no hay forma de colorear el grafo y por tanto el polinomio cromático es cero:

$$\text{Si } w < X(G) \text{ entonces } P(G,w) = 0$$

b) Si G es un grafo con un solo vértice y sin aristas, entonces el número de formas en que se puede colorear ese vértice con w colores es w:

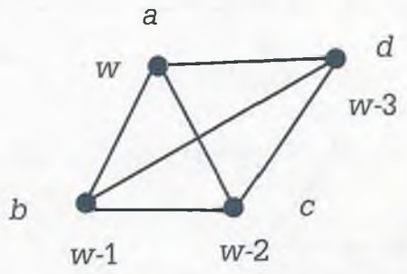
$$P(G,w) = w$$

c) Si G es un grafo completo de n vértices (K_n) entonces:

$$P(G,w) = P(K_n,w) = w(w-1)(w-2)\ldots[w-(n-1)] = w!$$

Por ejemplo, si el siguiente grafo se colorea a partir del vértice a se tiene que

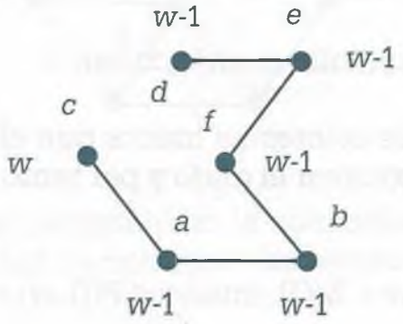
$$P(K_4,w) = w(w-1)(w-2)(w-3) = 4!$$



K_4

d) El número cromático para un camino simple es:

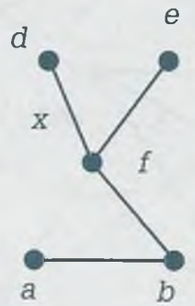
$$P(G,w) = w(w-1)(w-1)\ldots(w-1) = w(w-1)^{n-1}$$



e) Si G está integrado por subgrafos, es necesario descomponer el grafo en subgrafos cuyo polinomio cromático sea conocido. La descomposición se lleva a cabo eliminando aristas para obtener un subgrafo G_x y obtener de éste un subgrafo conocido que se llamará G'_x. En caso de ser necesario, se vuelve a descomponer G_x hasta obtener solamente subgrafos conocidos. La expresión que permite determinar el polinomio cromático de G es:

$$P(G,w) = P(G_x,w) - P(G'_x,w)$$

Ejemplo 7.20. Determinar el polinomio cromático del siguiente grafo G:

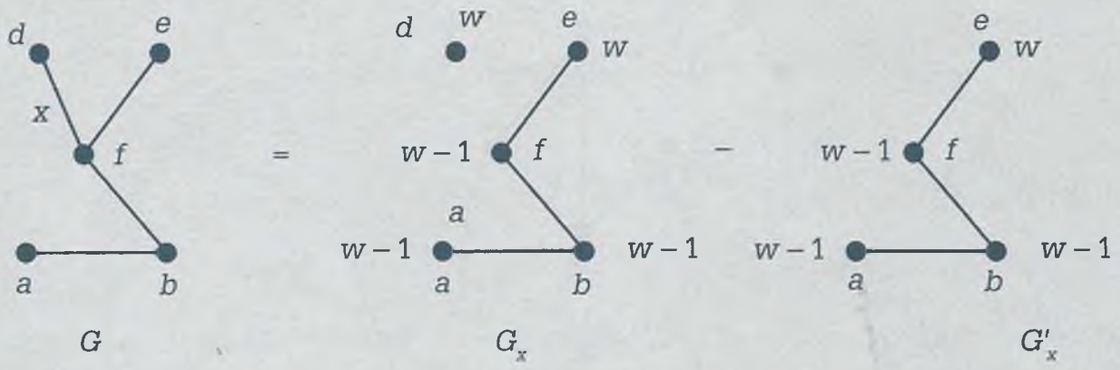


G

Solución

El polinomio cromático del grafo anterior no es conocido ya que no es un grafo integrado por un solo vértice sin aristas, tampoco es un camino simple, ni un grafo completo de n vértices K_n, por lo tanto se debe descomponer en subgrafos cuyo polinomio cromático sea conocido.

Considérese que se elimina la arista x que conecta a los vértices d y f para obtener el subgrafo G_x. Hay que observar también que G'_x es subgrafo de G_x.



Sustituyendo valores en

$$P(G,w) = P(G_x,w) - P(G'_x,w)$$

se tiene que

$$P(G,w) = w\,[w(w\text{-}1)(w-1)(w\text{-}1)] - w(w-1)(w-1)(w-1)$$

$$= w\,(w-1)(w-1)(w-1)(w-1) = w(w-1)^4$$

Con el polinomio cromático, también es posible obtener el número cromático de G. Hay que observar que para un valor de w = 1 se tiene que P(G,w) = 0 lo que quiere decir que no es posible colorear ninguna vez el grafo. Para w = 2 el número de formas en que se puede colorear G es:

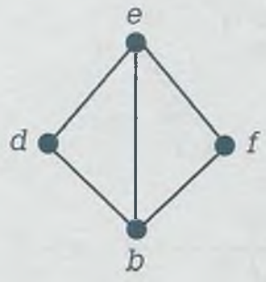
$$P(G,2) = 2(2-1)^4 = 2$$

Por lo tanto X(G) = 2 ya que es el menor valor de w con el que se puede colorear el grafo G.

Por otro lado, el número de formas que se puede colorear G con w = 5 es:

$$P(G,5) = 5(5-1)^4 = 1280$$

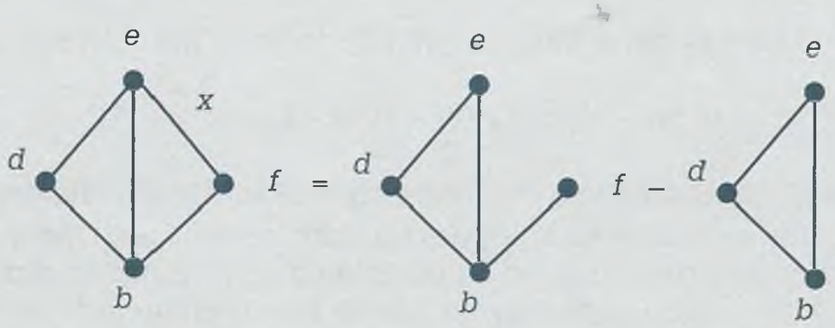
Ejemplo 7.21. Para el siguiente grafo G determinar el polinomio cromático P(G,w), el número cromático X(G) y el número de formas distintas en que se puede colorear dicho grafo si w = 5.



G

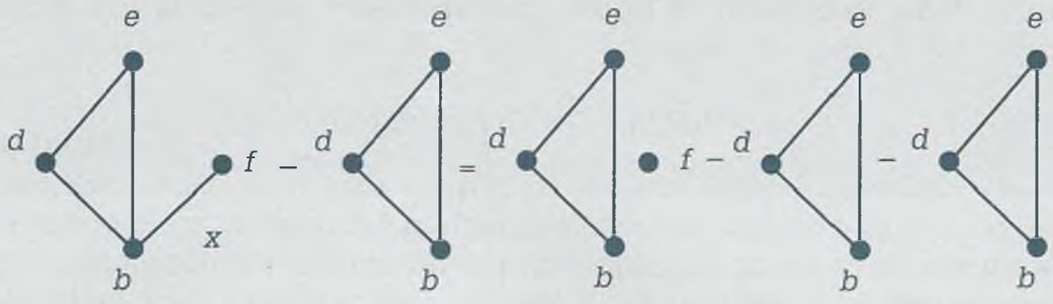
Solución

Debido a que G no es un grafo con el cual se pueda obtener directamente el polinomio cromático, es necesario descomponerlo en G_x y G'_x como se muestra en la figura.



G G_x G'_x

Como del subgrafo G_x tampoco es posible obtener su polinomio cromático, se descompone nuevamente:



G_x G'_x G_x G'_x G'_x

Hay que observar que ahora ya es posible obtener el polinomio cromático, considerando que G_x está integrado por un subgrafo completo de longitud 3 y un vértice sin aristas y que G'_x es dos veces un grafo completo K_3.

Se tiene que

$$P(G,w) = P(G_x,w) - (G'_x,w) - (G'_x,w)$$

Sustituyendo:

$$P(G,w) = w\,[w(w\text{-}1)(w-2)] - w(w-1)(w-2) - w(w-1)(w-2)$$

$$= w\,(w-1)(w-2)(w-1-1) = w(w-1)(w-2)^2$$

Hay que observar que con $w < 3$ no es posible colorear ninguna vez el grafo, por lo tanto $P(G,w) = 0$ y el valor de w mínimo con el cual se puede colorear G es $w = 3$ y por tanto el número cromático es $X(G) = 3$.

Si $w = 5$, entonces el número de maneras diferentes en que es posible colorear G es:

$$P(G,5) = 5(5-1)(5-2)^2 = 180$$

7.9 Aplicaciones de los grafos

Actualmente nada funciona de manera individual, por el contrario existe una relación entre los distintos elementos que integran un sistema, trabajo o equipo.

Puesto que los grafos permiten modelar todo aquello que está relacionado, es evidente que su aplicación es muy amplia y variada.

7.9.1 Reconocimiento de patrones mediante grafos de similaridad

Los grafos de similaridad permiten agrupar información con características semejantes. Esto implica formar subgrafos en donde los vértices de un subgrafo están relacionados entre sí, pero no tienen relación con los vértices del otro subgrafo, ya que no son similares. Una aplicación de este tipo de grafos se encuentra en el reconocimiento de patrones, en donde se agrupa información con propiedades muy parecidas de tal manera que se puedan detectar enfermedades como el cáncer, al agrupar conjuntos de células que comparten características similares. Otra aplicación se encuentra en la cartografía, en donde grupos de píxeles (pequeños cuadritos de una imagen) se agrupan porque son muy parecidos y por lo tanto se consideran similares.

En este tipo de grafo se debe definir una función que permita determinar la similaridad que existe entre los vértices, principalmente la distancia entre sus características. Una función muy usada para determinar la distancia es:

$$S(P_x - P_y) = \sum_{i=1}^{n} |P_{x,i} - P_{y,i}| = |P_{x,1} - P_{y,1}| + \\ + |P_{x,2} - P_{y,2}| + \ldots\ldots\ldots + |P_{x,n} - P_{y,m}|$$

en donde $P_{x,n}$ es la propiedad n del punto P_x y $P_{y,m}$ es la propiedad m del punto P_y. Con la función anterior es posible determinar la distancia que existe entre las propiedades de los vértices x, y. Un valor grande de $S(P_x - P_y)$ indica que no existe similaridad entre los vértices x, y mientras que un valor pequeño significa que son muy parecidos o similares.

Pero además de la función anterior, es necesario un valor referencial para discriminar la información que en este caso se llamará coeficiente de inferencia *C*. El valor de *C* se puede seleccionar de acuerdo con la experiencia que se tenga en el campo, o bien por medio de una prueba piloto. Si *C* es grande, la similaridad es poco exacta, sin embargo para un valor de *C* pequeño la similaridad es muy fuerte.

Una vez que se tiene $S(P_x - P_y)$ para todos los puntos (vértices) en cuestión se forman los grafos similares por medio del siguiente criterio: si $S(P_x - P_y) \leq C$, se traza una arista entre los vértices P_x y P_y y se dice que P_x y P_y son similares. Los grafos similares forman un subgrafo y los no similares otro distinto, como se muestra en el siguiente ejemplo.

Ejemplo 7.22. La siguiente tabla muestra las características que tienen las diferentes partes de tejido de igual sección transversal de un análisis de mama, de acuerdo con "temperatura", "intensidad de color" e "inflamación".

Parte (P)	*Temperatura (T)*	*Color (Co)*	*Inflamación (I)*
1	39	48	7
2	37	42	6
3	40	47	8
4	36	41	5
5	39	49	9

Determinar los grafos de similaridad para un coeficiente de inferencia C = 5.

Solución

Lo primero que se debe de hacer es obtener la distancia entre los puntos P_x y P_y por medio de la siguiente función:

$$S(P_x - P_y) = \sum_{i=1}^{n} |P_{x,i} - P_{y,i}| = |P_{x,1} - P_{y,1}| + |P_{x,2} - P_{y,2}| + \ldots + |P_{x,n} - P_{y,m}|$$

Si x = y se tiene que

$$S(P_1 - P_1) = |P_{1,1} - P_{1,1}| + |P_{1,2} - P_{1,2}| + |P_{1,3} - P_{1,3}|$$

$$= |39 - 39| + |48 - 48| + |7 - 7| = 0$$

Se observa claramente que entre P_1 y P_1 existe una gran similaridad, es más, son iguales. Por lo tanto, no tiene caso obtener la similaridad entre puntos iguales, de forma que se da por hecho que son similares y solamente se registrarán al final en el grafo correspondiente.

Si $x \neq y$

$$S(P_1 - P_2) = |P_{1,1} - P_{2,1}| + |P_{1,2} - P_{2,2}| + |P_{1,3} - P_{2,3}|$$
$$= |39 - 37| + |48 - 42| + |7 - 6| = 9$$

$$S(P_1 - P_3) = |P_{1,1} - P_{3,1}| + |P_{1,2} - P_{3,2}| + |P_{1,3} - P_{3,3}|$$
$$= |39 - 40| + |48 - 47| + |7 - 8| = 3$$

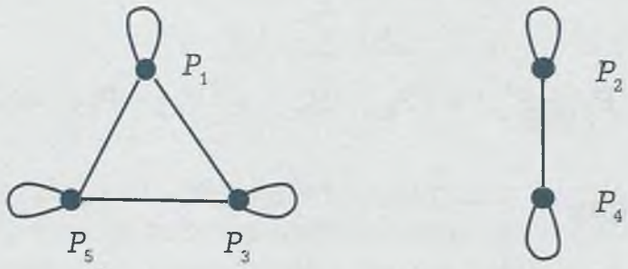
$$S(P_1 - P_4) = |P_{1,1} - P_{4,1}| + |P_{1,2} - P_{4,2}| + |P_{1,3} - P_{4,3}|$$
$$= |39 - 36| + |48 - 41| + |7 - 5| = 12$$

$$S(P_1 - P_5) = |P_{1,1} - P_{5,1}| + |P_{1,2} - P_{5,2}| + |P_{1,3} - P_{5,3}|$$
$$= |39 - 39| + |48 - 49| + |7 - 9| = 3$$

$$S(P_2 - P_3) = |P_{2,1} - P_{3,1}| + |P_{2,2} - P_{3,2}| + |P_{2,3} - P_{3,3}|$$
$$= |37 - 40| + |42 - 47| + |6 - 8| = 10$$

$$S(P_2 - P_4) = |37 - 36| + |42 - 41| + |6 - 5| = 3$$
$$S(P_2 - P_5) = |37 - 39| + |42 - 49| + |6 - 9| = 12$$
$$S(P_3 - P_4) = |40 - 36| + |47 - 41| + |8 - 5| = 13$$
$$S(P_3 - P_5) = |40 - 39| + |47 - 49| + |8 - 9| = 4$$
$$S(P_4 - P_5) = |36 - 39| + |41 - 49| + |5 - 9| = 15$$

Se considera que dos puntos son similares si $S(P_x - P_y) \leq C$ y en este caso se tiene que $S(P_x - P_y) \leq 5$, con lo cual se puede decir que el grafo está dividido en dos partes como se muestra a continuación:



Los lazos en cada uno de los vértices significan la similaridad entre ellos mismos. Con los grafos anteriores se puede considerar que las partes de tejido P_1, P_3, P_5 son similares así como P_2 y P_4 también lo son. Si se considera un mayor daño en las células cuyas características de temperatura, color e inflamación están más alejadas de los valores normales, se puede concluir que las áreas más dañadas son P_1, P_3 y P_5.

7.9.2 Determinación de la ruta más corta mediante grafos ponderados

En un grafo ponderado a las aristas se les asigna un valor al que se le llama *ponderación* y que podría representar la distancia que hay de un nodo a otro, o bien el costo de transportarse de una ciudad a otra.

Determinar la ruta más corta es un problema típico de la teoría de grafos, y consiste en encontrar el camino más corto para ir de una ciudad origen *w* a una ciudad destino *x*. Pueden existir distintas rutas para ir de un nodo a otro, pero el objetivo es encontrar la más corta o bien la más económica si es que la ponderación representa un costo.

El método más utilizado para encontrar la ruta más corta de un nodo cualquiera *w* a cualquier nodo de la red, es por medio del algoritmo de Dijkstra el cual consta de los siguientes pasos:

1. Seleccionar la ciudad origen *a*.
2. Usar una matriz que tenga como columnas el número de *iteración*, una columna para cada nodo (*a, b, c, d,...*), la columna *Actual* que se utilizará para indicar el vértice que se seleccione en cada iteración y una columna *Seleccionados* para registrar los vértices que se van seleccionado en el proceso, como se muestra en la siguiente tabla:

Iteración	*a*	*b*	*c*	*d*	...	Actual	Seleccionados

3. Colocar en la matriz la distancia que existe de la ciudad origen a ella misma (cuando se trata de encontrar la distancia de una ciudad a ella misma considerar que es 0). A todas las demás columnas se les coloca ∞ como distancia.

Iteración	*a*	*b*	*c*	*d*	...	Actual	Seleccionados
0	0	∞	∞	∞			

4. Colocar en la columna *Actual* el vértice que tenga la distancia más corta de entre todos los nodos (es obvio que en esta primera iteración es el nodo origen). En la columna *Seleccionados* registrar dicho nodo escogido para ya no volverlo a elegir. (En nuestro caso registramos esta distancia en tipo bold, negro y subrayado.)

Iteración	*a*	*b*	*c*	*d*	...	Actual	Seleccionados
0	**0**	∞	∞	∞	...	a	a

5. Registrar en la columna de cada uno de los nodos la distancia más corta que resulta de sumar la *distancia registrada en el nodo actual + distancia a los vértices adyacentes a él*, y seleccionar la distancia más corta cuyo nodo aún no esté seleccionado de esa fila de la matriz (suponer que $d_1 > d_2$), por lo tanto, la matriz será:

Iteración	*a*	*b*	*c*	*d*	...	Actual	Seleccionados
0	**0**	∞	∞	∞	...	a	a
1	**0**	d_1	d_2	∞	...		

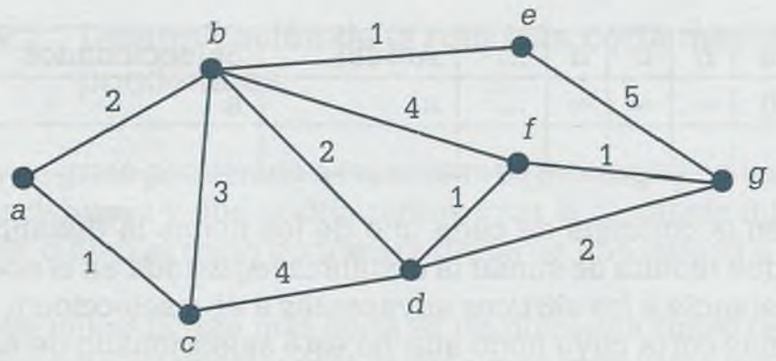
Si el nodo seleccionado tiene una distancia diferente de ∞, que es menor o igual a la que se obtiene de sumar la distancia registrada en la columna del nodo actual + la distancia de ese nodo actual a los nodos adyacentes a él, dejarla tal como está, en caso contrario cambiarla por la nueva suma.

6. Registrar en la columna *Actual* el vértice que tenga la distancia más corta de entre todos los nodos y que no haya sido seleccionado hasta ahora. Además de anotar en la columna *Seleccionados* dicho nodo para ya no volverlo a elegir.

Iteración	*a*	*b*	*c*	*d*	...	Actual	Seleccionados
0	0	∞	∞	∞	...	a	a
1	0	d_1	d_2	∞	...	c	a, c

7. Si ya están todos los vértices seleccionados, finalizar. En caso contrario regresar al paso 5.

Ejemplo 7.23. Considerar que el siguiente grafo se obtiene al observar la comunicación que existe entre las ciudades {*a*, *b*, *c*, *d*, *e*, *f*, *g*} y que la ponderación que tiene cada una de las aristas (carreteras) es la distancia que existe entre dos ciudades cualesquiera. Encontrar la ruta más corta para ir de la ciudad origen (*a*) a las ciudades restantes.



Solución

1. Seleccionar la ciudad origen (en este caso es el nodo *a*).
2. La matriz etiquetada es:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados

3. Después de colocar en la matriz la distancia que existe de la ciudad origen a ella misma y a todas las demás columnas ∞ se tiene que:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞		

4. Después de registrar en las columnas *Actual* y *Seleccionados* el vértice que tiene la distancia más corta de entre todos los nodos, la matriz queda como:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞	a	a

Hay que observar cómo la distancia del vértice elegido se distingue por el subrayado.

5. Una vez que se anotó en la columna de cada uno de los nodos (no seleccionados hasta ahora) la distancia más corta que resulta de sumar la *distancia registrada en el nodo actual + distancia a los vértices adyacentes a él* se tiene:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞		

6. Después de registrar en la columna *Actual* y agregar a la columna *Seleccionados* dicho nodo:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞	c	a, c

Repitiendo el paso 5 resulta:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞	c	a, c
2	0	2	1	5	∞	∞	∞		a, c

Repitiendo el paso 6 se obtiene:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞	c	a, c
2	0	2	1	5	∞	∞	∞	b	a, c, b

El proceso se repite hasta terminar con la siguiente matriz que muestra la distancia más corta que hay del nodo origen a todos los demás nodos del grafo:

Iteración	*a*	*b*	*c*	*d*	*e*	*f*	*g*	Actual	Seleccionados
0	0	∞	∞	∞	∞	∞	∞	a	a
1	0	2	1	∞	∞	∞	∞	c	a, c
2	0	2	1	5	∞	∞	∞	b	a, c, b
3	0	2	1	4	3	6	∞	e	a, c, b, e
4	0	2	1	4	3	5	8	d	a, c, b, e, d
5	0	2	1	4	3	5	6	f	a, c, b, e, d, f
6	0	2	1	4	3	5	6	g	a, c, b, e, d, f, g

Observar que en la iteración 3 en la columna *d* se registró una menor distancia, porque la suma del nodo actual en ese momento *b* más la distancia de ese nodo actual a *d* dan como resultado 2 + 2 = **4**, mismo que a su vez es menor que el **5** que tenía anteriormente la columna *d*, por tal razón se sustituye un valor mayor por uno menor.

Muchas de las aplicaciones de los grafos se pueden representar por medio de grafos ponderados, y por lo general siempre se desean optimizar los recursos reduciendo las distancias de tal manera que al encontrar la ruta más corta se disminuye el costo, tiempo y por supuesto distancia.

7.10 Resumen

Un grafo es un diagrama que consta de un conjunto de vértices y un conjunto de lados. Los nodos o vértices se indican por medio de un pequeño círculo y se les asigna un número o letra. Los lados o aristas son las líneas que unen un vértice con otro y se les asigna una letra, un número o una combinación de ambos. Cuando dos aristas unen a un mismo par de vértices, se les llama lados paralelos. Un lazo es aquella arista que sale de un vértice y regresa a él mismo. La valencia de un vértice es el número de aristas que salen o entran a un vértice.

Los tipos de grafos más comunes son:

- ***Grafo simple.*** Es aquel que no tiene lazos ni lados paralelos.

- ***Grafo completo de n vértices (K_n).*** Es aquel grafo en donde cada vértice está relacionado con todos los demás, sin lazos ni lados paralelos.

- ***Complemento de un grafo (G').*** Es aquél grafo que le falta al grafo G, para entre ambos formar un grafo completo de n vértices. Dicho grafo no tiene lazos ni lados paralelos.

- ***Grafo bipartido.*** Es aquel que está compuesto por dos conjuntos de vértices *A* y *B* en donde los vértices del conjunto *A* se relacionan con los del *B*, pero entre los vértices de un mismo conjunto no existe arista que los una.

- ***Grafo bipartido completo ($K_{n,m}$).*** Es aquel grafo que está compuesto por dos conjuntos de vértices, *A* y *B*, en donde cada vértice del conjunto A está unido con todos los vértices de *B*, pero entre los vértices de un mismo conjunto no existe arista que los una.

- ***Grafo conexo.*** Es aquél en el que para cualquier par de vértices w, x, distintos entre sí existe un camino para ir de w a x.

- ***Grafos isomorfos.*** Se dice que dos grafos G_1 y G_2 son isomorfos, cuando teniendo apariencia diferente realmente son iguales, porque tienen mismo número de lados, mismo número de vértices, mismo conjunto de valencias, ambos son o no conexos, ambos tienen el mismo número de circuitos de longitud n y ambos tienen o no circuito de Euler.

- ***Grafos de similaridad.*** Son aquellos que permiten agrupar información con características semejantes. Este tipo de grafos es útil en el reconocimiento de patrones, en donde se agrupa información con propiedades muy parecidas.

- ***Grafo plano.*** Es aquel que se puede dibujar en un solo plano y cuyas aristas no se cruzan entre sí. Euler estableció que la ecuación A = L – V + 2 se cumple para los grafos planos. Los grafos K_5 y $K_{3,3}$ son grafos no planos importantes y se utilizan como patrones para demostrar que otros más complejos no son planos.

- ***Grafos ponderados.*** Son aquellos en donde a las aristas se les asigna un valor al cual se le llama ponderación y que podría representar la distancia que hay de un nodo a otro, o bien el costo de transportarse de una ciudad a otra. Un problema típico de la teoría de grafos consiste en encontrar el camino más corto para ir de una ciudad origen (w) a una ciudad destino (x). Pueden existir distintas rutas para ir de un nodo a otro, pero el objetivo es encontrar el más corto o bien el más económico si es que la ponderación representa un costo. El método más utilizado para encontrar la ruta más corta es por medio del algoritmo de Dijkstra.

Un grafo se puede representar por medio de una *matriz de adyacencia* M_a (la cual es una matriz cuadrada en donde los vértices del grafo se indican como filas, pero también como columnas de la matriz) o bien por medio de una *matriz de incidencia* M_i (en la que los vértices del grafo se colocan como filas y las aristas como columnas).

La información de un grafo se puede recorrer de diferentes maneras y de acuerdo con sus características propias estos recorridos reciben respectivamente alguno de los siguientes nombres:

- ***Camino.*** Es una sucesión de lados que van de un vértice x a un vértice w.

- ***Circuito (Ciclo).*** Es un camino que regresa al mismo vértice de donde salió.

- ***Circuito simple de longitud n.*** Es aquel camino del vértice w al vértice w que solamente tiene un ciclo en la ruta que sigue.

- ***Camino simple de longitud n.*** Es una sucesión de lados que van de un vértice x a un vértice w, en donde los lados que componen dicho camino son distintos e iguales a n.

Algunos recorridos importantes en los grafos son los siguientes:

- ***Camino de Euler.*** Es aquel camino que recorre todos los vértices pasando por todas las ramas solamente una vez. Un camino de Euler deberá iniciar y terminar en vértices de valencia impar.

- ***Circuito de Euler.*** Es aquel ciclo que recorre todos los vértices pasando por todos los lados solamente una vez. Un grafo tiene circuito de Euler solamente si es conexo y todos su vértices tienen valencia par.

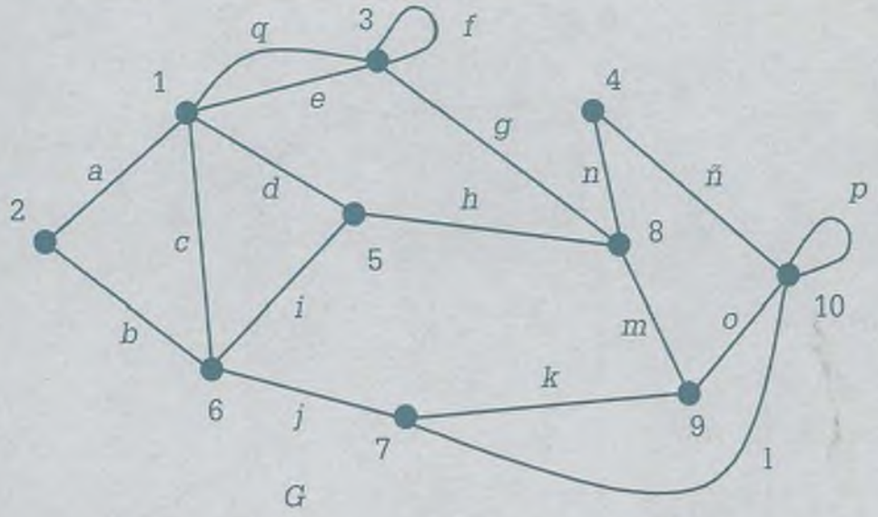
- ***Circuito de Hamilton.*** Es aquel circuito que pasa por cada vértice solamente una vez.

El número cromático de un grafo X(G) es el mínimo de colores necesarios para colorear dicho grafo de tal forma que vértices adyacentes no estén iluminados del mismo color. Todo grafo plano puede ser coloreado con máximo cuatro colores, de acuerdo con el teorema de Appel y Haken. Al número de formas en que se puede colorear el grafo G usando para ello w colores se llama polinomio cromático P(G,w).

El uso fundamental de los grafos son las redes carreteras, telefónicas, eléctricas, de agua potable, de alcantarillado, de computadoras, de cartografía y de distribución de tareas entre otras, y lo que siempre se busca es optimizar los recursos de dichas redes, reducir costos, disminuir distancias o aumentar la velocidad de comunicación. Por otro lado, los grafos permiten ilustrar estructuras químicas, organigramas de una empresa o algoritmos en el área de la computación.

7.11 Problemas

7.1 Considérese el siguiente grafo G:



G

a) ¿Es grafo simple?

b) ¿Es grafo K_n?

c) ¿Es grafo $K_{n,m}$?

d) ¿Es grafo conexo?

e) ¿Es grafo plano? ¿En caso de ser plano, se cumple la ecuación de Euler $A = L - V + 2$?

f) ¿Tiene un camino de Euler?

g) ¿Tiene un circuito de Euler?

h) ¿Tiene circuito de Hamilton?

i) Obtener:

- El conjunto de vértices (V).
- Conjunto de aristas (A).
- Conjunto de lazos (L).
- Conjunto de lados paralelos (P).

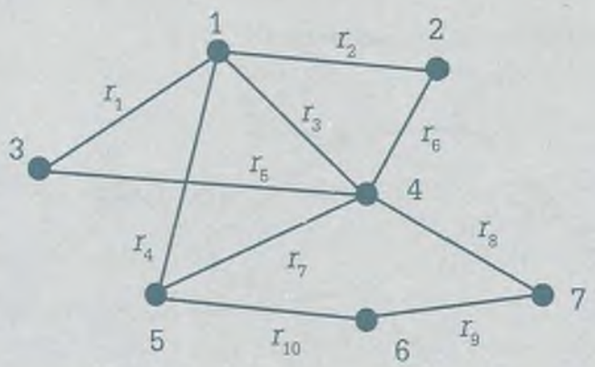
j) Obtener la matriz de adyacencia y la de incidencia.

k) ¿Cuál es la valencia de cada uno de los vértices?

l) Decir si los siguientes recorridos son caminos, camino simple, circuito, circuito simple:

- (1, 5, 8, 9, 7, 6, 1, 3).
- (5, 8, 4, 10, 10, 7, 6, 5).
- (1, 3, 3, 1).
- (4, 10, 9, 7, 6, 5, 1, 3, 8, 4).
- (2, 6, 5, 8, 9, 10).

7.2 Considérese el siguiente grafo G:



G

a) Es un grafo

- ¿Simple?
- ¿Conexo?

b) Obtener las matrices de adyacencia y de incidencia del grafo G.

c) ¿Tiene camino de Euler? Si es así, cuál es.

d) ¿Tiene circuito de Euler? Si es así, cuál es.

e) ¿Tiene circuito de Hamilton? Si es así, cuál es.

f) ¿Es plano? En caso de ser plano verificar que satisface la ecuación de Euler $A = L - V + 2$ y comprobar el resultado por medio de sus propiedades.

g) ¿Es un grafo K_n?

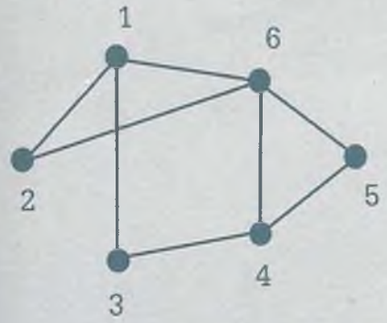
h) ¿Es un grafo $k_{n,m}$?

i) Obtener el complemento del grafo (G').

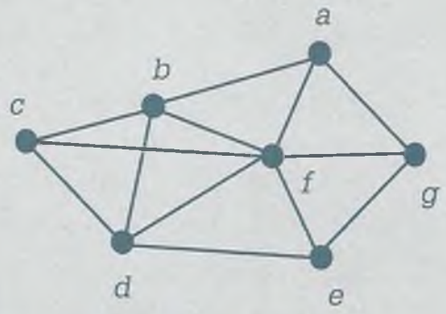
j) ¿Qué valencia debe tener cada uno de los vértices del grafo G para que se considere un grafo K_n, y cuál es el número total de aristas que debería tener?

7.3 En cada uno de los incisos obtener el complemento del grafo en caso de tratarse de un grafo simple, y en caso contrario explicar por qué no es grafo simple.

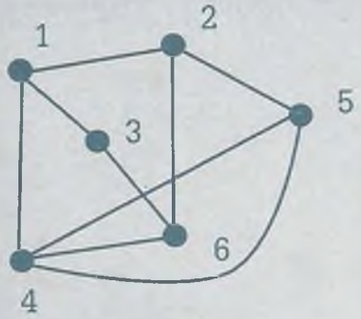
a)



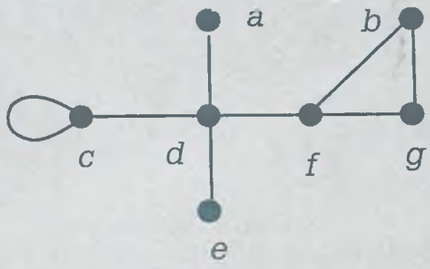
b)



c)

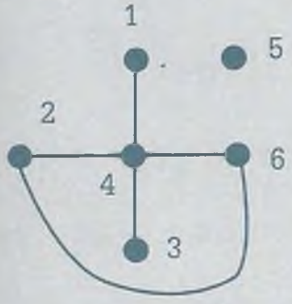


d)



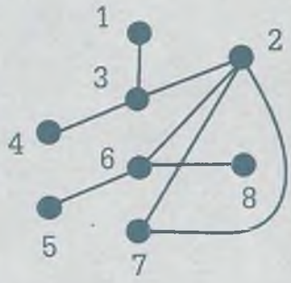
7.4 En cada uno de los siguientes incisos obtener el complemento del grafo en caso de ser un grafo simple, en caso contrario explicar por qué no es grafo simple.

a)



b)

c)



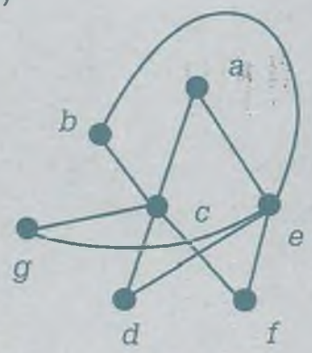
7.5 En relación con cada uno de los siguientes incisos determinar si el grafo es:

- Bipartido.
- Bipartido completo ($K_{n,m}$).
- Completo de n vértices (K_n).
- Ninguno de los anteriores.

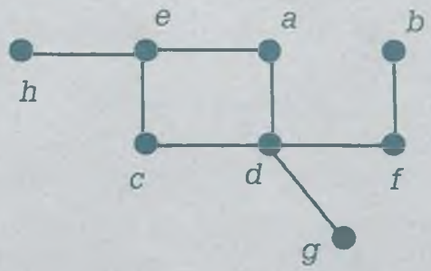
En caso de ser bipartido o bipartido completo ¿cuáles son los elementos de los conjuntos? En caso de ser completo de n vértices ¿cuáles son los elementos del conjunto y cuál es el valor de n?

a)

b)



c)

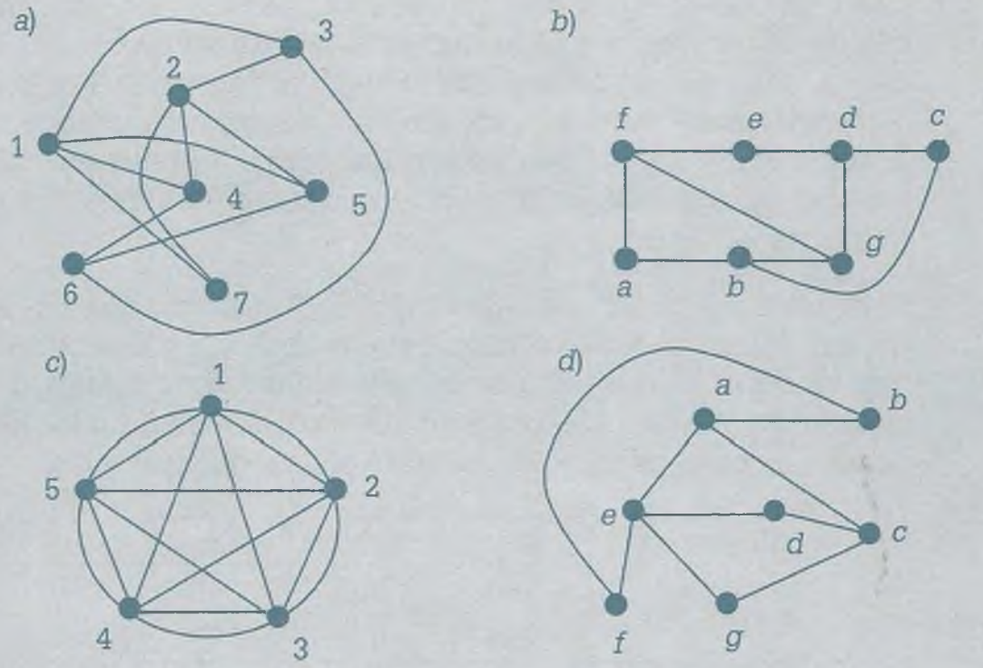


d)

7.6 Determinar si el grafo de cada uno de los siguientes incisos es:

- Bipartido.
- Bipartido completo ($K_{n,m}$).
- Completo de n vértices (K_n).
- Ninguno de los anteriores.

En caso de ser bipartido o bipartido completo ¿cuáles son los elementos de los conjuntos, y en caso de ser completo de n vértices cuáles son los elementos del conjunto y cuál es el valor de n?



7.7 Se desea enviar información a los usuarios de internet. Esto implica que si un usuario muestra interés en la información deportiva y cultural entonces sólo se le enviarán artículos de tipo cultural y deportivo, si muestra que sólo le interesa información científica entonces únicamente esa información se le enviará.

La información en internet está clasificada en artículos: culturales, deportivos y de carácter científico. Se realiza un registro mensual de las preferencias de los usuarios obteniéndose que en el mes anterior las preferencias fueron como se muestra en la siguiente tabla:

Usuario (U)	Culturales (C)	Deportivos (D)	Científicos (Ci)
1	10	2	0
2	2	14	1
3	0	15	2
4	3	9	7
5	8	1	1
6	1	12	4

a) Obtener los grafos de similaridad para un coeficiente de inferencia C = 6. ¿Qué interpretación se le puede dar a estos grafos?

b) ¿Cómo quedarían los grafos si C = 5? ¿Cuál es la interpretación de estos grafos?

7.8 Las computadoras trabajan con tres colores básicos a partir de los cuales construyen todos los demás mediante un proceso de mezcla por unidades de pantalla denominadas "pixels", estos colores son rojo, azul y verde. El sistema así definido se conoce como sistema RGB (Red, Green, Blue). Cada pixel tiene reservado un espacio en la memoria de la computadora, para almacenar la información.

Se desea determinar la similaridad del iris del ojo de una persona (1), con el iris del ojo de otras personas (2, 3, 4, 5 y 6) en una base de datos. Los bloques de pixels en estudio son de las mismas dimensiones, 30×30, y la forma en que se distribuye la información de las personas es como se muestra en la siguiente tabla:

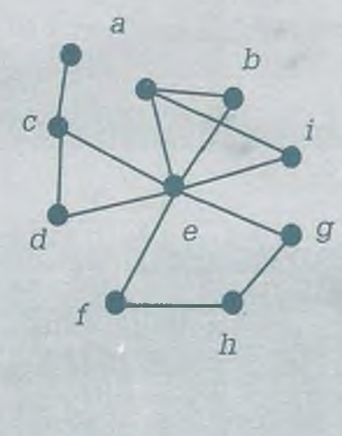
Persona (P)	Rojo	Verde	Azul
1	280	305	315
2	295	312	293
3	367	280	256
4	268	309	323
5	400	160	340

a) Obtener los grafos de similaridad para un coeficiente de inferencia C = 50. ¿Cuál es la interpretación de estos grafos?

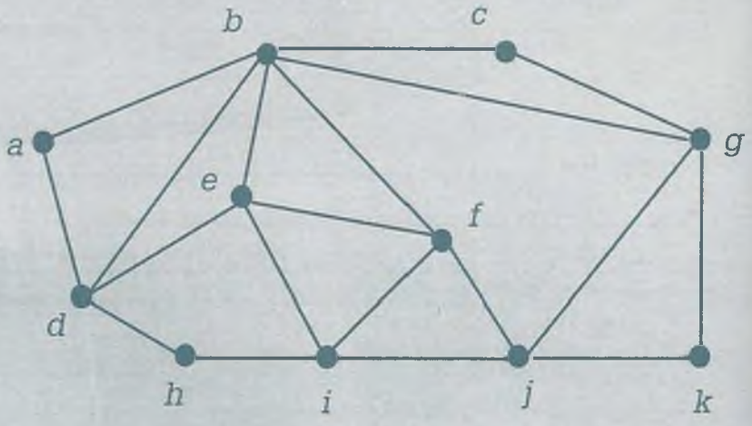
b) ¿Cómo quedarían los grafos si C = 30? Hacer una interpretación.

7.9 Establecer si cada uno de los siguientes grafos tiene camino de Euler y/o circuito de Euler.

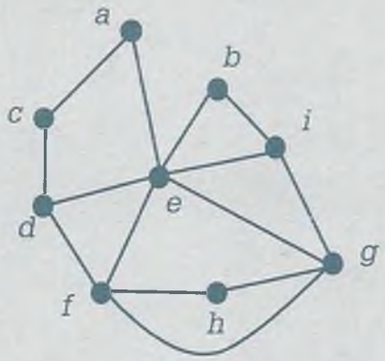
En caso afirmativo determinar dicho camino o circuito de Euler usando el algoritmo de Fleury. En caso negativo explicar el porqué.



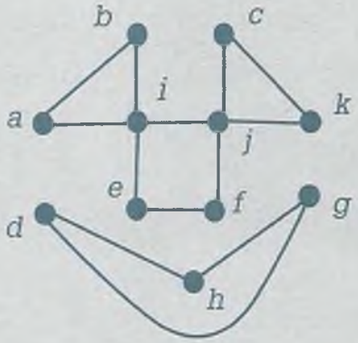
a)



b)



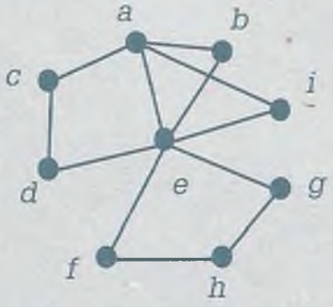
c)



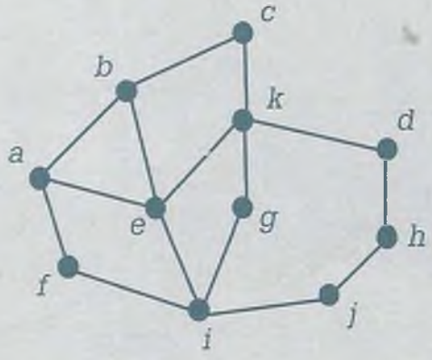
d)

7.10 Establecer si cada uno de los grafos siguientes tiene camino de Euler y/o circuito de Euler.

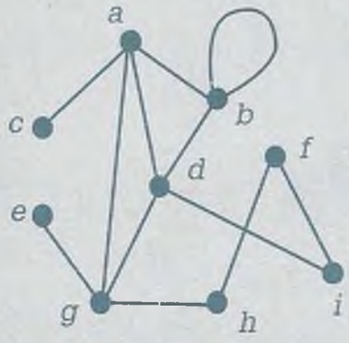
En caso afirmativo determinar dicho camino o circuito de Euler usando el algoritmo de Fleury. En caso negativo explicar el porqué.



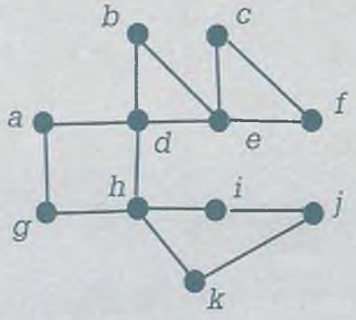
a)



b)

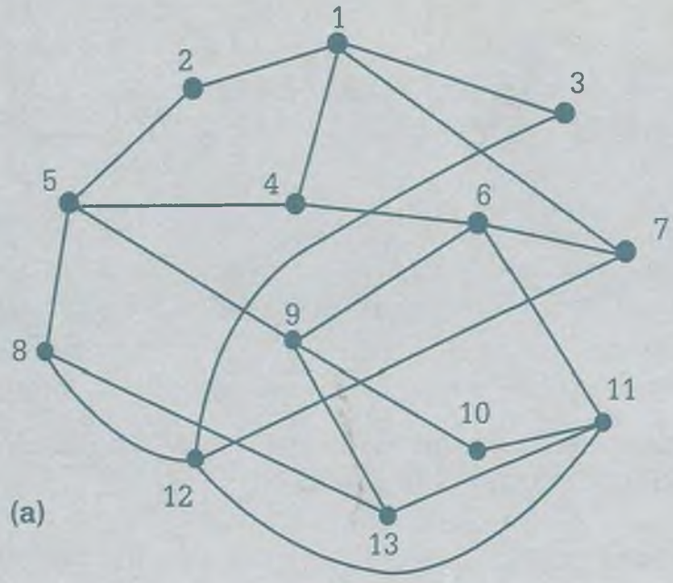


c)

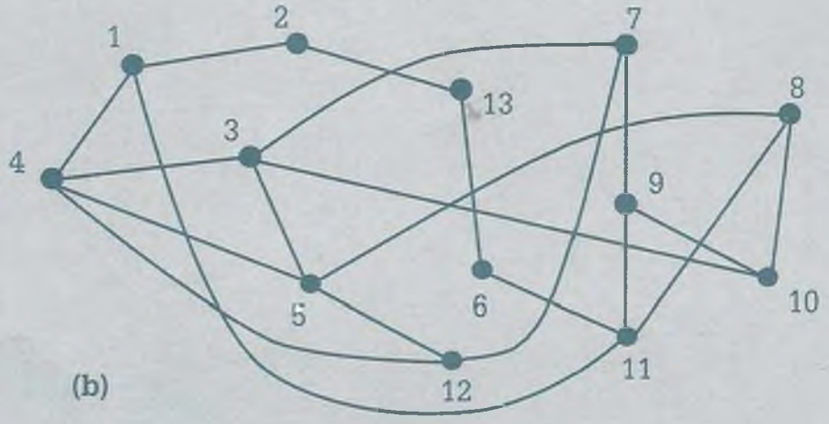


b)

7.11 Determinar si los siguientes grafos tienen circuito de Hamilton:



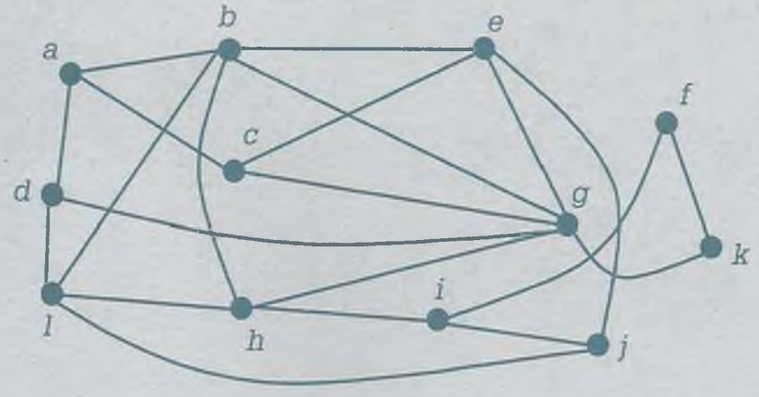
(a)



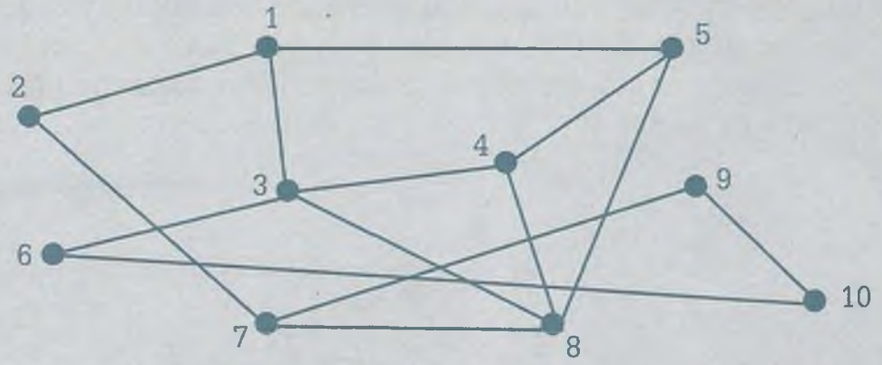
(b)

7.12 Determinar el circuito de Hamilton en cada uno de los siguientes grafos.

a)



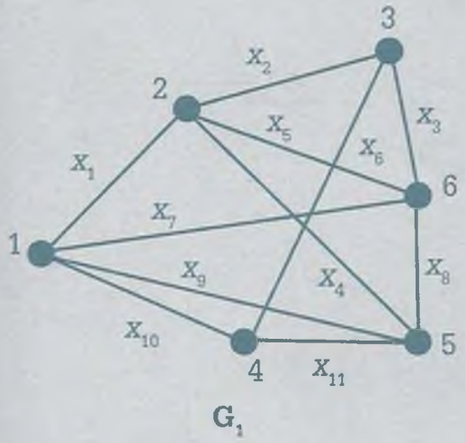
b)



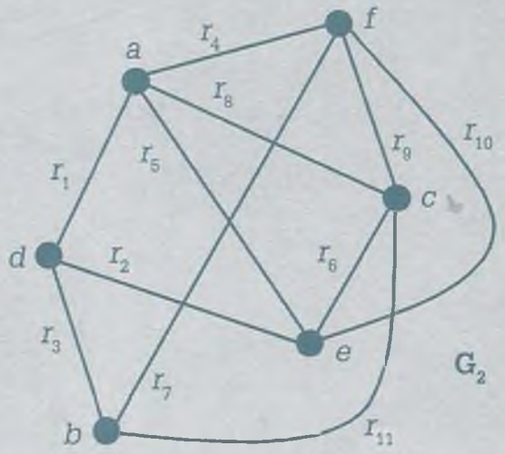
7.13 Determinar si los grafos siguientes G_1 y G_2 son isomorfos.

a) Por medio de las propiedades de G_1 y G_2.

b) Por medio de las matrices de incidencia.



G_1

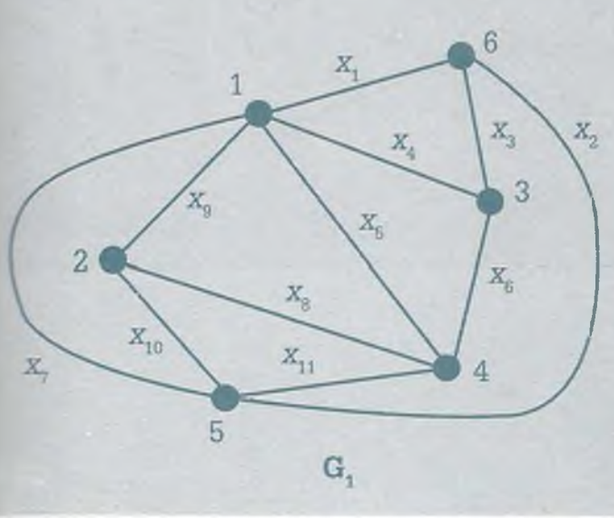


G_2

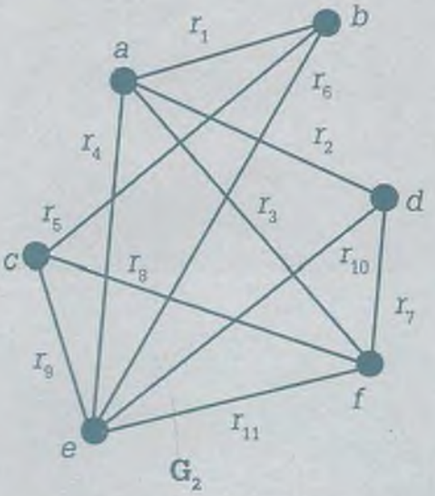
7.14 Determinar si los grafos siguientes G_1 y G_2 son isomorfos.

a) Por medio de las propiedades de G_1 y G_2.

b) Por medio de las matrices de incidencia.



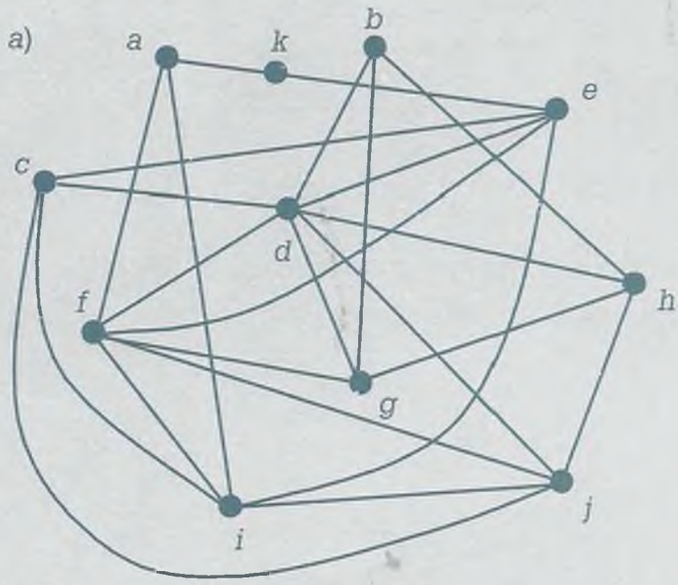
G_1



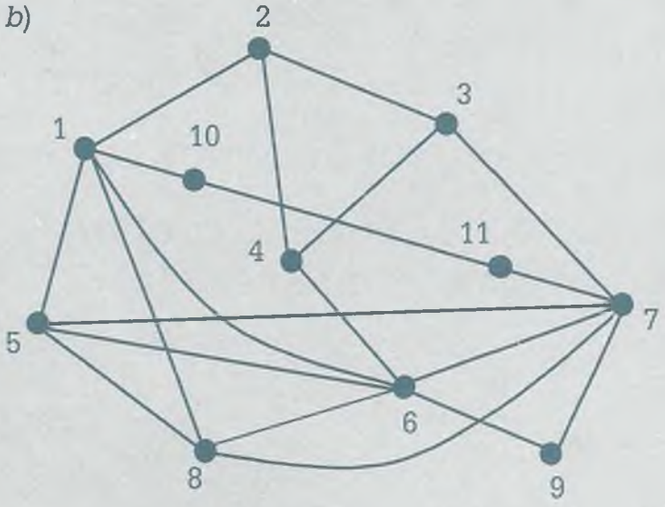
G_2

7.15 En cada uno de los siguientes incisos, dibujar el grafo en forma plana (en caso de ser plano probar que satisface la ecuación de Euler) o bien demostrar que no es plano obteniendo en él un grafo $K_{3,3}$ o K_5.

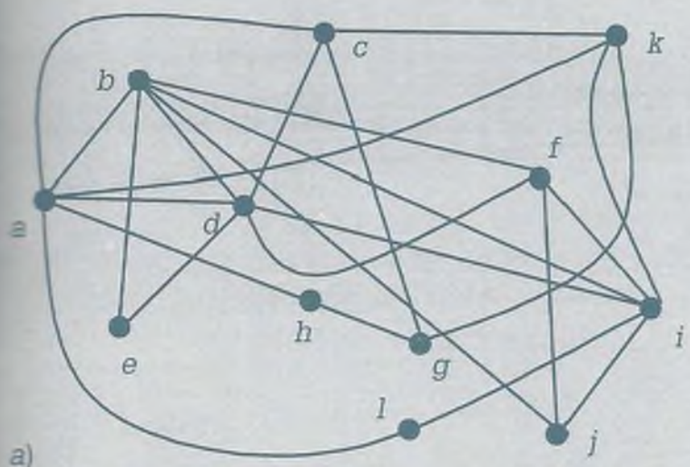
a)



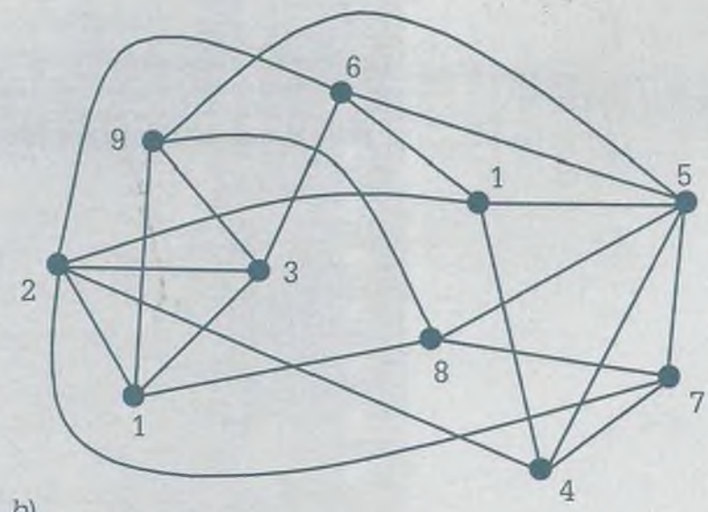
b)



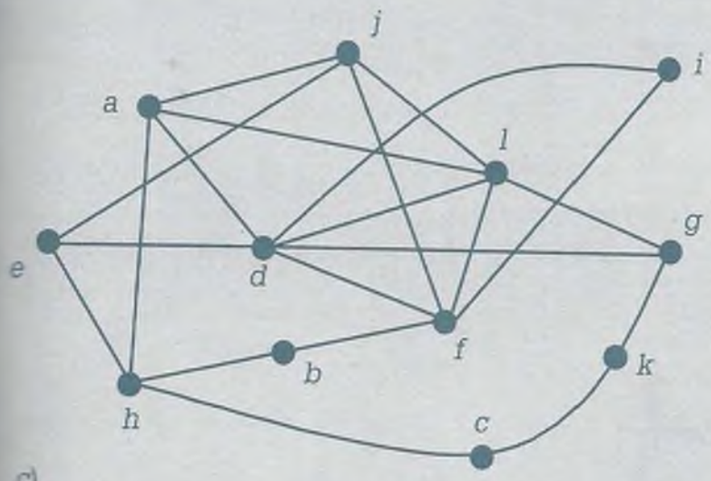
7.16 En cada uno de los siguientes incisos, dibujar el grafo en forma plana (en caso de ser plano mostrar que satisface la ecuación de Euler) o bien demostrar que no es plano obteniendo en él un grafo $K_{3,3}$ o K_5.



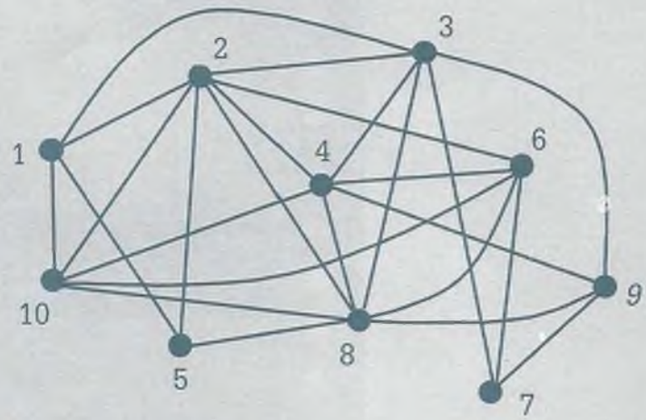
a)



b)



c)



d)

7.17 Dibujar un grafo plano conexo de 10 vértices y valencias 2, 2, 2, 3, 3, 4, 4, 4, 4, 6.

a) ¿Cuántos lados y cuántas caras tiene dicho grafo?

b) Probar que se cumple la ecuación de Euler.

7.18 Dibujar un grafo plano conexo de acuerdo a como se indica en cada inciso:

a) 9 vértices con valencias: 2, 3, 3, 4, 4, 4, 5, 5, 6.

b) 12 vértices con valencias: 2, 3, 3, 4, 4, 4, 4, 4, 5, 5, 6, 6.

Para cada uno de los incisos:

- Cuántos lados y cuántas caras tiene el grafo.
- Probar que se cumple la ecuación de Euler.

7.19 Colocar una E en la celda correspondiente de la siguiente tabla cuando el grafo $K_{n,m}$ tenga un circuito de Euler y colocar una H en la celda cuando el grafo $K_{n,m}$ tenga un circuito de Hamilton.

	1	2	3	4	5	6	...	m
1								
2								
3								
4								
5								
6								
.								
.								
n								

a) ¿Para qué valores de n y m un grafo $K_{n,m}$ tiene únicamente circuito de Euler?

b) ¿Para qué valores de n y m un grafo $K_{n,m}$ tiene únicamente un circuito de Hamilton?

c) ¿Para qué valores de n y m un grafo $K_{n,m}$ tiene un circuito de Euler y además un circuito de Hamilton?

7.20 Colocar en la siguiente tabla un asterisco en la fila E si el grafo K_n tiene un circuito de Euler, y colocar un asterisco en la fila H si el grafo K_n tiene un circuito de Hamilton.

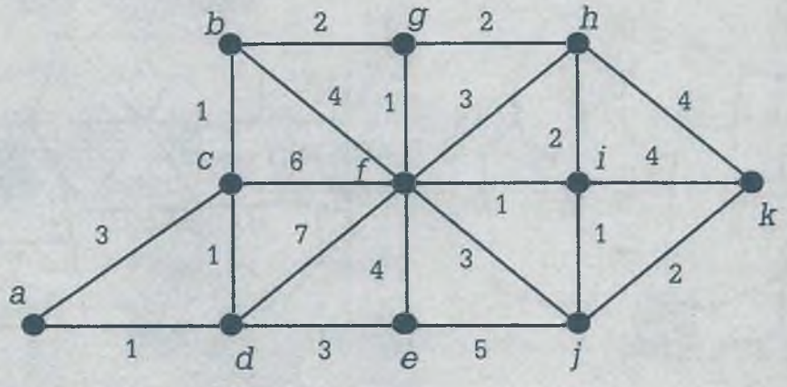
	1	2	3	4	5	6	7	...	n
E									
H									

a) ¿Para qué valores de n el grafo completo K_n no tiene circuito de Hamilton?

b) ¿Para qué valores de n el grafo K_n tiene circuito de Hamilton exclusivamente?

c) ¿Para qué valores de n el grafo K_n tiene un circuito de Euler y a la vez un circuito de Hamilton?

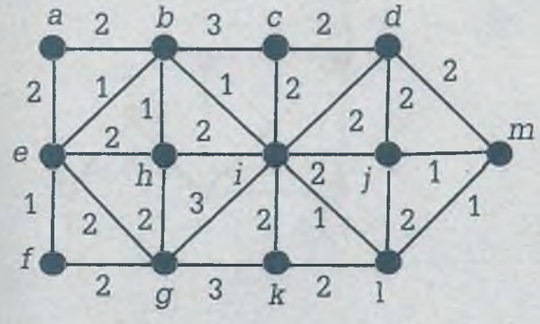
7.21 Determinar la ruta más corta en el siguiente grafo ponderado, para ir del nodo **a** hasta los nodos restantes del grafo, usando el algoritmo de Dijkstra.



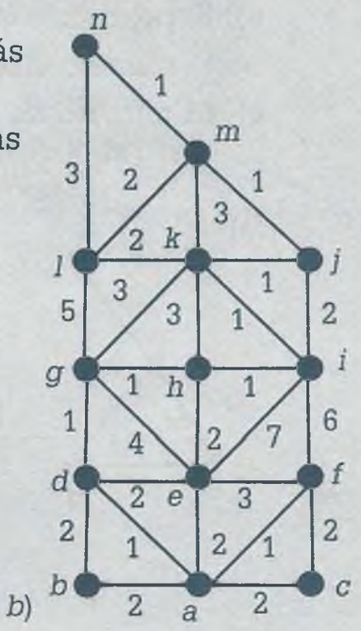
7.22 Determinar la ruta más corta en el siguiente grafo, usando el algoritmo de Dijkstra.

a) Para ir del nodo g a todos los demás en el inciso **a**.

b) Para ir del nodo c a todos los demás en el inciso **b**.

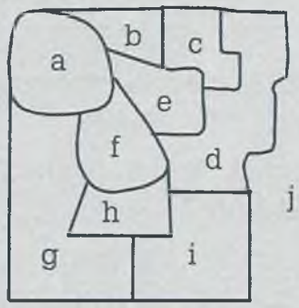


a)



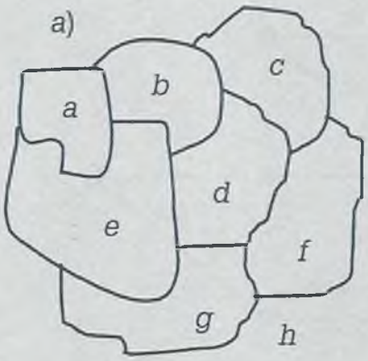
b)

7.23 Representar por medio de su grafo plano el siguiente mapa, colorear dicho grafo y obtener el número cromático X(G):

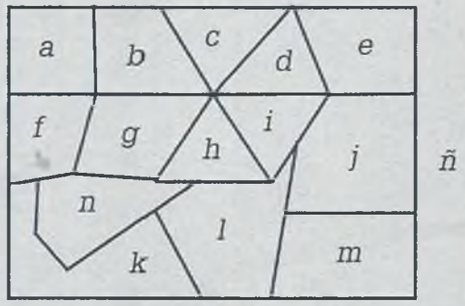


7.24 Representar por medio de grafos planos los mapas de cada uno de los incisos, colorear dichos grafos y obtener el número cromático X(G):

a)



b)



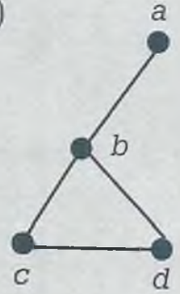
7.25 Para cada uno de los siguientes grafos obtener:

a) El polinomio cromático P(G,w).

b) El número cromático X(G).

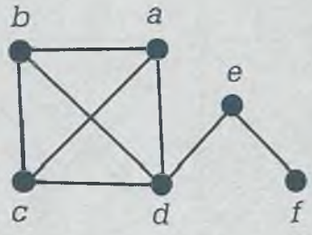
c) El número de maneras distintas que se puede coloreas cada grafo si w=6.

a)



G

b)



G

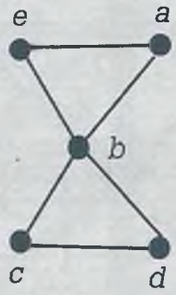
7.26 Para cada uno de los siguientes grafos obtener:

a) El polinomio cromático P(G,w).

b) El número cromático X(G).

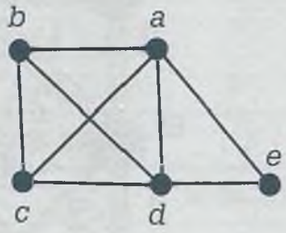
c) El número de maneras distintas en que se puede colorear cada grafo si w=6.

a)



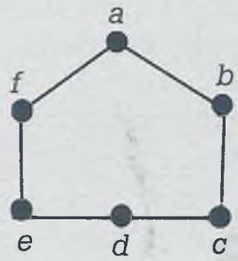
G

b)



G

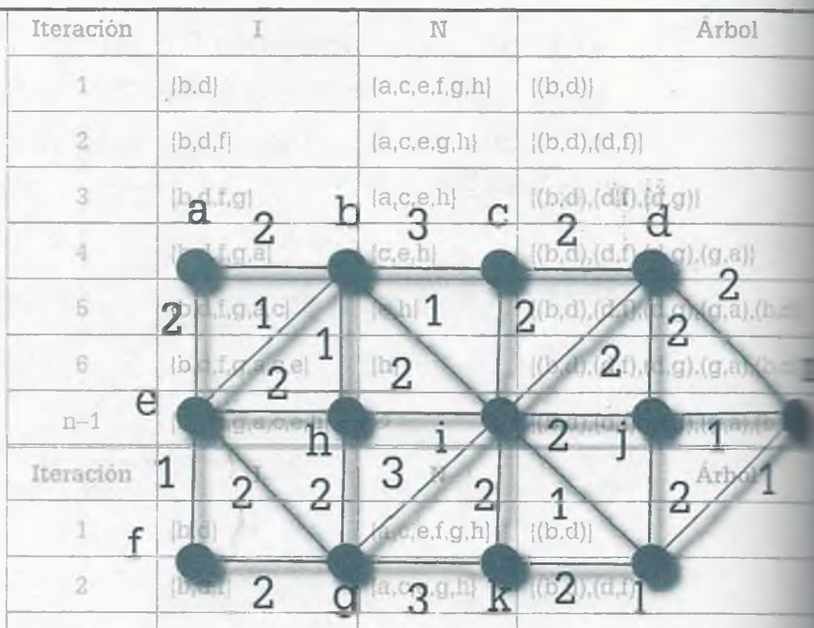
c)

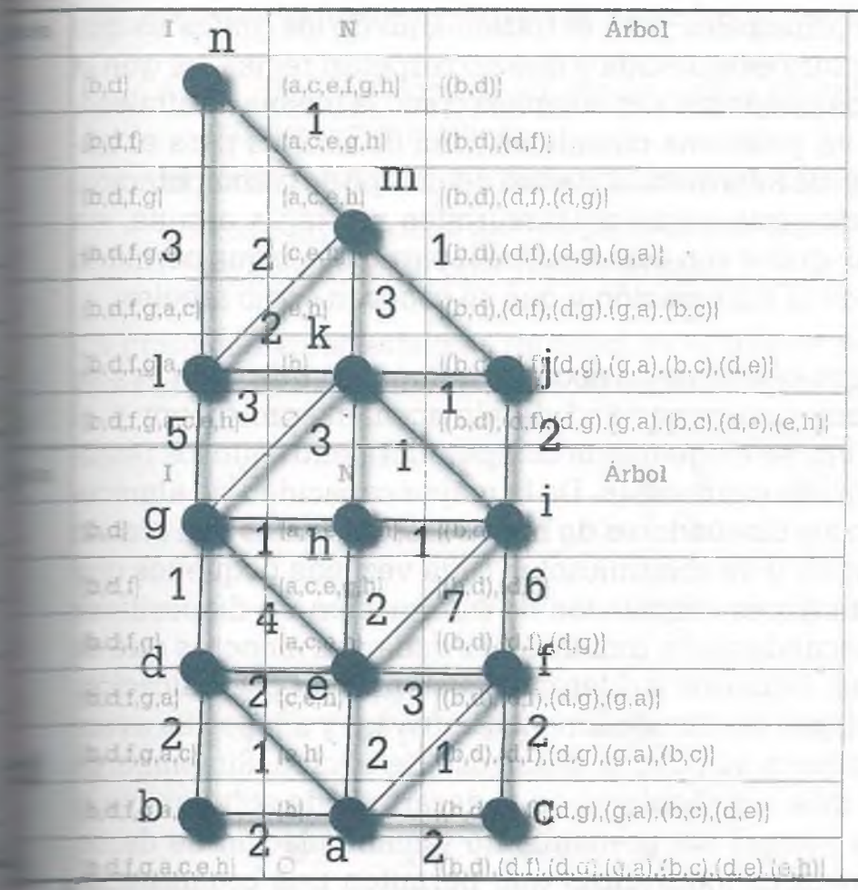


G

CAPÍTULO VIII

Árboles





Con los árboles se representa la dependencia lógica entre la deci los atributos

Nicolas Bourbaki

Objetivos

- Establecer los principios fundamentales para abordar el tema de árboles.
- Reconocer los diferentes tipos de árboles, sus propiedades y aplicaciones.
- Conocer las características que debe tener un grafo para ser considerado como árbol y obtener un árbol a partir de un grafo.
- Aprovechar las propiedades de los árboles para modelar y resolver problemas específicos.
- Aplicar la estructura de árbol en la organización y procesamiento de información.

8.1 Introducción

Uno de los problemas principales para el tratamiento de los grafos es que no guardan una estructura establecida y que no respetan reglas, ya que la relación entre los nodos puede ser tan compleja como la misma naturaleza. Sin embargo, este es un problema cuando se trata de usarlos para el tratamiento y organización de información, dentro del campo de la computación. En lugar de usar grafos que están estructurados sin regla alguna, en computación se utilizan grafos con características particulares que permiten un mejor tratamiento de la información y que se conocen como árboles.

En computación hay dos objetivos básicos: el primero es que cada vez se desarrollen equipos con una capacidad de almacenamiento mayor y el segundo es que cada vez se exige que la computadora entregue los resultados en forma más rápida y ordenada. De la mayor capacidad de almacenamiento se encargan los diseñadores de hardware, los cuales han creado equipos computacionales y de comunicación cada vez más pequeños que permiten almacenar mayores cantidades de información en dispositivos de almacenamiento secundario de dimensiones cada vez menores que se usan en computadoras, celulares y diferentes sistemas de comunicación. De lo segundo se encargan los diseñadores de software y aquí se ha avanzado muy poco en comparación con el desarrollo de equipo. Sin embargo uno de los progresos más significativos en el desarrollo de software es la utilización de árboles para el almacenamiento y manipulación de datos. Los árboles son estructuras jerárquicas que permiten una organización ordenada de la información, de forma que cuando se requiera se pueda encontrar en forma rápida y precisa.

Una de las primeras aplicaciones de los árboles se presentó en 1847 cuando Gustav Kirchhoff los utilizó en la manipulación de redes eléctricas; otra aplicación importante la llevó a cabo Grace Harper en 1951 al utilizarlos en el manejo de expresiones matemáticas. En la actualidad los árboles se usan en la computación en los procesos de clasificación de información, bases de datos, codificación de información, estructuras de datos y reconocimiento de patrones.

Tomando en cuenta la teoría de grafos expuesta en el capítulo anterior, el concepto de árbol se puede definir en los siguientes términos:

Definición. Un árbol es un grafo conexo que no tiene ciclos, lazos ni lados paralelos.

8.2 Propiedades de los árboles

Las propiedades básicas de un árbol son las siguientes:

a) Es un grafo conexo en donde existe un camino entre cualquier par de vértices (w, x).

b) Este grafo no tiene ciclos ni lados paralelos.

c) Todo árbol con al menos dos vértices tiene al menos una hoja (si se considera al otro vértice la raíz).

Un grafo con características de árbol es el que se parece a un árbol real con sus ramas hacia abajo, como se muestra en la figura 8.1.

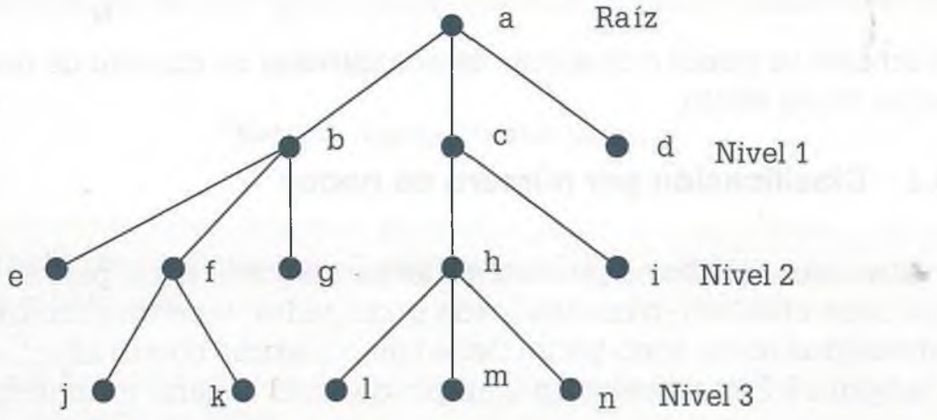


Figura 8.1 Ejemplo de árbol.

Los vértices de un árbol reciben el nombre de **nodos** y los lados de **ramas**. Un grafo está compuesto por niveles y el más alto de la jerarquía se llama raíz. La raíz tiene un nivel 0, los vértices inmediatamente debajo de la raíz tienen un nivel 1 y así sucesivamente. La altura o peso de un árbol es el valor de su nivel más bajo, por ejemplo en el árbol de la figura 8.1 la altura es 3.

Con excepción de la raíz, todo nodo está vinculado a otro de mayor nivel que recibe el nombre de **padre**, también cualquier nodo puede tener uno o más elementos relacionados en un nivel más bajo y a éstos se les llama **hijos**. En el árbol de la figura 8.1 la raíz es a y los hijos de la raíz son {b, c, d}. Como se ve, ningún hijo puede tener más de un padre.

A los elementos que están en las puntas de las ramas (es decir, que no tienen hijos) se les llama **hojas**. En el grafo de la figura 8.1 las hojas son {d, e, g, i, j, k, l, m, n}.

A todos los elementos colocados debajo de un nodo, independientemente de su nivel, se les llama **descendientes**. En el grafo de la figura 8.1 el nodo c tiene como descendientes a los nodos {h, i, l, m, n} y entre ellos se puede formar el subárbol {c, h, i, l, m, n}. En casos como éste se dice que un árbol puede estar integrado por varios subárboles.

Los elementos colocados en una misma línea de descendencia, antes de un nodo, se llaman **antecesores**. En la figura 8.1 los antecesores de k son {f, b, a}.

Por otro lado, se llaman **vértices internos** a todos aquellos que no son hojas. En la figura 8.1 {a, b, c, f, h} son vértices internos.

8.3 Tipos de árboles

Los árboles se pueden clasificar de acuerdo con su número de nodos y en función de su altura.

8.3.1 Clasificación por número de nodos

En este caso los árboles pueden ser **binarios** (cada nodo padre tiene uno o dos hijos máximo), **trinarios** (cada nodo padre tiene máximo tres hijos), **cuaternarios** (cada nodo padre tiene como máximo cuatro hijos), etcétera. En la figura 8.2 se muestra un ejemplo de árbol trinario y cuaternario.

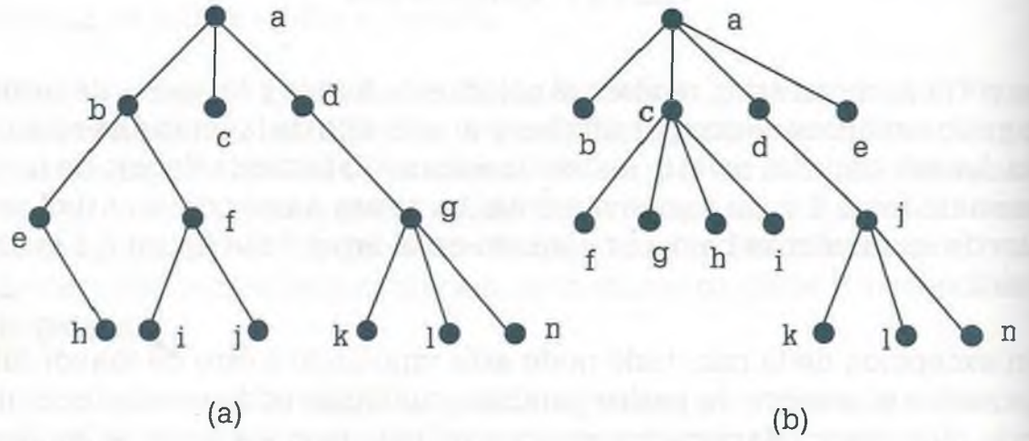


Figura 8.2 Dos tipos de árbol: (**a**) trinario; (**b**) cuaternario.

- **Árbol binario**. En este tipo de árbol cada nodo tiene como máximo dos hijos, esto es, el nodo puede tener dos ramas, una o ninguna, pero nunca puede tener más de dos. En la figura 8.3 se muestra un ejemplo de árbol binario.

Los árboles binarios son especialmente importantes en el área de la computación ya que por su naturaleza de tener solamente dos valores (0, 1), o bien falso o verdadero, son muy útiles en aplicaciones de sistemas digitales.

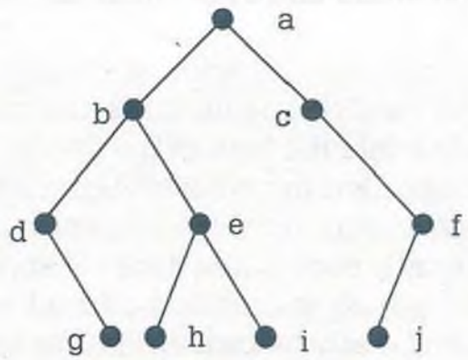


Figura 8.3 Ejemplo de árbol binario.

- **Árbol binario completo**. Es aquél en el que cada nodo tiene dos ramas o ninguna.

Un árbol binario completo con i nodos internos tiene (i + 1) hojas y (2i + 1) vértices en total.

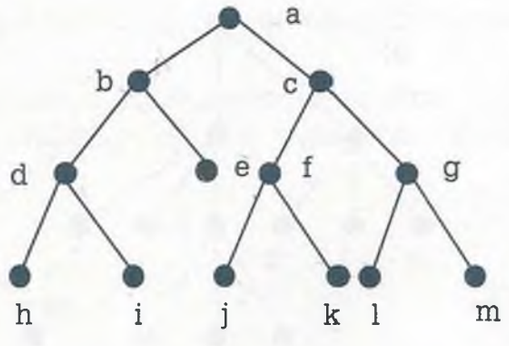


Figura 8.4 Ejemplo de árbol binario completo.

En el caso del árbol de la figura 8.4 los nodos internos son i = 6, por lo tanto tiene:

Hojas = i + 1 = 6 + 1 = 7

Total de vértices = 2i + 1 = 2 (6) + 1 = 13

Finalmente hay que destacar que los árboles completos trinarios, cuaternarios o con más hijos se usan para organizar información voluminosa.

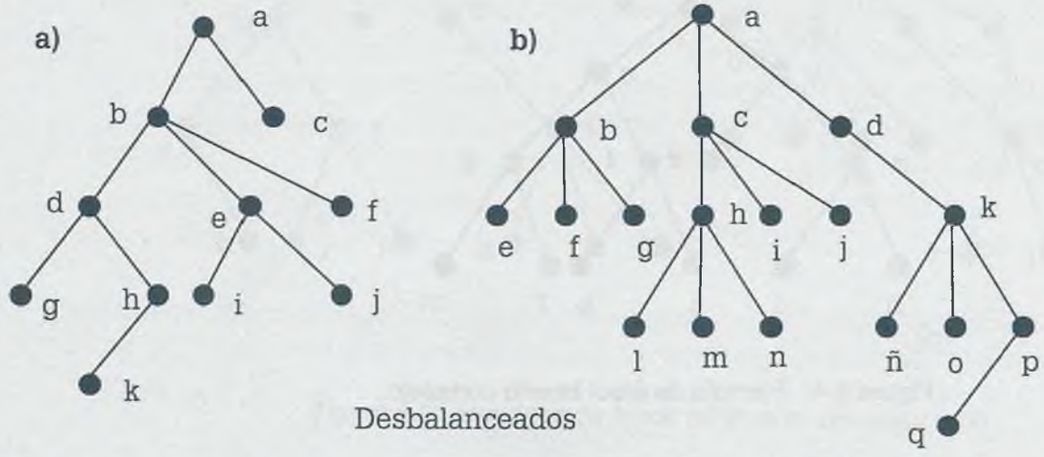
8.3.2 Clasificación por altura

De acuerdo con este criterio los árboles pueden ser **balanceados** (cuando la diferencia de altura entre sus ramas es máximo 1) y **desbalanceados** (cuando la diferencia de altura entre las ramas es mayor de 1).

- **Árbol balanceado**. Se dice que un árbol con una altura h está balanceado si el nivel de cualquier hoja es h o (h – 1), esto es, si hay una diferencia máxima de un nivel entre hojas. Algunos autores consideran que un árbol está balanceado cuando la diferencia máxima entre hojas es de 1, pero además cada nodo padre debe tener el mismo número de hijos, a excepción del que no se complete colocado en la parte baja del árbol. Es por esa razón que al balancear un árbol se debe indicar también el número de hijos que tendrá cada uno de los nodos.

  Para balancear un árbol con una cantidad constante de hijos de los nodos padres, se llenan empezando por la raíz y descendiendo con un avance de izquierda a derecha.

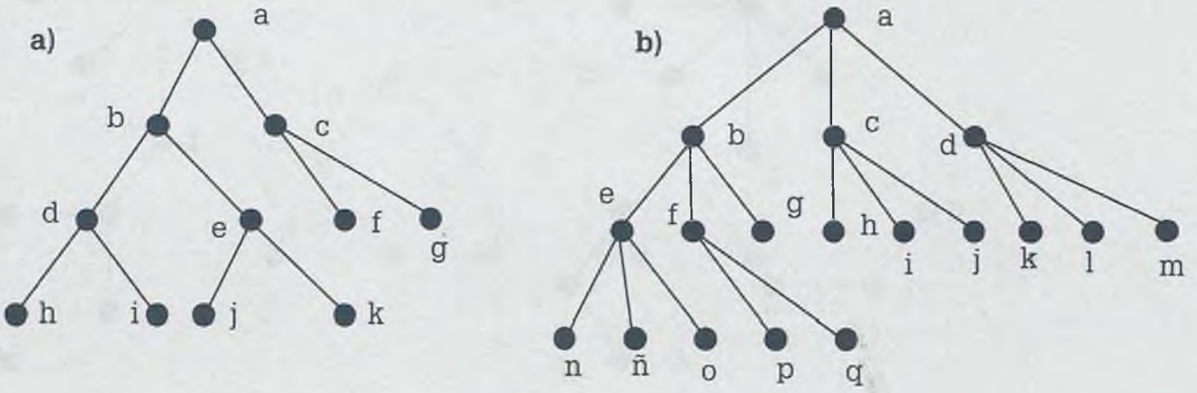
Ejemplo 8.1. Balancear como binario el árbol del inciso (a) y como trinario el del inciso (b).



Desbalanceados

Solución. Estos árboles son desbalanceados porque la diferencia de alturas entre hojas sobrepasa (h – 1). Por ejemplo, en el árbol del inciso (a) el nivel del vértice (c) es 1 mientras que la altura del árbol es h = 4 y ob-

viamente $1 \neq h - 1$. En la siguiente figura se tienen los mismos árboles balanceados por lo que la diferencia máxima entre hojas es de 1.



Balanceados

8.4 Bosques

Un **bosque** es un conjunto de árboles, en otras palabras un árbol es un bosque conectado.

De un árbol se pueden obtener varios subárboles, mismos que conforman un bosque. A su vez un árbol puede considerarse como un bosque conectado, sólo se debe tener en cuenta que el árbol más pequeño está integrado por cuando menos dos nodos conectados por una arista.

En la figura 8.5 se muestra un ejemplo de un árbol y de un grafo que no es árbol, mientras que en la figura 8.6 se presentan dos ejemplos de bosque.

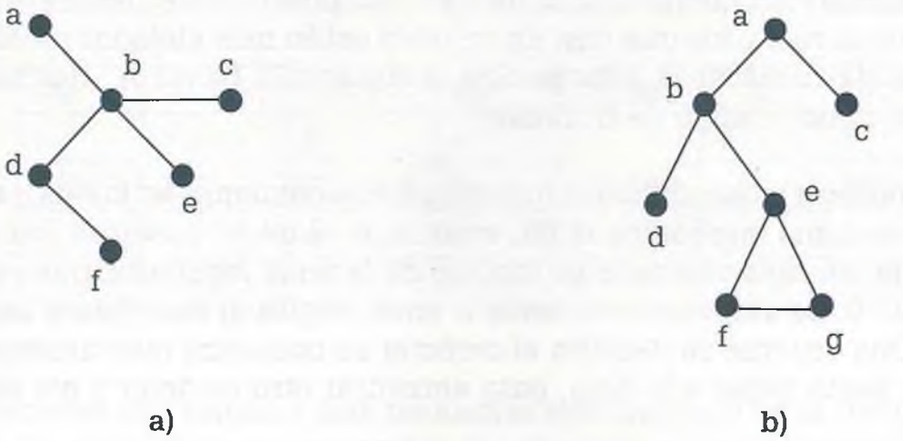


Figura 8.5 El grafo de (a) es un árbol ya que es conexo y no tiene ciclos, el de(b) no es árbol porque tiene ciclos.

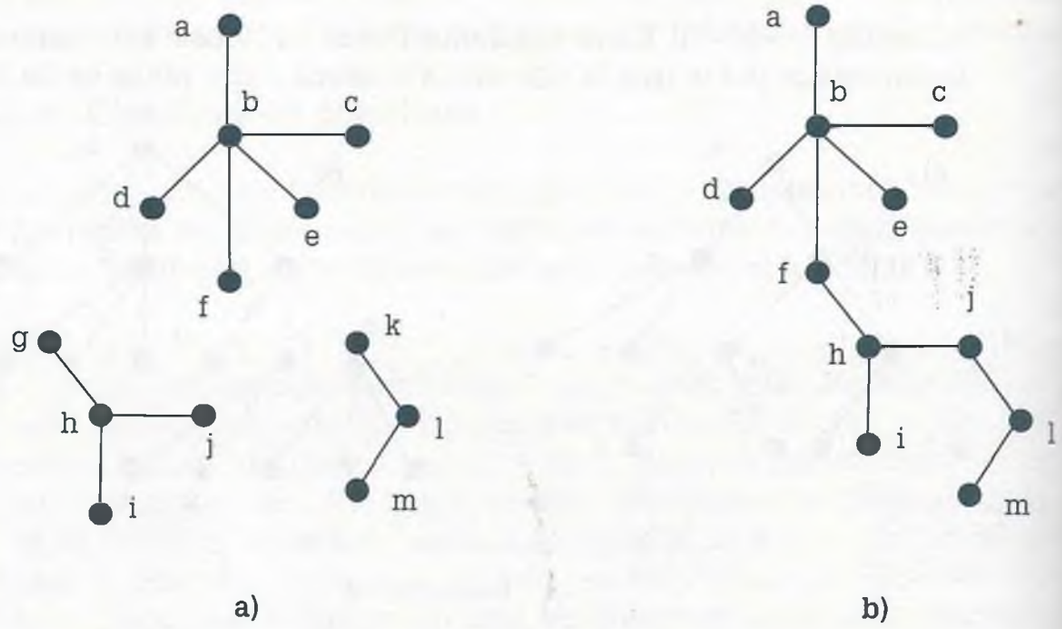


Figura 8.6 Considerando las tres partes que integran el grafo de (**a**) se tiene que éste no es árbol porque no es conexo, pero es bosque porque se compone de varios árboles. El grafo de (**b**) es árbol porque es conexo y no tiene ciclos, además de que es bosque porque es un árbol conectado.

8.5 Árboles con pesos

Para representar caracteres en el código ASCII se usan cadenas de 8 bits, sin embargo se puede aumentar la velocidad de procesamiento o bien aprovechar mejor la memoria de la computadora, mediante una compactación de la información, usando cadenas de diferente longitud. Las cadenas más pequeñas pueden representar a los caracteres que se presentan con más frecuencia, como son las vocales y las consonantes {b, c, d, f, m, n, p, r, s}.

Para codificar la información los bits se colocan en un árbol binario completo donde las cadenas de bits de los caracteres más frecuentes están más cerca de la raíz y los que casi no se usan están más alejados de ella. Esta técnica de codificar la información la desarrolló David A. Huffman y se conoce como "código de Huffman".

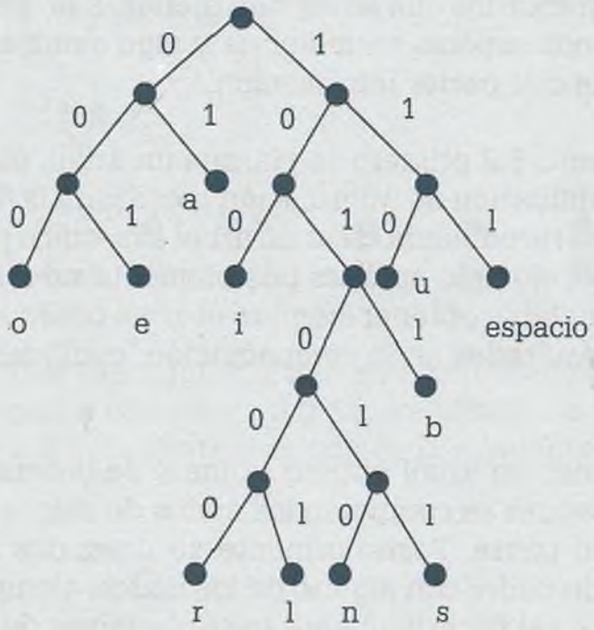
Para codificar o decodificar la información se comienza en la raíz y se avanza por la rama que indica el bit, esto es, si el bit es 1 avanza por la rama derecha, en caso contrario se toma el de la rama izquierda, que está marcado un 0. Se van tomando ceros o unos, según el caso hasta llegar a la hoja. Una vez que se descifra el carácter se comienza nuevamente desde la raíz hasta llegar a la hoja, para encontrar otro carácter y así sucesivamente.

David Albert Huffman
(1925-1999)

Fue un ingeniero eléctrico estadounidense que hizo importantes contribuciones en el estudio de aparatos finitos, circuitos aleatorios, síntesis de procedimientos y diseño de señales, teoría de la información y codificación, diseños de señal para aplicaciones de radar y comunicaciones así como procedimientos de diseño para circuitos lógicos asíncronos; sin embargo es más conocido por su "código de Huffman", un sistema de compresión y codificación de longitud variable.

El código de Huffman puede ser usado en casi cualquier aplicación ya que es un sistema válido para la compresión y posterior transmisión de cualquier dato en formato digital, pudiendo aplicarse a faxes, módems, redes de computadoras y televisión.

Ejemplo 8.2. Usando el siguiente árbol



¿cuál es el significado de la cadena de caracteres?

01101000101100010100100110101111110111001010100110100010000001 01011

Solución. Comenzando por la raíz y tomando bits hasta llegar a la hoja, se tienen los códigos de los siguientes caracteres

000 = o	100 = i	101010 = n	111 = espacio
01 = a	101000 = r	101011 = s	1011 = b
001 = e	101001 = l	110 = u	

de forma que la decodificación del mensaje anterior es

01	101000	1011	000	101001	001	101011	111
a	r	b	o	l	e	s	espacio
1011	100	101010	01	101000	100	000	101011
b	i	n	a	r	i	o	s

Los caracteres con cadenas más pequeñas son producto de la frecuencia del uso. Cuando se desea compactar la información de un documento lo que se recomienda primero es darle una pasada para determinar el núme-

ro de veces que se presenta cada uno de los caracteres que integran dicho documento, posteriormente se elabora el árbol óptimo que permita codificar con menos bits los caracteres que se repiten más y con un número de bits más grande los que se repiten menos. Esto permite que la información ocupe menos espacio en memoria y algo semejante hace el software encargado de compactar información.

En el ejemplo 8.2 primero se plantea un árbol, mismo que se utiliza tanto para la codificación de información como para la decodificación de ésta, no obstante se recomienda crear un árbol específico para cada uno de los usos. El árbol del ejemplo anterior posiblemente no fue el óptimo, sin embargo es recomendable obtener siempre el árbol óptimo con la finalidad de lograr mejores resultados en la compactación, codificación, decodificación y velocidad.

Para obtener un árbol óptimo primero se ordenan los pesos de menor a mayor, después se combinan los nodos de menor peso y se coloca la suma en el nodo padre. Posteriormente se unen dos nuevos nodos, o bien el nuevo nodo padre con alguno de los nodos, siempre y cuando la suma sea la menor, y así sucesivamente hasta terminar de integrar todos los nodos al árbol óptimo.

El árbol binario óptimo se llama así porque su altura es mínima y los pesos o frecuencias están distribuidos de manera que los más pesados están más cercanos a la raíz y los menos pesados se encuentran más alejados de ella, además de que se trata de un árbol binario completo.

Ejemplo 8.3. Se tienen los caracteres y frecuencia de uso de cada carácter en la siguiente tabla. ¿Cuál es el árbol binario óptimo para el código de Huffman?

Carácter	Peso o frecuencia
d	15
e	23
m	5
a	31
t	3
s	16

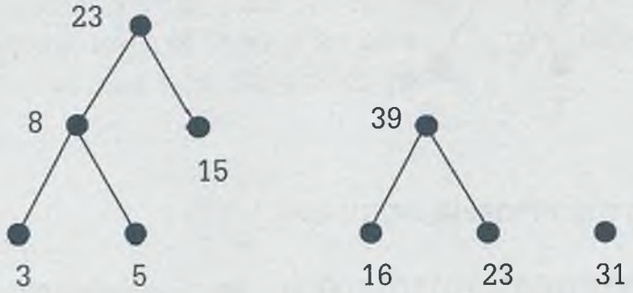
Descifrar con dicho árbol el siguiente mensaje:

00010100111010110110000

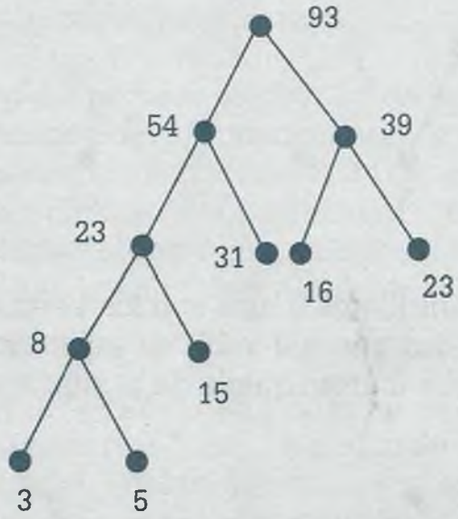
Solución. Para obtener el árbol óptimo, primero se ordenan los pesos de menor a mayor. Después se combinan los nodos de menor peso y se coloca la suma en el nodo padre:

Los pares de nodos candidatos a unir son los vértices con pesos 15 y 16 además de los nodos con pesos 8 y 15; se toman los últimos porque su suma es menor y así el subárbol queda de la siguiente manera:

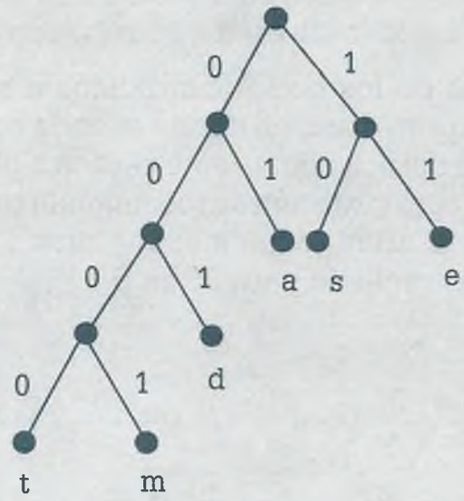
Algunas veces la suma de los nodos candidatos a ser combinados son iguales, como ocurre en este caso en donde el nodo con peso 16 se puede unir con el que está a la derecha de un solo vértice o bien al subárbol cuyo peso también es 23; en esta situación se recomienda unir los nodos nuevos con la finalidad de que el árbol tenga menor altura. Combinando el nodo con peso 16 y el que está solo de peso 23 se tiene:



Sumando ahora los nodos con pesos 23 y 31, finalmente se combinan los nodos con pesos 54 y 39 para obtener el árbol óptimo siguiente:



Por último se cambian los pesos por los caracteres y se les asignan los bits 0 y 1 en las ramas correspondientes:



La decodificación del mensaje siguiente:

00010100111010110110000 es **madeaset**

Es obvio que en la práctica los árboles son más grandes ya que seguramente se tienen todas las letras del alfabeto además de números y símbolos raros, lo que además hace que el número de bits de los caracteres menos frecuentes se incremente de manera considerable, circunstancia que es compensada cuando se tienen cadenas muy cortas de caracteres frecuentes como las vocales.

Por otro lado, no necesariamente deben ser letras las que se codifiquen, también pueden ser palabras de un lenguaje como C, Java o Pascal, en donde sólo se utilizan unas cuantas palabras para editar un programa y en donde palabras como printf() o write() son frecuentes, de forma que si se representan con cadenas de caracteres pequeñas los programas podrían ser compactados para ocupar menos memoria.

8.6 Árboles generadores

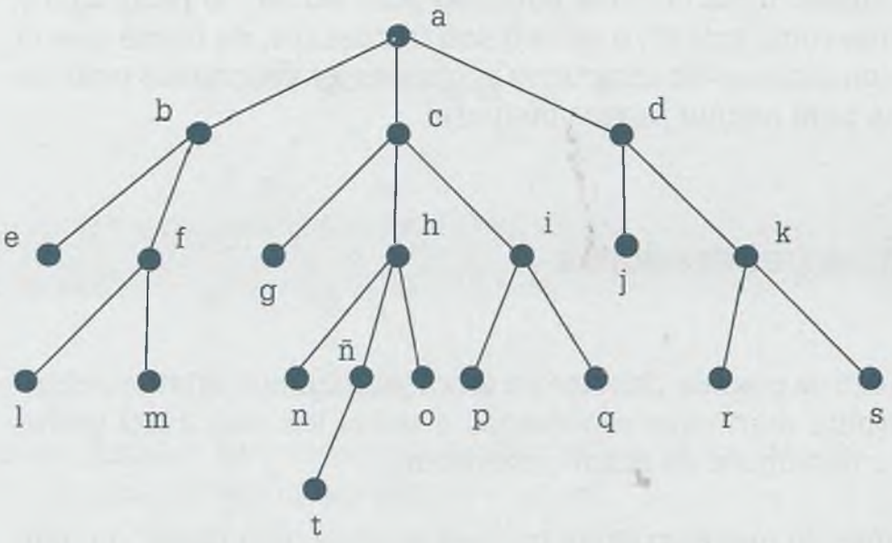
De un grafo conexo es posible obtener un árbol (eliminando aristas redundantes) que permite mantener conectados a todos los nodos del grafo; este árbol recibe el nombre de **árbol generador**.

Existen dos formas en que es posible obtener el árbol generador: usando **búsqueda en profundidad** o bien por medio de **búsqueda a lo ancho**. A continuación se exponen ambos procedimientos para luego aplicarlos en la obtención de un árbol generador en grafos.

8.6.1 Búsqueda a lo ancho

En este procedimiento se comienza en la raíz y después se examinan todos los hijos de la misma de izquierda a derecha. Si la información que se busca no se encuentra en ese nivel, se procede a buscar en el siguiente nivel también de izquierda a derecha, y así sucesivamente hasta encontrar la información. Si se recorrió todo el árbol y no se encontró la información buscada, se debe mandar el mensaje correspondiente.

Ejemplo 8.4. Considérese que en el siguiente árbol se está buscando la letra "n". Primero se busca en la raíz (a), después en todos los nodos dependientes de la raíz de izquierda a derecha (en este caso b, c, d) y como aún no se encuentra la información se procede a buscar en el siguiente nivel (e, f, g, h, i, j, k) y así sucesivamente de manera que el recorrido que se debe realizar para encontrar el nodo buscado es (a, b, c, d, e, f, g, h, i, j, k, l, m, n).



Si se recorre todo un árbol y no se encuentra la información es conveniente mandar el mensaje "información inexistente". La búsqueda a lo ancho es útil cuando los árboles están balanceados o tienen pocos niveles en relación a la información que contienen.

8.6.2 Búsqueda en profundidad

En este caso se comienza en el nodo raíz, después se busca en el hijo de la izquierda y si este nodo tiene hijos se continúa con el de la izquierda y así sucesivamente hasta llegar a la parte más baja del árbol. Si este nodo ya no tiene hijo izquierdo, se continúa con el hijo de la derecha (o el que se encuentra más a la izquierda en caso de no ser un árbol binario) hasta llegar a la hoja. Si no se ha encontrado la información, se recorre el camino andado hasta el nodo inmediato anterior que tenga hijos y cuyos hijos no hayan sido inspeccionados, dando preferencia al de más a la izquierda. Cuando se llega nuevamente a la hoja, se regresa hasta el nodo inmediato anterior que tenga hijos sin inspeccionar y así sucesivamente.

En la búsqueda en profundidad posiblemente se regrese hasta el nodo raíz (lo cual significa que se revisó toda la rama dependiente del hijo izquierdo

de la raíz). Si la raíz tiene hijos que aún no han sido inspeccionados se selecciona el nodo más cercano al hijo izquierdo y todos sus descendientes, dando preferencia a los nodos de la izquierda. Este procedimiento se lleva a cabo hasta encontrar la información o bien recorrer todo el árbol.

En el árbol del ejemplo 8.4 el recorrido para encontrar la letra "n" es:

- Comenzar en la raíz (a).
- Como tiene hijos, continuar con el de la izquierda (a, b).
- Como tiene hijos se sigue con el de la izquierda (a, b, e).

Como el nodo "e" ya es una hoja y no se ha encontrado la información buscada, se regresa al nodo antecesor "b" y se busca en el nodo inspeccionado "f" de tal manera que el recorrido es (a, b, e, f); continuando con el de la izquierda se llega al nodo "l" para tener el recorrido (a, b, e, f, l) y así se continúa hasta llegar a la información buscada. De esta forma el recorrido en profundidad para buscar información en el árbol hasta encontrar la letra "n" es: (a, b, e, f, l, m, c, g, h, n). El recorrido del árbol completo por medio de la búsqueda en profundidad es: (a, b, e, f, l, m, c, g, h, n, ñ, t, o, i, p, q, d, j, k, r, s).

Es importante mencionar que en este caso se está dando preferencia para buscar la información al nodo de la izquierda, pero podría ser al de la derecha; siempre y cuando se respete la forma de búsqueda en todos los casos, se estará hablando de una búsqueda en profundidad.

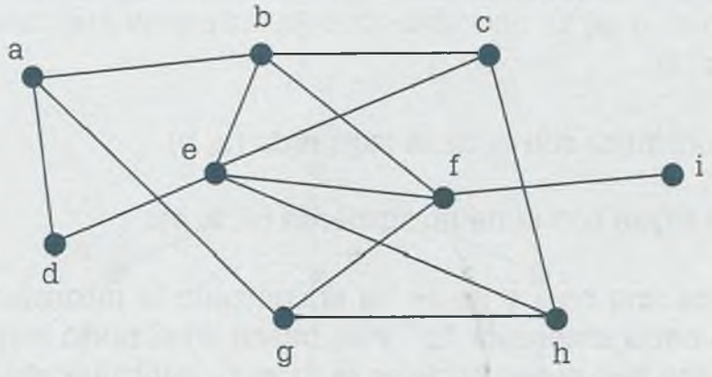
8.6.3 Obtención de árboles generadores

Un árbol es un grafo muy importante ya que permite la estructuración y tratamiento de la información de manera más sencilla. De todo grafo conexo se puede obtener un árbol que recibe el nombre de **árbol generador** y cuyas características son:

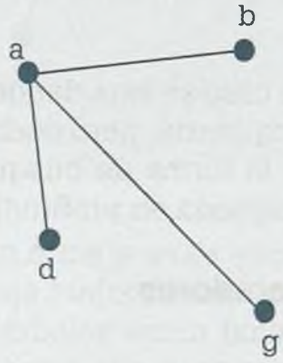
a) Es un grafo conexo.
b) Es un grafo que no tiene ciclos.

Es posible obtener un árbol generador por medio de búsqueda a lo ancho o bien por búsqueda en profundidad, sin embargo en un grafo cualquiera no se puede establecer niveles, ni tampoco se puede identificar el nodo de la izquierda como se realizó en el caso de árboles. Por lo tanto se recomienda establecer un orden de preferencia de búsqueda, mismo que puede ser en forma ascendente en caso de que los nodos sean números o bien alfabéticamente si los vértices se indican con letras.

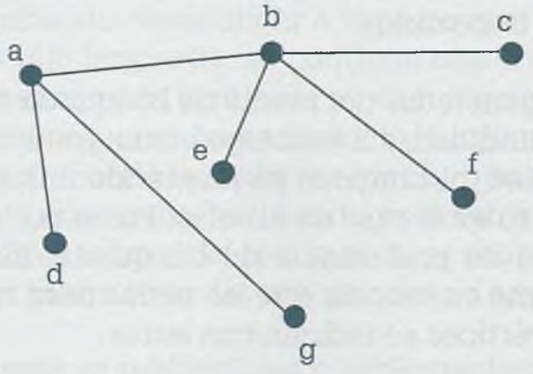
Ejemplo 8.5. Obtener un árbol generador a partir del siguiente grafo utilizando una búsqueda a lo ancho y una búsqueda en profundidad, partiendo del vértice (a) y considerando prioridad en la elección el orden alfabético.



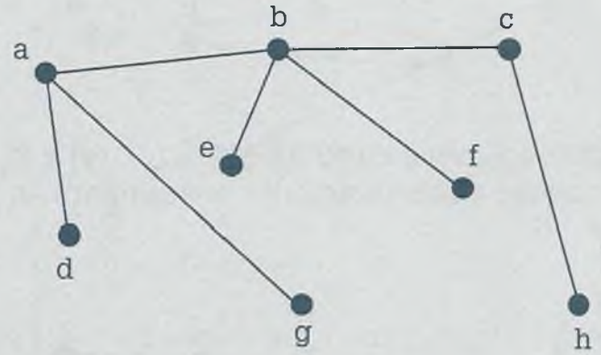
Por búsqueda a lo ancho. El árbol generador por medio de una búsqueda a lo ancho comienza en el vértice origen (a), después se integran los nodos adyacentes a él para obtener el siguiente grafo:



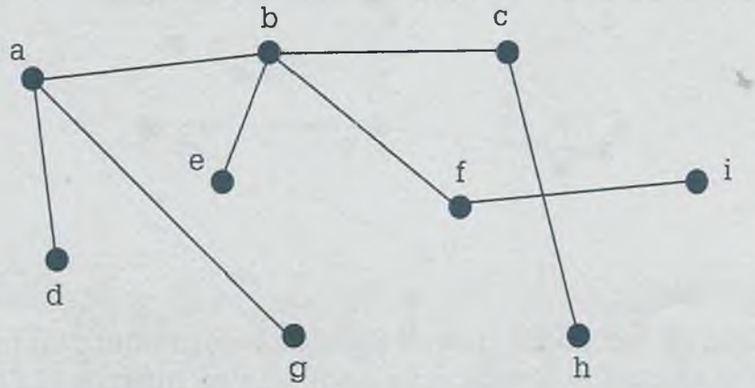
El nodo (a) ya no tiene nodos adyacentes a él que no hayan sido integrados y alfabéticamente el nodo (b) es el siguiente a inspeccionar, de forma que se anexan los nodos adyacentes a éste cuidando que no se formen ciclos con los nodos ya existentes y lo que se obtiene es la siguiente figura:



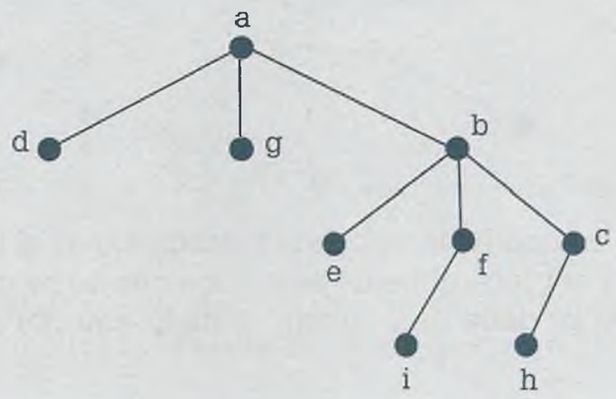
Los nodos (a) y (b) del grafo anterior ya no tienen nodos adyacentes sin inspeccionar, y de los nodos restantes el más pequeño alfabéticamente es el (c). Por lo tanto se integra al árbol generador el nodo (h):



Finalmente el árbol generador queda de la siguiente forma:

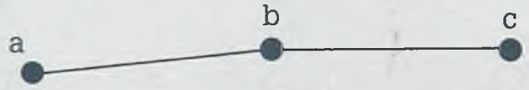


Se puede observar que se trata de un árbol porque es conexo y no tiene ciclos. Es un árbol "libre" porque no tiene raíz ya que cualquiera de sus nodos puede hacer las veces de raíz. Si se considera que la raíz es el vértice inicial (a) el árbol generador es el siguiente:

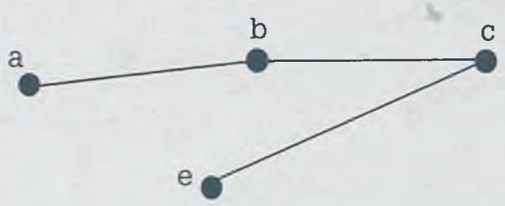


Por búsqueda en profundidad. Considerando el inicio del árbol generador en el nodo (a) y de los vértices adyacentes (b), (d) y (g), se selecciona el vértice (b) por ser el menor alfabéticamente y se integra al árbol generador como se muestra en la siguiente figura:

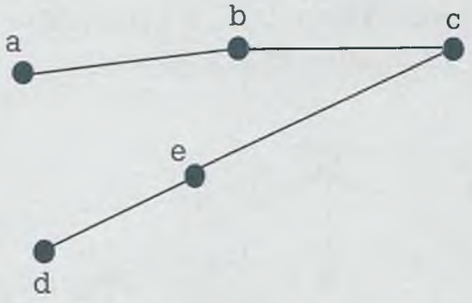
Ahora los nodos potenciales a integrarse son (c), (e) y (f), y como el nodo actual es el (b) entonces alfabéticamente el siguiente es (c), por lo que se tiene:



A partir del nodo (c), los nodos potenciales a integrar son (e) y (h) por lo que tomando el más pequeño alfabéticamente se tiene:

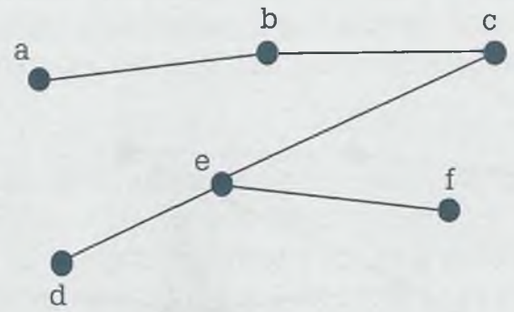


Luego del nodo (e) los nodos que se pueden seleccionar son (d), (f) y (h), y como (d) es el alfabéticamente más pequeño se integra al árbol generador:

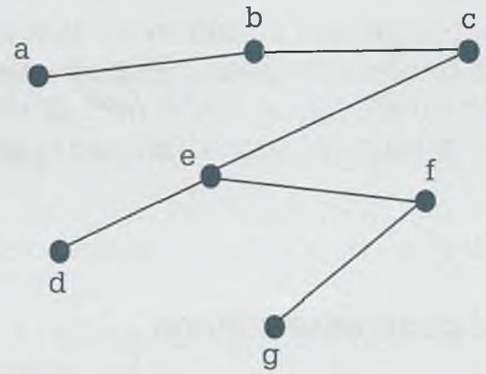


Del nodo (d) el único nodo que se puede seleccionar es el (a) pero con ello se forma un ciclo y el procedimiento establece que no se deben de formar ciclos, por lo tanto se debe de regresar al nodo anterior para ver si aún

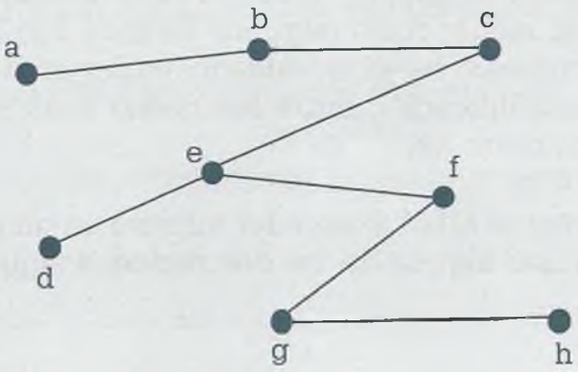
existen nodos que no han sido integrados. De esta forma se integra al árbol generador el nodo (f):



A partir del nodo (f) se puede seleccionar (g) o (i), y por la prioridad alfabética se selecciona (g):

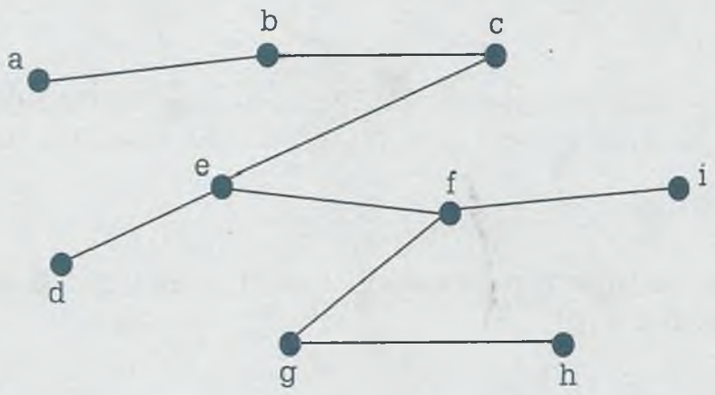


Desde el nodo (g) sólo se puede seleccionar (h) por lo que se tiene:



A partir de la posición del nodo (h) ya no es posible seleccionar ningún nodo, ya que tanto (c) y (e) están en el árbol y volverlos a seleccionar im-

plicaría la formación de ciclos. Por esta razón se regresa primero al nodo (g) y después al (f) de donde se puede seleccionar el nodo (i) que permite tener el siguiente árbol generador final, obtenido por medio de una búsqueda a profundidad partiendo del nodo (a) y con prioridad de selección en orden alfabético:



Se observa que el árbol generador obtenido por medio de búsqueda a profundidad es diferente del que se obtuvo por búsqueda a lo ancho. Esto muestra que un grafo puede tener dos o más árboles generadores, los cuales se obtienen en función del procedimiento aplicado.

8.6.4 Árbol generador mínimo

Se llama árbol generador mínimo de un grafo conexo a aquel que permite mantener unidos a todos los vértices y que no tiene ciclos, además de que es la forma más barata o corta ya que la trayectoria o costo es mínimo. Existen varios campos en donde es conveniente mantener funcionando los sistemas al menor costo (algunos de ellos son las redes telefónicas, eléctricas, carreteras, de alcantarillado, etc.) y en donde es conveniente mantener la comunicación entre los nodos pero sin aristas redundantes o demasiado costosas.

Para obtener el árbol generador mínimo en un grafo conexo con pesos se puede utilizar alguno de los dos métodos siguientes: el de Prim o el de Kruskal.

Método de Prim

Robert Prim desarrolló un método para obtener un árbol generador mínimo a partir de un grafo conexo etiquetado. Este método se puede poner en práctica aplicando un algoritmo que se utiliza cuando se desea eliminar conexiones redundantes y dejar sólo las aristas del grafo que tengan menor distancia o costo, además de ser sólo las necesarias para mantener conectados todos los nodos.

En este método los vértices se dividen en dos conjuntos: vértices integrados (I) —que son los que forman parte del árbol generador mínimo— y vértices no integrados (N). El conjunto ***Árbol*** contiene todas las aristas que se van integrando en cada iteración, y así en cada paso se agrega un vértice en el conjunto I mientras que el conjunto N es disminuido en uno. El algoritmo se puede resumir de la siguiente forma:

1) Se selecciona una arista de costo mínimo (s, x).
2) Se integran los vértices de esa arista al conjunto I = {s, x}, ya que serán los primeros nodos que conforman el árbol generador mínimo.
3) Todos los demás nodos del grafo pertenecen al conjunto N = N – I.
4) El Árbol tiene como primera arista (s, x), esto es, Árbol = {(s, x)}.
5) Mientras el conjunto N sea diferente de vacío ($N \neq \varnothing$) iterar. En caso contrario terminar.
6) Seleccionar nueva arista con costo mínimo (w, x). Una característica es que w debe estar contenido en I ($w \in I$) mientras que x debe estar en N ($x \in N$).
7) Se agrega al árbol la arista mínima seleccionada: Árbol = Árbol $\cup$ (w, x).
8) Se agrega al conjunto I el nuevo nodo seleccionado: I = I $\cup$ (x).
9) Se elimina el nodo seleccionado del conjunto N, esto es, N = N – (x).
10) Regresar al paso 5.

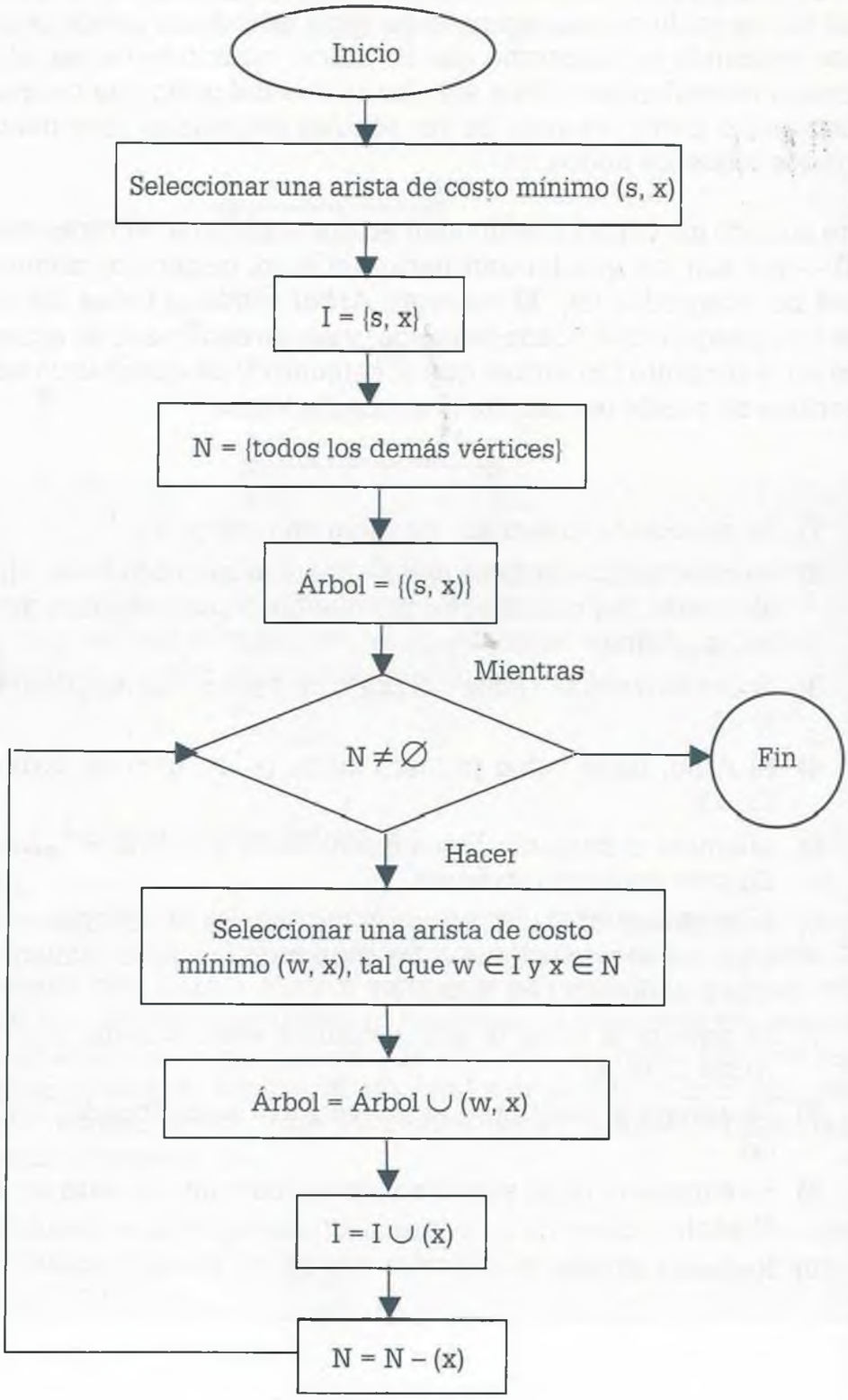
Robert Clay Prim
(1921)

Nació en 1921 en Sweetwater, Estados Unidos. En 1941 terminó la carrera de ingeniero electricista en la Universidad de Princeton, posteriormente en la misma Universidad obtuvo el grado de doctor en el área de matemáticas y fue investigador de 1948 a 1949.

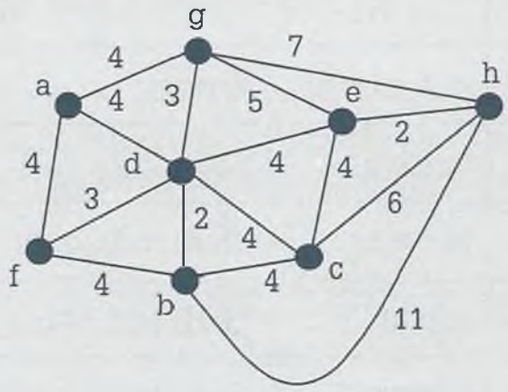
En plena segunda guerra mundial, Prim trabajó como ingeniero para *General Electric*. Desde 1944 hasta 1949 fue contratado por la *United States Naval Ordenance Lab* como ingeniero y más tarde como matemático. En los laboratorios *Bell* trabajó como director de investigación matemática desde 1958 hasta 1961 y ahí Prim desarrolló el conocido Algoritmo de Prim. Después de su estancia en los laboratorios *Bell*, Prim pasó a ser vicepresidente de investigación en *Sandia National Laboratorios*.

Durante su carrera en los laboratorios *Bell*, Robert Prim junto a su compañero Joseph Kruskal desarrolló dos algoritmos diferentes para encontrar los árboles generadores mínimos en un grafo ponderado.

Otra forma de expresar este algoritmo es por medio de un diagrama de flujo como el que se muestra a continuación:



Ejemplo 8.6. Determinar en el siguiente grafo el árbol generador mínimo aplicando el método de Prim.



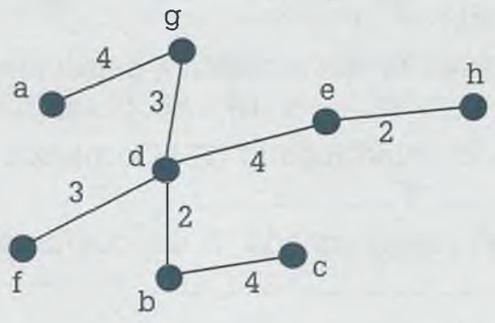
1) Supóngase que se selecciona la arista (b, d) ya que su costo es de 2.
2) I = {b, d}.
3) N = {a, c, e, f, g, h}.
4) Árbol = {(b, d)}.
5) Mientras (N ≠ ∅) iterar, en caso contrario terminar. En este caso N es diferente de vacío, por lo tanto se debe iterar.
6) Seleccionar nueva arista. Considérese que se selecciona (d, f) ya que su costo es de 3. No puede ser (e, h) porque un requisito es que uno de los vértices de la arista esté seleccionado (w ∈ I) y el otro no (x ∈ N). En nuestro caso (d ∈ I) y el otro no (f ∈ N).
7) Se agrega al árbol la arista mínima seleccionada (Árbol = Árbol ∪ {(d, f)} = {(b, d)} ∪ {(d, f)} = {(b, d),(d, f)}.)
8) Integrar el nodo seleccionado (f) al conjunto I = I ∪ (f) = {b, d, f}.
9) Eliminar el nodo seleccionado (f) del conjunto N = {a, c, e, g, h}.
10) Regresar al paso 5.

En la siguiente tabla se resume el resultado de cada una de las iteraciones:

Iteración	I	N	Árbol
1	{b,d}	{a,c,e,f,g,h}	{(b,d)}
2	{b,d,f}	{a,c,e,g,h}	{(b,d),(d,f)}
3	{b,d,f,g}	{a,c,e,h}	{(b,d),(d,f),(d,g)}
4	{b,d,f,g,a}	{c,e,h}	{(b,d),(d,f),(d,g),(g,a)}
5	{b,d,f,g,a,c}	{e,h}	{(b,d),(d,f),(d,g),(g,a),(b,c)}
6	{b,d,f,g,a,c,e}	{h}	{(b,d),(d,f),(d,g),(g,a),(b,c),(d,e)}
n–1	{b,d,f,g,a,c,e,h}	∅	{(b,d),(d,f),(d,g),(g,a),(b,c),(d,e),(e,h)}

Aquí n es el número de vértices que integran el grafo.

De esta forma, con el conjunto Árbol se puede obtener el siguiente árbol generador mínimo:



Cuando se tienen aristas de la misma longitud, como en el ejemplo anterior, es posible obtener más de un árbol generador mínimo, por lo tanto si en el momento de seleccionar la arista se opta por otra rama de igual longitud (o peso) entonces el árbol generador mínimo es diferente.

Método de Kruskal

Este método se destaca por integrar al árbol generador mínimo a aquellas aristas que tengan menor costo, cuidando siempre que no se formen ciclos. El algoritmo se puede resumir de la siguiente forma:

1) Ordenar los costos de las aristas del grafo en forma ascendente y colocar en el conjunto N las aristas, de acuerdo a este orden.
2) Se incluye la arista con menor costo Árbol = {(s, t)}.
3) Se resta del conjunto N la arista seleccionada N = N – (s, t).
4) Se registra en un contador el número de aristas incluidas, en este caso C = 1, ya que sólo se ha incluido una arista.
5) Mientras C ≤ (n – 1) iterar. En caso contrario finalizar.
6) Si Árbol ∪ (w, x) no forma ciclos, entonces
   Árbol = Árbol ∪ (w, x)
   C = C + 1
7) Regresar a paso 4.

Aquí se tiene que:

Árbol: conjunto que contiene las aristas que integran el árbol generador mínimo.

N: conjunto que contiene las aristas que aún no han sido seleccionadas.

C: variable para contar las aristas incluidas.

n: número de vértices que conforman el grafo.

(s, t): arista con menor costo que se integra al árbol generador.

(w, x): arista a integrarse al árbol generador, siempre y cuando con ella no se formen ciclos.

Joseph B. Kruskal

Fue investigador del Centro de Matemáticas de los Laboratorios Bell, y en 1956 descubrió el algoritmo para obtener un árbol generador mínimo a partir de un grafo ponderado: el árbol del costo total mínimo también llamado árbol recubridor euclídeo mínimo. Este problema es un problema típico de optimización combinatoria, y fue considerado originalmente por Otakar Boruvka (1926) mientras estudiaba la necesidad de electrificación rural en el sur de Moravia en Checoslovaquia.

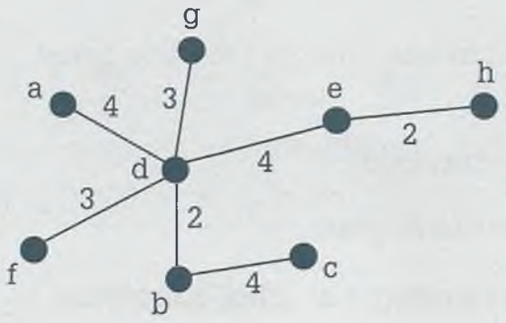
La aplicación típica de este problema es el diseño de redes telefónicas: una empresa con diferentes oficinas trata de trazar líneas de teléfono para conectarlas unas con otras. La compañía telefónica ofrece esta interconexión, pero presenta tarifas o costos diferentes por conectar cada par de oficinas. ¿Entonces cómo conectar las oficinas al mínimo costo total?

La formulación de este problema también ha sido aplicada para determinar soluciones en diversas áreas: diseño de redes de transporte, diseño de redes de telecomunicaciones (TV por cable, sistemas distribuidos), interpretación de datos climatológicos, visión artificial (análisis de imágenes), extracción de rasgos de parentesco, análisis de clusters y búsqueda de superestructuras de quasar, plegamiento de proteínas, reconocimiento de células cancerosas, y otros.

Ejemplo 8.7. La siguiente tabla muestra los resultados obtenidos en cada una de las iteraciones del algoritmo de Kruskal aplicado al caso del ejemplo 8.6.

C	Árbol	N
0	∅	{(b,d),(e,h),(d,g),(d,f),(a,d),(a,f),(a,g),(b,c),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
1	{(b,d)}	{(e,h),(d,g),(d,f),(a,d),(a,f),(a,g),(b,c),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
2	{(b,d),(e,h)}	{(d,g),(d,f),(a,d),(a,f),(a,g),(b,c),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
3	{(b,d),(e,h),(d,g)}	{(d,f),(a,d),(a,f),(a,g),(b,c),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
4	{(b,d),(e,h),(d,g),(d,f)}	{(a,d),(a,f),(a,g),(b,c),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
5	{(b,d),(e,h),(d,g),(d,f),(a,d)}	{(a,f),(a,g),(b,c),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
6	{(b,d),(e,h),(d,g),(d,f),(a,d),(b,c)}	{(a,f),(a,g),(b,f),(c,d),(d,e),(c,e),(e,g),(c,h),(g,h),(b,h)}
7	{(b,d),(e,h),(d,g),(d,f),(a,d),(b,c),(d,e)}	{(a,f),(a,g),(b,f),(c,d)(c,e),(e,g),(c,h),(g,h),(b,h)}

De esta forma el árbol generador mínimo queda de la siguiente manera:



En relación con los ejemplos 8.6 y 8.7 es necesario observar que el árbol generador mínimo obtenido por el método de Kruskal es diferente al que se obtuvo por el método de Prim, debido principalmente a características propias del método cuando se presentan aristas con la misma ponderación.

8.7 Recorrido de un árbol

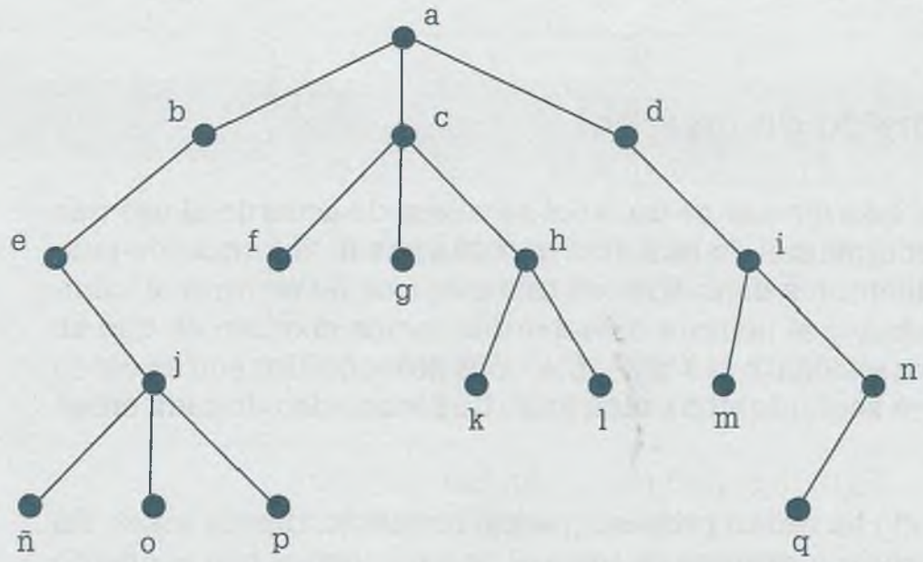
Por lo general la información de un árbol se coloca de acuerdo al uso que se le dará posteriormente, de tal forma que una misma información puede servir para diferentes usos. Existen tres maneras de recorrer la información de un árbol, y el nombre del recorrido indica el orden en que se coloca el padre en relación a sus hijos. Los tipos de recorridos son en orden primero, en orden segundo y en orden final. La descripción de cada orden es la siguiente:

a) Recorrido en **orden primero** (padre, izquierdo, demás hijos). En este recorrido **primero** se toma el padre, luego el hijo izquierdo y al final los demás hijos. Se comienza por la raíz, después se sigue por el nodo de la izquierda, si este nodo tiene hijos se sigue por el de la izquierda hasta llegar a la hoja. Si esta hoja tiene hermanos se toma el que está más cercano a ella (más a la izquierda). Después de que se termina con la rama izquierda, continúa con la rama más cercana a ella y así sucesivamente hasta terminar con el recorrido de todo el árbol.

b) Recorrido en **orden segundo** (izquierdo, padre, demás hijos). En este recorrido primero se toma el hijo izquierdo, **segundo** el padre y al final los demás hijos. Comienza con la hoja que se encuentra más a la izquierda del árbol, después se regresa al padre y posteriormente a todos los hermanos, después se regresa al padre de esta rama y con las ramas de éste (tomando siempre la que está más a la izquierda) y así sucesivamente hasta terminar el recorrido del árbol completo.

c) Recorrido en **orden final** (izquierdo, demás hijos, padre). En éste recorrido se toma primero el hijo izquierdo, después los demás hijos y al **final** el padre. Se comienza en la hoja que se encuentra más a la izquierda del árbol, después se continúa con los hermanos, si éstos tienen hijos primeramente recorre los hijos y hasta el final el padre, dando preferencia a los hijos de la izquierda y hasta el final el padre. En este tipo de recorrido lo último que se recorre es la raíz, ya que tienen preferencia los hijos sobre el padre.

Ejemplo 8.8. Considérese el siguiente árbol:



Entonces se tienen los siguientes recorridos.

- Primero: (a,b,e,j,ñ,o,p,c,f,g,h,k,l,d,i,m,n,q).
- Segundo: (e,ñ,j,o,p,b,a,f,c,g,k,h,l,d,m,i,q,n).
- Final: (ñ,o,p,j,e,b,f,g,k,l,h,c,m,q,n,i,d,a).

8.7.1 Recorridos en árboles etiquetados

En el área de la computación los árboles etiquetados se usan para evaluar expresiones matemáticas, y las constantes o variables se ubican en las hojas mientras que los operadores (signos aritméticos o funciones) se colocan como nodos intermedios.

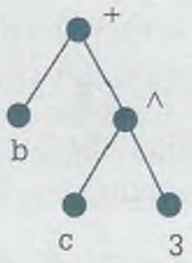
Ejemplo 8.9. Elaborar el árbol binario completo que representa la expresión:

$$\frac{a(b+c^3)}{b-(c+d)e}$$

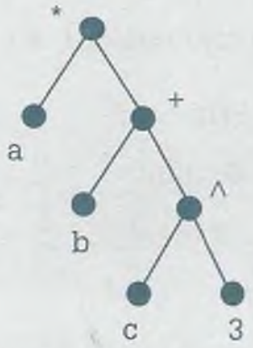
y determinar los recorridos en orden primero, segundo y final.

Solución. De acuerdo a la jerarquía de operación que se usa en computación el árbol se estructura de la siguiente forma:

c^3

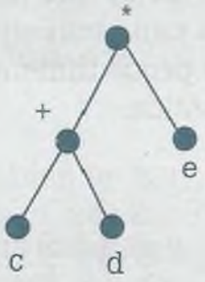


$b + c^3$

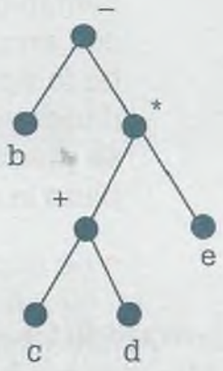


$a(b + c^3)$

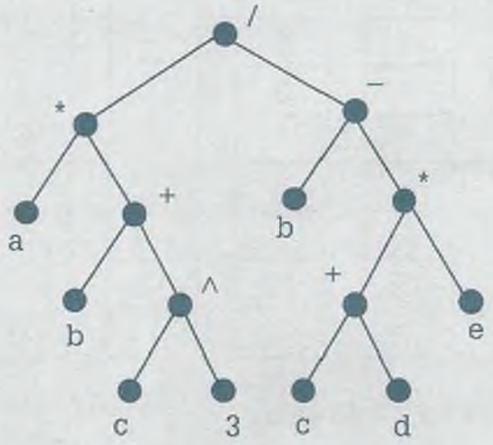
$c + d$



$(c + d)e$



$b - (c + d)e$



$$\frac{a(b + c^3)}{b - (c + d)e}$$

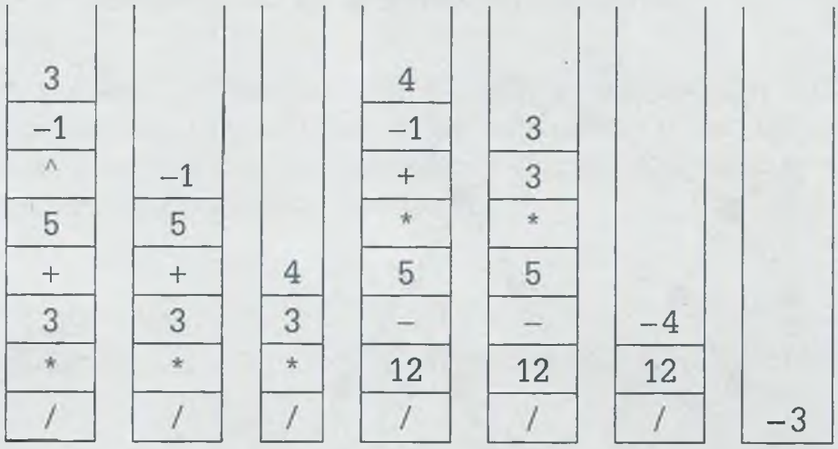
Hay que observar cómo crece el árbol de acuerdo con la literal de la ecuación: si la literal se encuentra a la derecha crece hacia la derecha y si se encuentra a la izquierda crece en ese mismo sentido.

El recorrido del árbol en orden primero, segundo y final es:

Primero:	/*a +b^c3 –b* +cde
Segundo:	a*b +c^3/b –c +d*e
Final:	abc3^ + *bcd +e*–/

El recorrido en "orden primero" también recibe el nombre de "notación polaca" y es una de las maneras más usadas para evaluar expresiones matemáticas dentro de la computación. La forma en que se realiza es colocando en una pila operadores aritméticos y variables, hasta que se presentan dos variables seguidas se sacan de la pila junto con el operador mas cercano y se realiza la operación correspondiente, se guarda el resultado de la operación en la pila hasta que nuevamente se presenten dos cantidades seguidas, se sacan junto con su operador, se realiza la operación y se guarda en la pila. Este procedimiento se repite hasta evaluar por completo la expresión matemática.

Ejemplo 8.10. Supóngase que en el árbol anterior las variables tienen los valores a = 3, b = 5, c = –1, d = 4 y e = 3. Si el recorrido en orden primero es /*a + b^c3 –b* + cde, la forma en que se evalúa la expresión es:



Estados de la pila.

Se puede observar en primer lugar que se introducen en la pila todos los valores y signos aritméticos, hasta que se tengan dos variables (o cantida-

des seguidas), en este caso cuando están seguidos –1 y 3 que son los valores de c y su exponente. En el siguiente estado de la pila se observa que se sacó –1 y se elevó a la potencia 3, (–1)^3. Como nuevamente quedan dos valores juntos, se extraen de la pila y se les aplica el operador más cercano a ellos, "+", por lo que se obtiene 5 + (–1) = 4 y se vuelve a guardar el resultado en la pila.

En el tercer estado de la pila nuevamente vuelven a quedar juntas dos cantidades, mismas que se extraen de la pila con su operador más cercano para obtener 3*4=12. Como ya no existen dos cantidades seguidas, es necesario introducir a la pila más valores.

Posteriormente se extraen de la pila –1 y 4 juntamente con el operador "+" para llevar a cabo la operación (–1 + 4 = 3) y así sucesivamente hasta obtener como resultado el –3 que está en el último estado de la pila, el cual es el mismo resultado obtenido al evaluar la expresión matemática:

$$\frac{a(b + c^3)}{b(c + d)e} = \frac{3(5 + (-1)^3)}{5 - (-1 + 4)^3} = -3$$

El recorrido en "orden final" también es muy útil en la evaluación de expresiones matemáticas, sólo que en este caso se introducen en la pila los valores de las variables hasta que llega un signo aritmético. En este momento se extraen de la pila el signo aritmético y los valores que están más cerca de él, se realiza la operación y se guarda el resultado. Cuando llega otro signo aritmético se vuelven a extraer dos valores juntamente con el signo para realizar la operación correspondiente y guardar el resultado. Estos pasos se llevan a cabo hasta terminar de evaluar la expresión matemática como se muestra a continuación.

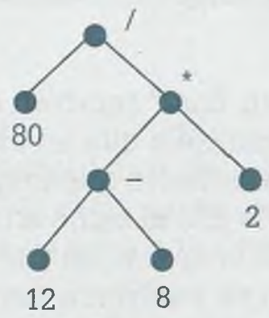
Ejemplo 8.11. Supóngase nuevamente que los valores son a = 3, b = 5, c = –1, d = 4 y e = 3. Si el recorrido en orden final es abc3^ + *bcd + e*–/, la forma en que se evalúa la expresión es:

^			+	*			
3	+		4	3	–		
–1	–1	*	–1	3	9		
5	5	4	5	5	5	–4	
3	3	3	12	12	12	12	–3

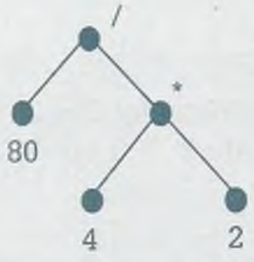
Primero se introducen valores a la pila hasta que se tenga un signo aritmético. Al llegar el signo "^" se extraen los valores más cercanos a él, se lleva a cabo la operación (–1)^ 3 = –1 y se guarda el resultado en la pila como lo muestra la figura.

La siguiente información de acuerdo al orden final es el signo "+", como lo muestra la pila correspondiente, mismo que se saca de la pila junto con los valores que están cercanos a él para realizar la siguiente operación 5 + (–1) = 4. Se continúa de esta manera guardando la información en la pila y sacando las dos últimas cantidades guardadas, una vez que se presenta un signo aritmético, para llevar a cabo la evaluación correspondiente hasta obtener el resultado que se muestra en el último estado de la pila.

En el recorrido por "orden segundo" la evaluación de la información en el árbol debe ser por niveles y de izquierda a derecha. Por ejemplo, si los valores son a = 80, b = 12, a = 5, d = 3 y e = 2, primero se evalúan los niveles más bajos y si hay empate en cuanto al nivel tienen prioridad las operaciones que están más a la izquierda. Sustituyendo valores por variables la evaluación se lleva a cabo de la siguiente forma:



Después de sumar 5 + 3 = 8



Después de restar
12 – 8 = 4

Después de multiplicar
4*2 = 8

Finalmente
se divide
80/8 = 10

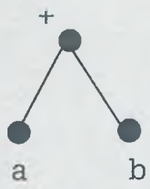
Cuando se evalúa una expresión matemática puede haber más de una hoja que tenga el mismo valor, así como también es frecuente que dos o más nodos intermedios tengan el mismo signo aritmético.

Ejemplo 8.12. Considérese que el recorrido de un árbol binario en orden final es ab + cd/ef – gb + *–*. Determinar:

a) El árbol.

b) El recorrido en orden primero y segundo.

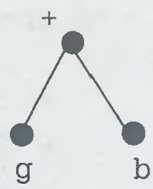
Solución. En el recorrido en orden final una operación se lleva a cabo hasta que se tienen las dos cantidades y el signo de la operación que se realizará. El procedimiento completo para estructurar el árbol es:



ab +

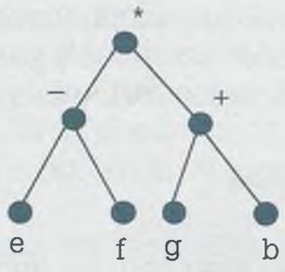
cd/

ef–

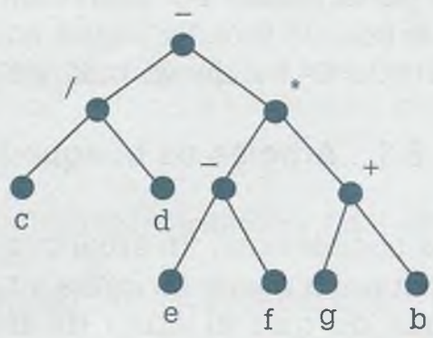


gb +

En todos los casos anteriores primero se deben de tener dos cantidades y la operación aritmética para de esa manera unir los nodos. La operación aritmética se lleva a cabo con las dos cantidades más cercanas a la izquierda del signo. Una vez que se realiza la operación el resultado se considera como una cantidad, misma que se puede tomar en el momento en que se presente un signo. Hasta ahora se lleva evaluado ab + cd/ef – gb + del recorrido en orden posterior.



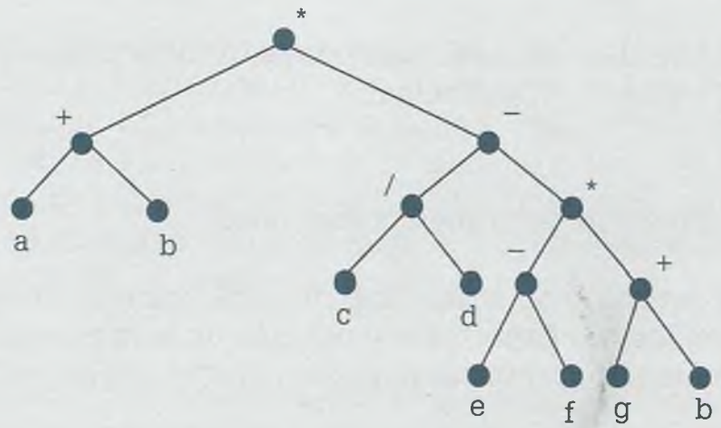
Dos últimos resultados con el nuevo operador aritmético ef – gb+*



Más recientes resultados con el nuevo operador aritmético ab + cd/ef – gb+* –

Finalmente se obtiene el siguiente árbol.

a)



b) Recorridos en orden:

- Primero: * + ab – /cd* – ef + gb
- Segundo: a + b*c/d – e – f*g + b

8.8 Búsquedas

Se puede considerar que uno de los usos principales de la computadora es guardar la información para después recuperarla en el orden deseado y en forma rápida. Cuando la información es pequeña no hay ningún problema ya que el tiempo en que encuentra la información almacenada es relativamente pequeño, pero conforme ésta aumenta el tiempo de respuesta es importantísimo. Por esta razón es necesario guardar los datos de forma que sea posible acceder a ellos en un tiempo razonable y para esto se utilizan árboles de búsqueda binarios (ABB), árboles AVL y árboles B.

8.8.1 Árboles de búsqueda binarios

Es posible crear un árbol que desde el momento en que se captura la información quede de forma que sea relativamente fácil acceder a ella, además de que el árbol de búsqueda binario es sencillo de crear y manipular.

Ejemplo 8.13. Crear un árbol de búsqueda binario con la siguiente información: 30, –10, 3, 4, 9, 68, 50, 30, 3, 7, 6, 72, 98, –7, 56, 31, 58.

Solución. Se entiende que la raíz es el primer dato, en este caso el 30, y que los siguientes datos de colocarán a la izquierda si son menores que 30 o a la derecha si son mayores o iguales a 30. Lo mismo sucederá en los nodos restantes: el nuevo dato estará a la izquierda de un nodo cualquiera si es menor y a la derecha si es mayor o igual a él. Con estas reglas para estructurar el árbol se tendrá lo siguiente:

Una vez estructurado el árbol es posible obtener la información de sus nodos siguiendo las mismas reglas que se usaron para elaborarlo: se buscará a la izquierda si es menor que la información del nodo y a la derecha si es mayor o igual a él. Por ejemplo, si el dato a buscar es el número 56, dado que es mayor que 30 se busca a la derecha ya que se tiene la certeza de que todos los datos que están a la izquierda de 30 son menores que él. A la derecha de 30 está el 68, pero como el 56 es menor ahora se busca a la izquierda para llegar al 50 y como el 56 es mayor se busca a la derecha para localizarlo finalmente. Si en algún caso se llega a una hoja y no se encuentra la información que se busca, entonces se deberá mandar un mensaje de "información inexistente".

Por otro lado, los recorridos en orden primero, segundo y final son:

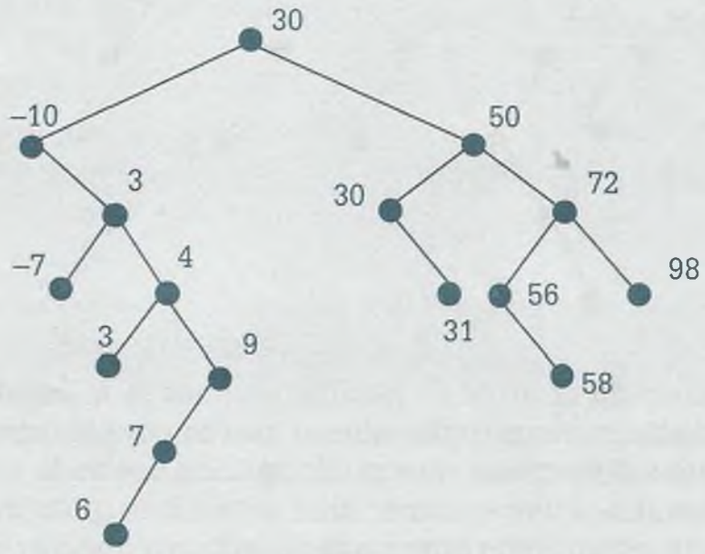
Primero: 30, –10, 3, –7, 4, 3, 9, 7, 6, 68, 50, 30, 31, 56, 58, 72, 98

Segundo: –10, –7, 3, 3, 4, 6, 7, 9, 30, 30, 31, 50, 56, 58, 68, 72, 98

Final: –7, 3, 6, 7, 9, 4, 3, –10, 31, 30, 58, 56, 50, 98, 72, 68, 30

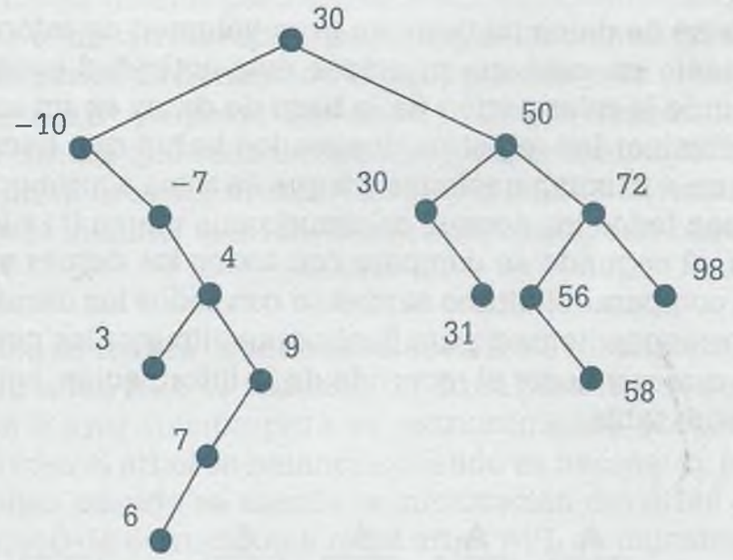
Hay que observar que el recorrido en orden segundo permite obtener la información del árbol ordenada en forma ascendente. Esa es una de las ventajas de acomodar la información en un árbol con estas características, ya que se puede encontrar un dato en forma relativamente fácil y además se puede obtener el total de la información del árbol en forma ordenada.

Considérese ahora que se desea dar de baja algunos elementos del árbol. Si el nodo que se da de baja es una hoja no hay ningún problema, solamente se quita el nodo y la demás información no se mueve de su lugar. Pero cuando el nodo es intermedio (o sea que tiene hijos) es recomendable establecer cuál de los hijos tendrá prioridad para ocupar el lugar del padre. Se acostumbra dar prioridad al nodo izquierdo, esto implica que al desaparecer el padre el hijo izquierdo tomará su lugar cambiando la reestructuración del árbol. Por ejemplo, si se da de baja el nodo 68 el árbol quedará de la siguiente manera



Se entiende que si el nodo que ocupa la posición del padre tiene hijo derecho, ese hijo derecho con toda su descendencia pasa a formar parte de la rama derecha como ocurre con el nodo 56 y su descendiente 58, que dependían del 50 y ahora dependen del 72.

Si ahora se da de baja al nodo 3 el árbol queda de la siguiente manera:



En este caso se observa que cuando hay información repetida, como es el nodo 3, el que se da de baja es el primero que se encuentra ya que está más cercano a la raíz.

Un problema importante de los árboles de búsqueda binario es que algunas veces crecen en forma descontrolada. Cuando el volumen de información es considerable el tiempo de búsqueda se incrementa, ya que se trata de árboles no balanceados, y en estos casos es conveniente utilizar árboles AVL o árboles B.

8.9 Aplicación de los árboles

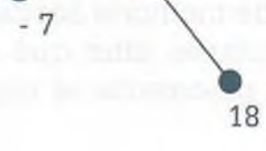
La estructura de árbol, independientemente de si se trata de árboles binarios, AVL o B, se usa principalmente para guardar la información organizada de tal manera que sea posible tener un rápido acceso a ella. La diferencia principal que permite decidir qué tipo de árbol usar depende de la forma en que está estructurada la información, pero sobre todo del volumen de la misma (si la información es poca, entonces hay que usar árboles binarios o AVL dependiendo de la forma en que esté organizada; si el volumen de información es grande entonces hay que usar árboles B) ya que cuando es posible manipular la información en memoria principal, los árboles binarios y AVL son recomendables. Sin embargo, cuando no se puede transferir la información completamente de memoria secundaria a memoria principal para posteriormente manipularse, sino que se trae parcialmente a memoria principal y después de procesarla se regresa al

dispositivo de almacenamiento secundario, entonces es recomendable la utilización de árboles B o B++. Por ejemplo, para eliminar información duplicada en una base de datos.

Si la base de datos no tiene un gran volumen de información y es posible su manejo en memoria principal, esta actividad normalmente se realiza colocando la información de la base de datos en un arreglo de forma que para eliminar los registros duplicados habrá que hacer en el peor de los casos $nn = n^2$ comparaciones, ya que se toma el primer elemento y se compara con todos los demás, colocando una marca (*) a los elementos duplicados, el segundo se compara con todos los demás y así sucesivamente hasta comparar el último elemento con todos los demás y colocar las marcas correspondientes, para finalmente eliminar las posiciones que tengan dicha marca y hacer el recorrido de la información, como se muestra en la siguiente tabla.

						Finalmente
	A	A	A	A	A	A
1	20	20	20	20	20	20
2	3	3	3	3	3	3
3	–7	–7	–7	–7	–7	–7
4	50	50	50	50	50	50
5	9	9	9	9	9	9
6	3	3	*	*	*	18
7	20	*	*	*	*	
8	–7	–7	–7	*	*	
9	18	18	18	18	18	
Iteración	1ª	2ª	3ª	4ª	..... n	

Sin embargo, si esto mismo se lleva a cabo por medio de un árbol binario, el cual se crea con la información de la base de datos en el momento en que se manda la información a memoria principal, de tal manera que sólo se registran como nodos aquellos registros que no están duplicados y después se regresa nuevamente la información a la base de datos, entonces el número de comparaciones se reduce sustancialmente de tal forma que con la información anterior se tendría el siguiente árbol binario.



	A
1	20
2	3
3	–7
4	9
5	18
6	50

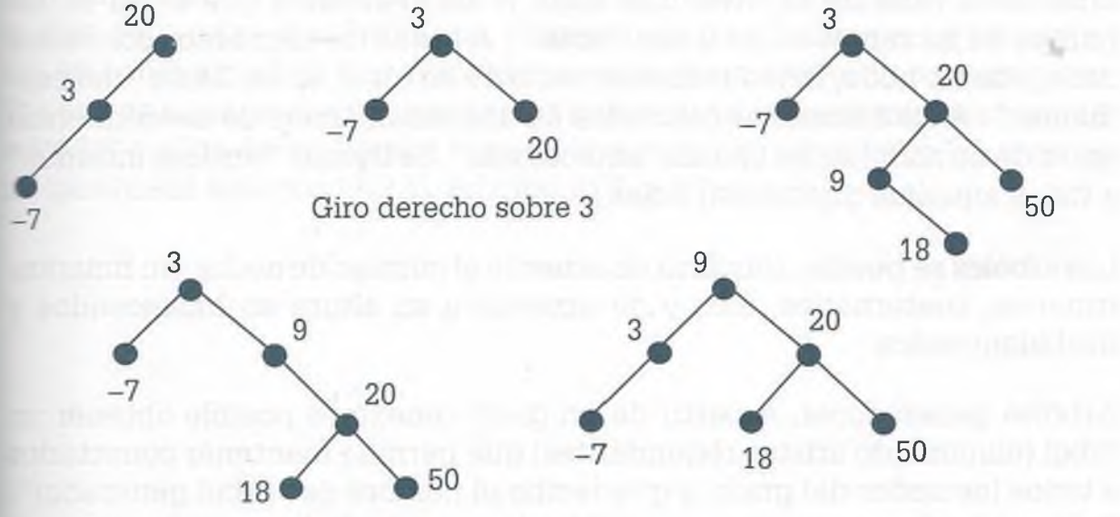
Primero

	A
1	–7
2	3
3	9
4	18
5	20
6	50

Segundo

Si después de crear el árbol se regresa la información a la base de datos en orden *primero*, la información conserva una estructura semejante a la que tenía, como se ve en la figura anterior, pero si se pasa la información del árbol a la base de datos, usando para ello el recorrido en orden *segundo*, entonces no sólo se eliminan los repetidos sino que además se colocan ordenados ascendentemente. El número de comparaciones para crear un árbol binario es menor que n^2, ya que el dato no se tiene que comparar con todos sino solamente con los que sean necesarios para encontrar la colocación del dato en el árbol. De esta manera se reduce significativamente el tiempo requerido para eliminar la información repetida en una base de datos.

Si esta misma operación se realiza usando árboles AVL los resultados son semejantes, ya que un árbol AVL es también un árbol binario que seguramente requerirá de mayor tiempo para su estructuración porque al mismo tiempo que se crea el árbol se balancea cuando es necesario, pero ese tiempo se compensa cuando se manda la información del árbol a la base de datos. El proceso de estructuración del árbol AVL se muestra en la siguiente figura.



Doble giro, uno derecho y luego izquierdo sobre 9

Hay que recordar que para buscar información en un árbol AVL el número de comparaciones en el peor de los casos es log n, sin embargo en un árbol binario que no es AVL el peor de los casos es n (cuando los hijos están alineados en un mismo lado con respecto al padre).

La diferencia entre los árboles B, AVL y binarios es que para usar los árboles B se requiere más trabajo para desarrollar y programar las operaciones de altas, bajas y cambios, pero una vez que ya se tienen programados los algoritmos, la velocidad de respuesta en los árboles B es significativamente menor ya que además de mantenerse balanceados como ocurre con los árboles AVL, la propiedad de tener información de más de un registro

de una base de datos en un mismo nodo permite el manejo de un mayor volumen de información. Al incrementarse el número de registros en un mismo nodo se reduce la altura del árbol, las operaciones para balancear el árbol decrecen y de esta manera se aumenta la eficiencia de los árboles B. Considérese también que los árboles B se utilizan cuando la memoria principal de la computadora no es lo suficientemente grande como para manejar toda la información de la base de datos, sino que se aprovechan las características de los discos rígidos (memoria secundaria) de tal manera que un nodo ocupa un bloque de disco con espacio, para n registros de la base de datos, procurando que la información quede en pistas o cilindros cercanos para evitar la pérdida de tiempo en la lectura y grabación de la información.

8.10 Resumen

Un árbol es un grafo conexo que no tiene ciclos, ni lazos, ni lados paralelos, además de que está compuesto por niveles y al más alto de la jerarquía se le llama "raíz". La raíz tiene un nivel 0, los vértices inmediatamente debajo de la raíz tienen un nivel 1 y así sucesivamente. La altura o peso de un árbol es el valor de su nivel más bajo. A los elementos que están en las puntas de las ramas se les llama "hojas". A todos los elementos colocados debajo de un nodo, independientemente de su nivel, se les llama "descendientes". A los elementos colocados en una misma línea de descendencia antes de un nodo, se les llaman "antecesores". Se llaman "vértices internos" a todos aquellos que no son hojas.

Los árboles se pueden clasificar de acuerdo al número de nodos en: binarios, trinarios, cuaternarios, etc., y de acuerdo a su altura en balanceados y desbalanceados.

Árboles generadores. A partir de un grafo conexo es posible obtener un árbol (eliminando aristas redundantes) que permite mantener conectados a todos los nodos del grafo, y que recibe el nombre de "árbol generador". Existen dos formas en que es posible obtener el árbol generador: usando búsqueda en profundidad, en donde se busca por ramas de arriba hacia abajo y de izquierda a derecha, o bien por medio de búsqueda a lo ancho en donde la búsqueda se hace por niveles.

Un árbol generador mínimo de un grafo es aquel que permite mantener unidos a todos los vértices y que no tiene ciclos, pero que además es la forma más barata o corta ya que la trayectoria o costo es mínimo. Existen varios campos en donde es conveniente mantener funcionando los sistemas al menor costo y algunos de ellos son las redes telefónicas, eléctricas, carreteras, de alcantarillado, etc., en donde es conveniente mantener la comunicación entre los nodos pero sin aristas redundantes o demasiado costosas. Para obtener el árbol generador mínimo en un grafo conexo con pesos es posible aplicar el método de Prim o bien el de Kruskal.

Existen tres maneras de recorrer la información de un árbol y el nombre del recorrido indica el orden en que se coloca el padre en relación a sus hijos. Los recorridos son:

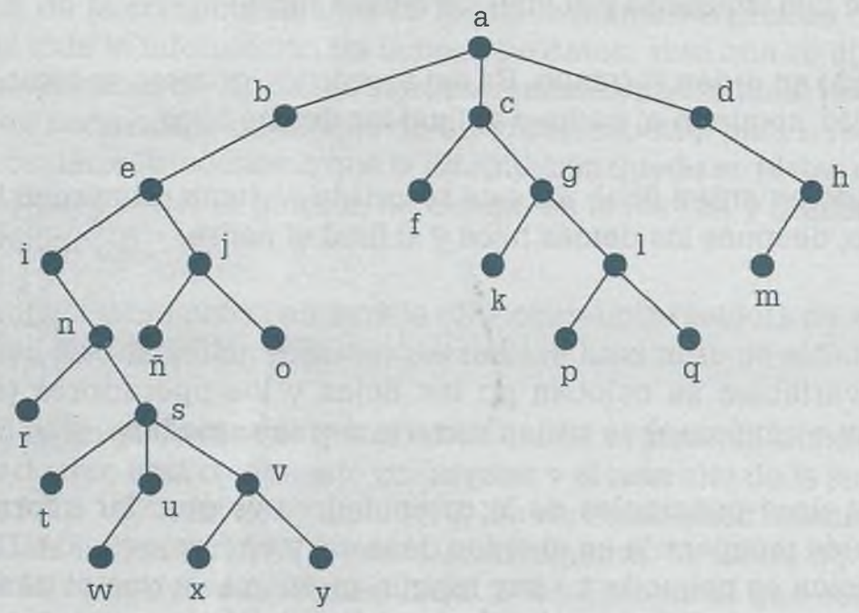
- Recorrido en orden primero. En este recorrido primero se toma el padre, luego el hijo izquierdo y al final los demás hijos.
- Recorrido en orden segundo. En este recorrido primero se toma el hijo izquierdo, segundo el padre y al final los demás hijos.
- Recorrido en orden final. En este recorrido se toma primero el hijo izquierdo, después los demás hijos y al final el padre.

Recorridos en árboles etiquetados. En el área de la computación los árboles etiquetados se usan para evaluar expresiones matemáticas. Las constantes o variables se colocan en las hojas y los operadores (signos aritméticos o funciones) se sitúan como nodos intermedios.

Uno de los usos principales de la computadora es guardar información para después recuperarla en el orden deseado y en forma rápida. Cuando la información es pequeña no hay ningún problema ya que el tiempo en el que encuentra la información almacenada es relativamente pequeño, sin embargo a medida que crece el tiempo de respuesta es importantísimo. Por tal razón es necesario guardar los datos de tal manera que sea posible acceder a ellos en un tiempo razonable y para ello se utilizan los árboles de búsqueda binarios (ABB), árboles AVL y árboles B.

8.11 Problemas

8.1 Considérese el siguiente árbol.

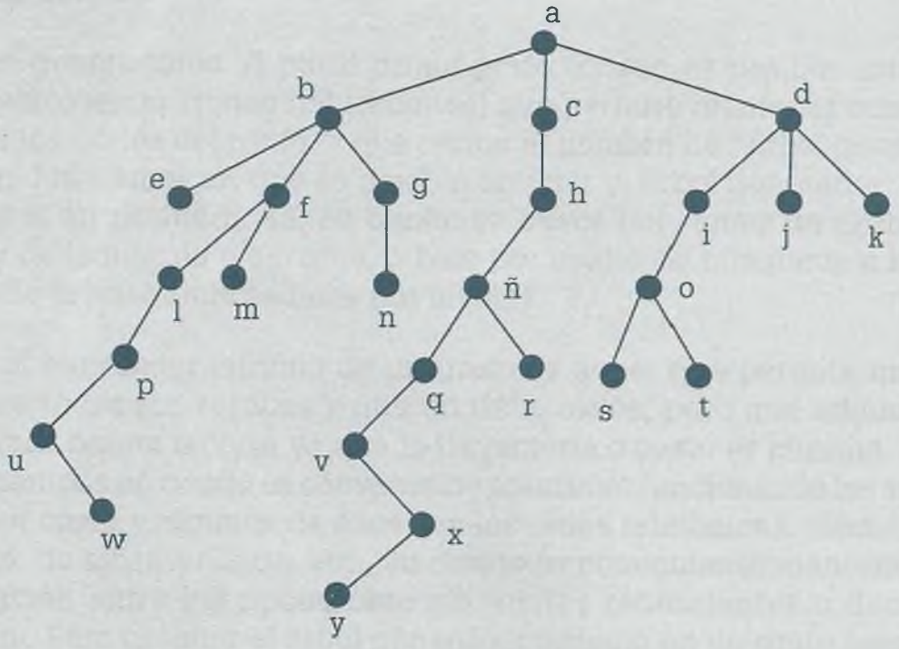


a) ¿Cuál es el recorrido en orden primero, segundo y final?

b) Balancear el árbol a trinario.

c) ¿Cuál es el recorrido en orden primero, segundo y final, ahora que está balanceado?

8.2 Considérese el siguiente árbol.



a) ¿Cuál es el recorrido en orden primero, segundo y final?

b) Balancear el árbol a binario.

c) ¿Cuál es el recorrido en orden primero, segundo y final, ahora que está balanceado?

8.3 Sea la siguiente expresión $\dfrac{(a + b) - bc}{\dfrac{a}{c} + \dfrac{dc}{b}}$. Determinar

a) El árbol binario que representa dicha expresión.

b) El recorrido en orden:

- Primero.
- Segundo.
- Final.

c) Si a = 6, b = 2, c = 3 y d = 4, ¿cuál es la forma en que se lleva a cabo la evaluación en orden primero y final?

d) Si los valores son los que se indican en el inciso (c), por medio de un árbol ilustrar la evaluación en orden segundo.

8.4 Considérense la expresión de cada uno de los incisos:

a) $\dfrac{\dfrac{a}{b - c} + dc}{c - \dfrac{d + a}{b}}$

b) $\dfrac{\dfrac{(a - bc)d}{e}}{\dfrac{(b + a)ce}{b} + f}$

c) $\dfrac{c + d}{a} - \dfrac{ca - bc}{\dfrac{e}{c + d}}$

d) $\dfrac{\dfrac{(e + d)a + ca - e}{b + f}}{\dfrac{a + b}{c} - \dfrac{adf}{e}}$

Determinar:

- El árbol binario que representa la expresión.
- El recorrido en orden primero, segundo y final.
- Si a = 8, b = 4, c = 2, d = 1, e = –2 y f = 6, ¿cuál es la forma en que se lleva a cabo la evaluación en orden primero y final en cada uno de los incisos?
- Si los valores son los que se indican anteriormente, por medio de un árbol ilustrar la evaluación en orden segundo.

8.5 Sea el recorrido en orden primero. +a––*bcd/e+fg.

a) Construir el árbol binario.
b) ¿Cuál es el recorrido en orden: segundo y final?
c) ¿Cuál es la simulación de la evaluación, usando pilas para el recorrido primero y final?
d) ¿Cuál es la evaluación en orden segundo, ilustrándolo por medio de árboles? Con los valores donde sea necesario de a = 1, b = 2, c = 3, d = 5, e = 4, f = –3, g = 5.
e) ¿Cuál es la ecuación matemática que representa el árbol binario?

8.6 Sea el recorrido en orden:

a)	Final:	db*b–cab*/+a*d–
b)	Final:	ab+cd/ef–gh+*–*
c)	Segundo:	a+b–e/c–e+a/b*d
d)	Segundo:	a*d/f/g*h+e+b–c
e)	Primero:	/a+*+bc–ec/d–ba
f)	Primero:	*+/c*+abfde

En cada uno de los incisos realizar lo siguiente:

- Construir el árbol binario.
- Determinar el recorrido en orden segundo y final.
- Simular la evaluación usando pilas para el recorrido primero y final. Con los valores donde sea necesario de a = 3, b = 2, c = –1, d = 1, e = 4, f = 6, g = –3, h = 4.
- Ilustrar la evaluación en orden segundo, usando árboles.
- Determinar la ecuación que representa el árbol binario.

8.7 Con la siguiente información: (m, d, c, e, a, f, g, c, b, m, z, a):

a) Elaborar un árbol binario de búsqueda. Considerar que la información se coloca a la izquierda de un nodo determinado si es menor o igual al nodo y a la derecha exclusivamente si es mayor.

b) ¿Cuál es el recorrido en orden primero, segundo y final?

c) ¿Cómo queda el árbol si se dan de baja los nodos m y c, dando prioridad al nodo izquierdo sobre el derecho para ocupar el lugar del padre? Además dar de alta los nodos n, i y h.

d) ¿Cuál es el recorrido en orden segundo una vez que se han llevado a cabo los ajustes correspondientes?

8.8 Considérese la información siguiente:

a) 40, 12, –8, 0, 60, 35, 5, –1, 7, 23, 42, 70, 38

b) 0, 7, 17, –7, 4, –2, 60, 11, -5, 0, 8

c) 100, 3, 7, 5, –2, 60, 80, 115, –1, 0

d) 48, 32, 12, 27, 72, 85, 0, 1, 60, 70, 56, 49, 48

En relación con cada uno de los incisos:

- Elaborar un árbol binario de búsqueda. Considerar que la información se coloca a la izquierda de un nodo determinado si es menor o igual al nodo y a la derecha exclusivamente si es mayor.
- ¿Cuál es el recorrido en orden primero, segundo y final?
- ¿Cómo queda el árbol, si se hacen los siguientes movimientos en cada uno de los incisos? Concediendo prioridad al nodo izquierdo sobre el derecho para ocupar el lugar del padre cuando existe alguna baja.

a) Baja: 12; altas: 58, 3 y 6.

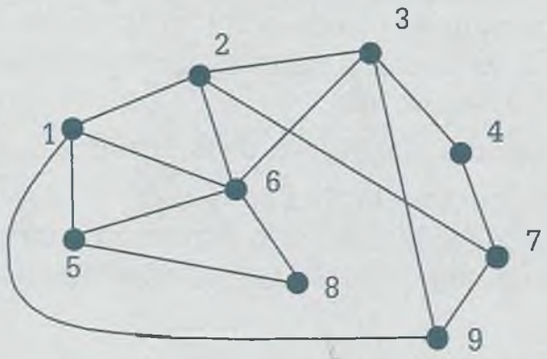
b) Bajas: 17 y –7; altas: 0, –2 y 18.

c) Bajas: –2 y 80; altas: 9 y 72.

d) Bajas: 48 y 12; altas: –9, 64 y 40.

- ¿Cuál es el recorrido en orden primero, después de haber hecho los ajustes?

8.9 Considérese el siguiente grafo:



Determinar el árbol generador, partiendo del nodo 1 y prioridad en orden ascendente. Por medio de búsqueda:

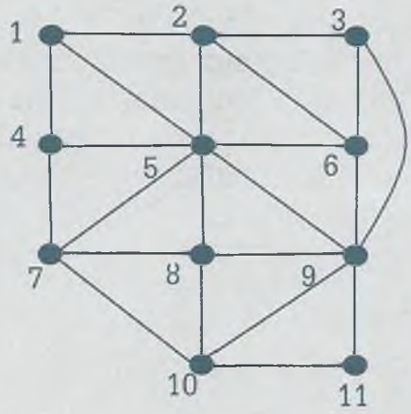
a) A lo ancho.

b) En profundidad.

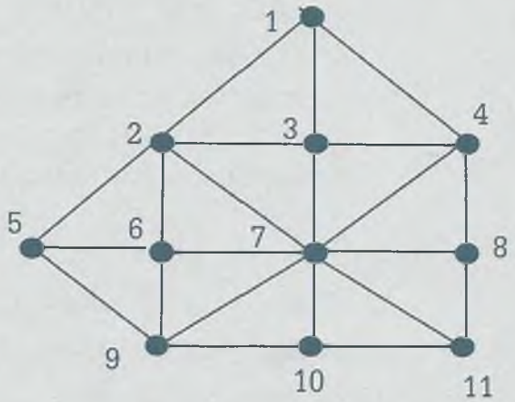
8.10 Determinar el árbol generador de los grafos de cada inciso, partiendo del nodo 1 y con prioridad de selección ascendente, usando para ello búsqueda.

- A lo ancho.
- En profundidad.

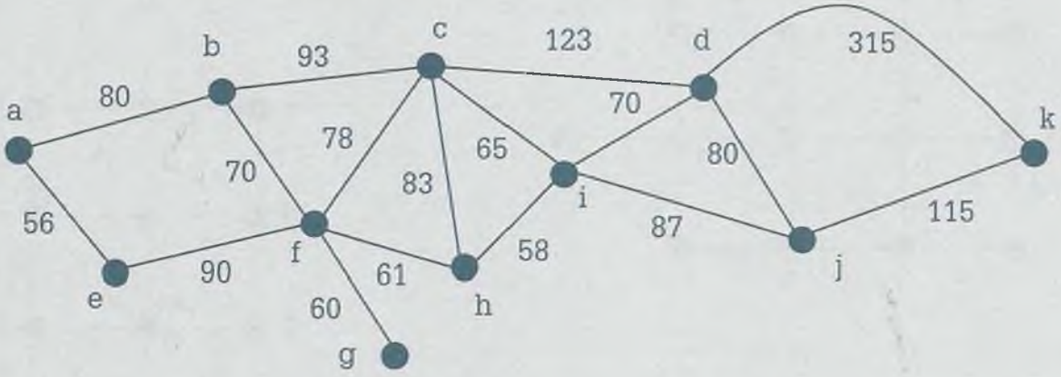
a)



b)



8.11 El siguiente grafo representa la red carretera entre las ciudades a, b, c, d, e, f, g, h, i, j y k, así como la longitud de cada una de las carreteras que unen las distintas ciudades:



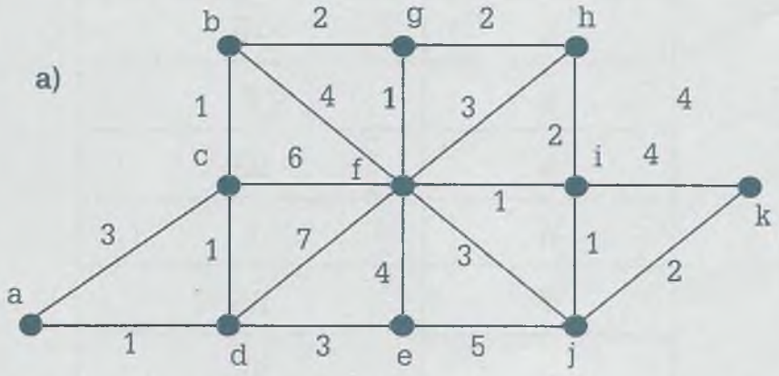
Determinar el árbol generador mínimo usando el método de:

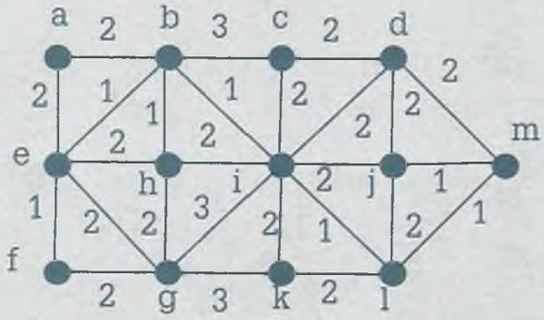
a) Prim.

b) Kruskal.

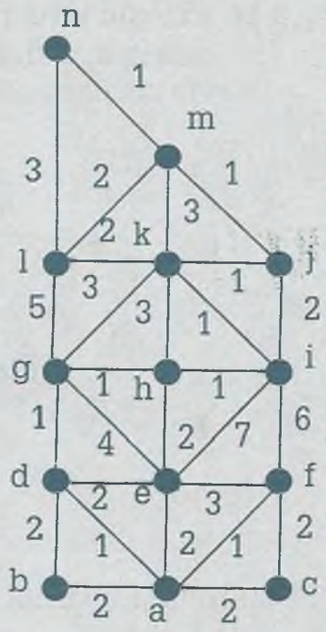
8.12 Determinar el árbol generador mínimo de los grafos de los incisos (a), (b) y (c) usando para ello los métodos de Prim y Kruskal. Si existen más árboles de expansión mínimos muestre adicionalmente uno para cada grafo.

a)





b)



c)

8.13 En un texto se encontró que la frecuencia de uso de cada uno de los caracteres es la siguiente.

Carácter	Peso o frecuencia
o	20
l	8
u	15
s	6
a	23
n	6
i	25
b	10
e	19

a) ¿Cuál es el árbol optimal para el código de Huffman?.

b) Con el árbol obtenido, codificar el mensaje: **solounabuenailusion**.

8.14 En un documento se encontró que la frecuencia con la que ocurren los caracteres es:

Carácter	Peso o frecuencia
a	80
e	72
f	55
g	28
h	19
i	78
l	33
m	43
o	69
s	36
u	48
espacio	47

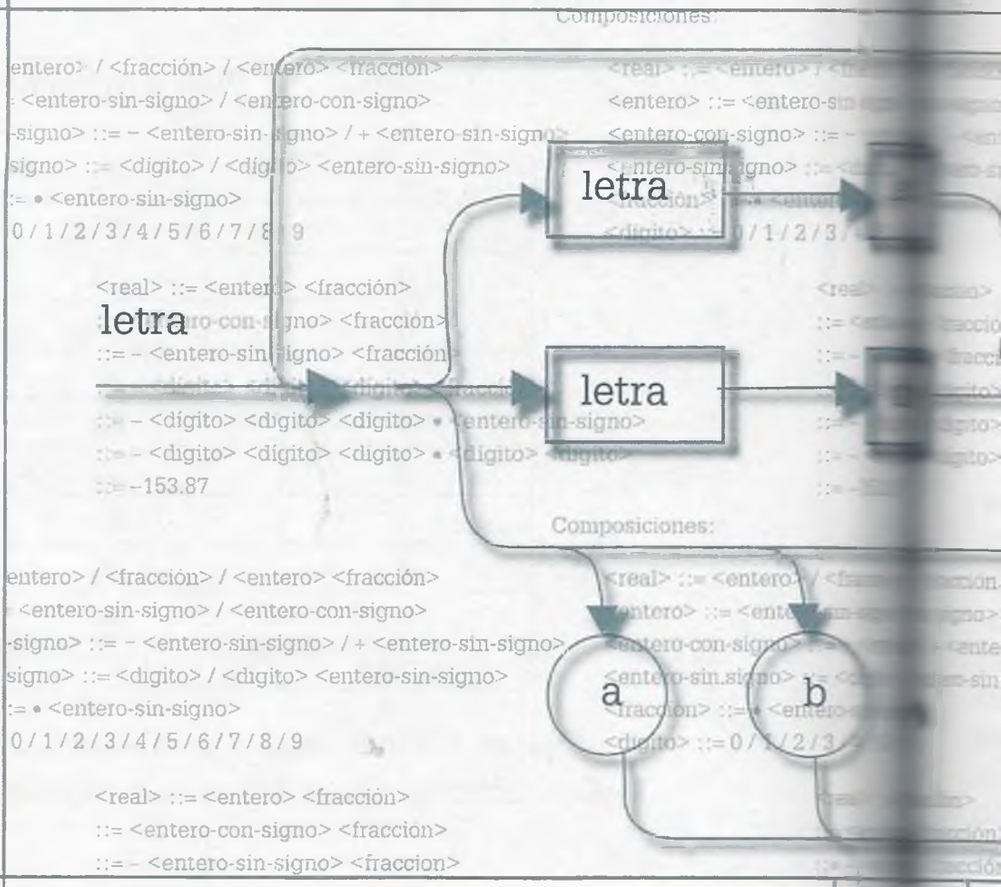
a) ¿Cuál es el árbol optimal para el código de Huffman?

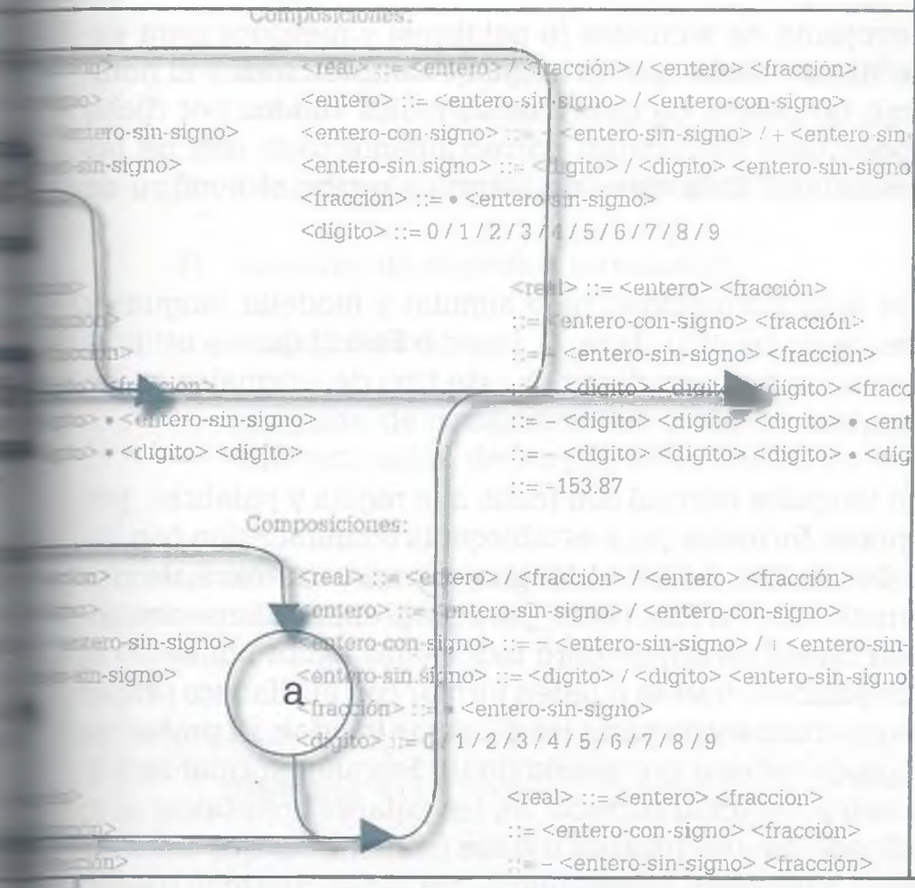
b) Con dicho árbol codificar el mensaje: **hola amigos**.

CAPÍTULO

IX

Introducción a los lenguajes formales





Las palabras o el lenguaje, ya sea escrito o hablado, no parecen desempeñar ningún papel en mi mecanismo de pensamiento. Las entidades físicas que parecen servir como elementos del pensamiento son ciertos signos e imágenes más o menos claros que pueden reproducirse y combinarse voluntariamente.

Albert Einstein

Objetivos

- Identificar las gramáticas regulares, libres de contexto y sensibles al contexto, así como la importancia que tienen en el área de la computación.
- Representar las gramáticas regulares usando autómatas finitos y máquinas de estado finito.
- Convertir un autómata finito no determinístico (AFN) en un autómata finito determinístico (AFD) equivalente.
- Representar las gramáticas libres de contexto por medio de árboles de derivación, diagramas sintácticos y diagramas Backus-Naur (BNF).
- Comprender el funcionamiento de las máquinas de Turing en el manejo de cadenas.
- Comprender la teoría de la computabilidad y de la complejidad, así como la utilidad en el área de la computación.

9.1 Introducción

Un lenguaje es un conjunto de símbolos (o palabras) y métodos para estructurar y combinar dichos símbolos. Un lenguaje también recibe el nombre de idioma y como tal consta de todos los símbolos válidos por dicho lenguaje y los métodos para estructurar correctamente cada una de las palabras, frases y oraciones. Esta clase de lenguaje recibe el nombre de lenguaje natural.

Existen lenguajes de menor capacidad para simular y modelar lenguajes naturales, como el lenguaje binario, Java, C, Basic o Pascal que se utilizan en la comunicación con las computadoras. A este tipo de lenguajes se les llama lenguajes formales.

Es difícil modelar un lenguaje natural con todas sus reglas y palabras, por esto se utilizan lenguajes formales para establecer la comunicación con las computadoras y sus periféricos. Se tienen lenguajes para controlar sistemas de producción automatizados en empresas, para programar robots, controlar aviones o manejar bases de datos, pero todos ellos están limitados al conjunto reducido de palabras que se pueden formar con el alfabeto propio del lenguaje y las reglas para estructurar las palabras válidas. El problema para simular un lenguaje natural por medio de un lenguaje formal radica no solamente en la estructuración correcta de las palabras (sintaxis) sino también en el significado de una palabra o frase (semántica) que muchas veces varía de persona a persona, dependiendo del lugar, contexto o estado de ánimo.

9.2 Gramáticas y lenguajes formales

Lenguaje L(G). Este tipo de lenguaje se basa en la gramática, así como en las reglas o métodos para la creación de palabras propias del lenguaje.

Un lenguaje L(G) consiste en una gramática (G) con todos los arreglos que se pueden obtener a partir del estado inicial (s) y las composiciones (c).

9.2.1 Estructuración de las gramáticas

Las gramáticas están integradas por varios elementos que permiten la estructuración de palabras. La gramática

$$G = \{\Sigma, N, T, s, c\}$$

es el sistema que permite establecer las reglas que han de aplicarse a un lenguaje y aquí se tiene que:

- Σ: es el alfabeto o conjunto de símbolos con el cual se forman palabras de un lenguaje.
- *N:* conjunto de símbolos no terminales en un lenguaje.
- *T:* conjunto de símbolos terminales.
- *s:* estado inicial.
- *c:* conjunto de composiciones o reglas que se deben usar para la estructuración de las palabras válidas en el lenguaje.

Un símbolo es cualquier carácter o figura, por ejemplo: a, b, p, 3, 5, #, &, @, etcétera.

Ejemplo 9.1. Sea

$$\Sigma = \{a, b, c, d, \ldots, z\}$$
$$L(G) = \{hola, conejo, pera, piña, \ldots,\}$$

Con los símbolos del alfabeto es posible estructurar palabras de un lenguaje como *hola, conejo, pera,* siguiendo ciertas reglas para su correcta estructuración.

En una gramática los símbolos terminales (*T*) se indican por medio de números o letras minúsculas y los símbolos no terminales (*N*) por letras mayúsculas o la letra *s* para indicar que se trata del símbolo inicial.

Las palabras válidas en un lenguaje dependen del alfabeto y de las composiciones (*c*) propias de la gramática. Las composiciones de un lenguaje están integradas por símbolos terminales y no terminales.

Ejemplo 9.2. Sea

$$\Sigma = \{a, g, h, i, l, m, o, r\}$$

cuyas composiciones (c) son

$s \rightarrow hA$	$B \rightarrow rD$	$F \rightarrow gC$
$A \rightarrow oB$	$D \rightarrow mE$	$C \rightarrow a$
$B \rightarrow lC$	$E \rightarrow iF$	

Comenzando en el estado inicial (s) es posible formar palabras propias del lenguaje como

$$s \rightarrow hA \rightarrow hoB \rightarrow holC \rightarrow hola$$
$$s \rightarrow hA \rightarrow hoB \rightarrow horD \rightarrow hormE \rightarrow hormiC \rightarrow hormiga$$

Hay que observar cómo se pueden sustituir los símbolos no terminales (s, A, B, C, D, E, F) por su equivalente para la estructuración de las palabras de un lenguaje, de forma que el lenguaje conste de todas aquellas palabras que se puedan estructurar partiendo de la gramática:

$$L(G) = \{hola, hormiga, \ldots\}$$

Si alguna otra palabra, además de *hola* y *hormiga*, se puede derivar con la gramática anterior, esta palabra también será parte del lenguaje L(G).

Para que una palabra se considere parte de un lenguaje deberá estar formada solamente por símbolos terminales, ya que los símbolos no terminales forzosamente llevarán a integrar nuevos símbolos terminales y símbolos no terminales. Siempre se deberá comenzar en el símbolo no terminal (s) para la formación de palabras de un lenguaje.

9.2.2 Clasificación de las gramáticas

Las gramáticas se pueden clasificar como:

Tipo 0: si no se pone ninguna restricción a las composiciones de G.

Tipo 1: si para cualquier composición $d_1 \rightarrow d_2$ de la gramática G, la longitud de símbolos de la izquierda de la composición (d_1) es menor o igual a la longitud de símbolos de la derecha (d_2).

Tipo 2: si el lado izquierdo de cada composición es un símbolo no terminal y el lado derecho consta de uno o más símbolos terminales y/o no terminales.

Tipo 3: si el lado izquierdo de la composición es un símbolo no terminal y el lado derecho tiene uno o más símbolos, incluyendo a lo más un símbolo no terminal.

A las gramáticas 0 y 1 también se les conoce como ***Sensibles al contexto***. Este tipo de gramáticas son muy complicadas y, por lo tanto, difíciles de analizar y estudiar. En general, son muy pocos los conocimientos que se tienen de ellas debido principalmente a la libertad que se tiene para formar palabras de un lenguaje.

La gramática tipo 2 recibe el nombre de gramática ***Libre de contexto***. Este tipo de gramáticas es el que se usa actualmente para la creación de lenguajes formales, además de que tiene relación con los autómatas finitos, autómatas de pila y máquinas de Turing.

La gramática tipo 3 o también llamada ***Regular,*** tiene reglas muy simples de sustitución y generación de palabras de un lenguaje. En esta gramática se sustituye un símbolo no terminal por uno terminal o bien por uno terminal seguido de uno no terminal. La gramática regular tiene una gran relación con los autómatas finitos.

Hay que observar que una gramática regular es, a su vez, una gramática libre de contexto y también una gramática sensible al contexto, de la misma forma que una gramática libre de contexto es una gramática sensible al contexto.

Ejemplo 9.3. Considérese la siguiente gramática G en donde:

$T = \{a, b\}$ Conjunto de símbolos terminales.

$N = \{s, \text{A}, \text{B}\}$ Conjunto de símbolos no terminales, donde s es el símbolo inicial.

Composiciones:

$s \rightarrow a\text{A}$ $\quad$ $\text{A} \rightarrow aa\text{B}$ $\quad$ $\text{AB} \rightarrow s\text{B}$

$\text{A}b\text{B} \rightarrow a\text{A}$ $\quad$ $\text{B} \rightarrow b$ $\quad$ $\text{A}a \rightarrow a$

Esta gramática es sensible al contexto, ya que no presenta ninguna restricción a las composiciones. Una característica es que el número de símbolos del lado izquierdo de la composición $\text{A}b\text{B} \rightarrow a\text{A}$ es mayor que el número de símbolos del lado derecho, cosa que no es permitido en ninguna de las demás gramáticas.

Ejemplo 9.4. Considérese la siguiente gramática G en donde:

$T = \{a, b\}$

$N = \{s, \text{A}, \text{B}\}$

Composiciones:

$s \rightarrow a\text{B}$ $\quad$ $\text{A} \rightarrow b\text{A}$ $\quad$ $\text{B} \rightarrow ab\text{A}$

$\text{A} \rightarrow bs$ $\quad$ $\text{A} \rightarrow b$ $\quad$ $\text{B} \rightarrow a$

En este caso se trata de una gramática regular, ya que en el lado izquierdo de las composiciones hay un solo símbolo no terminal y del lado derecho hay uno o más símbolos terminales (a, b) y a lo más un símbolo no terminal (s, A, B). Se sabe que una gramática regular también es libre de contexto y sensible al contexto, porque reúne los requisitos para ello.

9.2.3 Representación de las gramáticas

Como ya se vio, existen gramáticas regulares, libres de contexto y sensibles al contexto, y cada una de ellas tiene su propia forma de representación.

Gramáticas regulares. Se representan principalmente por medio de autómatas finitos los cuales son elementos generadores de palabras de un lenguaje, en donde una palabra se genera comenzando en el símbolo inicial y terminando como una cadena de símbolos terminales que no contiene ningún símbolo no terminal.

Se puede decir que la gramática de un lenguaje regular es semejante a un autómata finito, lo que se debe de hacer es representar en forma gráfica cada una de las composiciones de la gramática teniendo en cuenta que los símbolos no terminales se encierran en círculos y los símbolos terminales son las etiquetas de las aristas del autómata finito.

Ejemplo 9.5. Sea la gramática G, en donde:

$T = \{a, b\}$

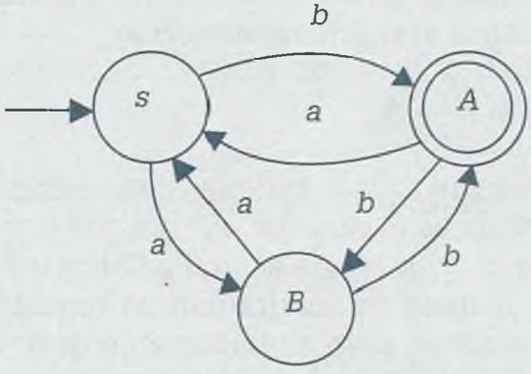
$N = \{s, A, B\}$

Composiciones:

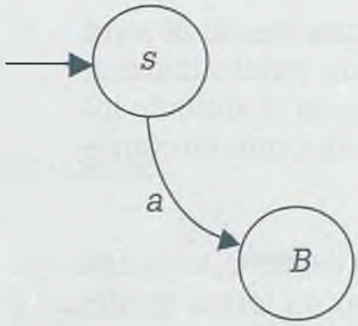
$s \rightarrow aB$ $\quad A \rightarrow bB$ $\quad B \rightarrow bA$ $\quad B \rightarrow b$

$s \rightarrow bA$ $\quad B \rightarrow as$ $\quad A \rightarrow as$ $\quad s \rightarrow b$

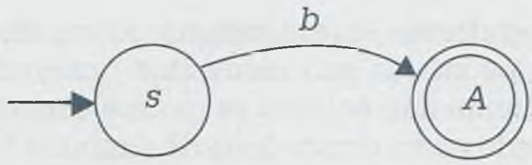
El autómata finito que representa esta gramática es:



Hay que observar cómo cada una de las transiciones o composiciones está representada en el autómata finito por medio de una arista o flecha con la etiqueta respectiva.



$s \rightarrow aB$



$s \rightarrow bA$

La flecha que incide sobre el círculo que contiene a *s* indica que se trata del símbolo inicial. El símbolo no terminal que tiene doble círculo y que permite saber si la palabra pertenece o no al lenguaje se obtiene con las transiciones B $\rightarrow$ *b* y s $\rightarrow$ *b*, ya que ambas inciden sobre el símbolo no terminal "A", además de que estas transiciones ya no contienen símbolos no terminales en el segundo término, de tal manera que al llevar a cabo un recorrido con una palabra y terminar en el símbolo de aceptación, esto significa que dicha palabra pertenece al lenguaje. Si el recorrido de una palabra a través del autómata finito no termina en el símbolo de aceptación, entonces dicha palabra no forma parte del lenguaje.

Por ejemplo, para determinar si la palabra bbaab pertenece al lenguaje L(G) se pueden usar las composiciones de la siguiente manera:

$$s \rightarrow b\text{A} \rightarrow bb\text{B} \rightarrow bbas \rightarrow bbaa\text{B} \rightarrow bbaab$$

Usando el autómata finito se puede observar que a partir del símbolo inicial *s* la cadena *bbaab* realiza el siguiente recorrido:

$$s \rightarrow \text{A} \rightarrow \text{B} \rightarrow s \rightarrow \text{B} \rightarrow \text{A}$$

Después de que se recorre toda la cadena, si se termina en un símbolo no terminal aceptado, como es el caso de "A" (ya que tiene doble círculo), se dice que la palabra o cadena *bbaab* es una palabra del lenguaje L(G). Los lenguajes regulares propios de las gramáticas regulares se tratarán con más detalle posteriormente, pero es importante dejar claro que los autómatas finitos son su mejor forma de representación.

Gramáticas libres de contexto. Este tipo de gramáticas son las más usadas en la elaboración de compiladores, traductores e intérpretes, y se pueden representar por medio de árboles de derivación, representación BNF y diagramas sintácticos.

Representación mediante árboles de derivación. El procedimiento para determinar si una palabra pertenece a un lenguaje por el método de árboles de derivación es semejante al desarrollado por medio de composiciones, sólo que en este caso se estructura un árbol teniendo como raíz de ese árbol al símbolo inicial *s* y colocando como hijos a los signos del lado derecho de la composición. De esta manera se representan cada una de las composiciones de la gramática en el árbol, sustituyendo el padre cuando se trata de un símbolo no terminal por los símbolos del lado derecho de cada una de las composiciones y respetando el orden en que se encuentran colocados cada uno de los símbolos de las composiciones. El árbol se termina de estructurar cuando todas sus hojas son símbolos terminales y la palabra formada por los símbolos terminales de izquierda a derecha es una palabra que pertenece al lenguaje L(G).

Ejemplo 9.6. Sea la gramática libre de contexto G:

$T = \{a, b\}$

$N = \{s, \mathrm{A}, \mathrm{B}\}$

Composiciones:

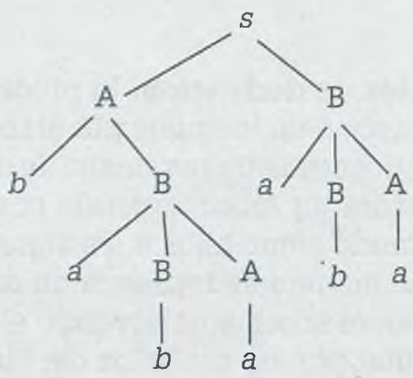
$s \rightarrow \mathrm{AB} \quad \mathrm{A} \rightarrow a \quad \mathrm{B} \rightarrow b$

$\mathrm{A} \rightarrow \mathrm{bB} \quad \mathrm{B} \rightarrow a\mathrm{BA}$

Usando las composiciones se puede derivar la cadena *babaaba* de la siguiente manera:

$s \rightarrow \mathrm{AB} \rightarrow b\mathrm{BB} \rightarrow ba\mathrm{BAB} \rightarrow baba\mathrm{B} \rightarrow babaa\mathrm{BA} \rightarrow babaaba$

O bien por medio del árbol de derivación:



Debido a que la palabra que se forma con las hojas del árbol de izquierda a derecha es *babaaba* se dice que esta palabra forma parte del lenguaje L(G). Es obvio que existen muchas palabras que se pueden obtener con esta gramática y todas ellas forman parte del lenguaje.

La diferencia principal entre los autómatas finitos y los árboles de derivación es que con el mismo autómata finito es posible determinar si una o varias palabras pertenecen a un lenguaje, y en el caso de árboles de derivación cada palabra requiere de la estructuración de un árbol de derivación diferente.

Representación BNF (Backus-Naur). Se mencionó con anterioridad que la gramática libre de contexto se utiliza con frecuencia para la representación de lenguajes formales como C, Pascal, Basic, etc. La representación BNF es un buen ejemplo de ello.

En la gramática BNF la flecha (→) de una composición se indica con "::=". Por ejemplo, la composición $s \rightarrow AB$, se representa de la siguiente manera:

$$s ::= AB$$

Cuando un mismo símbolo no terminal implica diferentes cadenas de símbolos es posible compactar la información usando para ello el carácter "/", que tiene el significado de *o*. Las siguientes composiciones $s \rightarrow AB$, $s \rightarrow abB$, $s \rightarrow bAa$, $s \rightarrow b$ equivalen en representación BNF a

$$s ::= AB \ / \ abB \ / \ bAa \ / \ b$$

Ejemplo 9.7. Sea la gramática libre de contexto G:

$T = \{x, y\}$

$N = \{s, A, B, C\}$

Composiciones:

$s \rightarrow xB$ $\quad$ $B \rightarrow yxC$ $\quad$ $C \rightarrow BxA$

$A \rightarrow xBy$ $\quad$ $B \rightarrow yyC$ $\quad$ $C \rightarrow xy$

$A \rightarrow CBx$ $\quad$ $B \rightarrow x$ $\quad$ $A \rightarrow y$

La representación de la gramática por medio de BNF es:

$s ::= xB$ $\quad$ $B ::= yxC \mid yyC \mid x$

$A ::= xBy \mid CBx \mid y$ $\quad$ $C ::= BxA \mid xy$

Algunas veces en la representación BNF los símbolos no terminales empiezan con "<", terminan con ">" y la flecha ($\rightarrow$) de una composición se indica con "::=". Por ejemplo, la composición $s \rightarrow AB$ se representa de la siguiente manera:

<s> ::= <A> <B>

Composiciones de la forma $s \rightarrow AB$, $s \rightarrow abB$, $s \rightarrow bAa$, $s \rightarrow b$ son equivalentes a

$s ::= <A> <B> \mid ab <B> \mid b <A> a \mid b$

Ejemplo 9.8. La gramática libre de contexto G en representación BNF para leer un número real en Pascal es:

$T = \{., -, +, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9\}$

$N =$ {real, entero, fracción, entero-sin-signo, entero-con-signo, dígito}

Composiciones:

<real> ::= <entero> / <fracción> / <entero> <fracción>
<entero> ::= <entero-sin-signo> / <entero-con-signo>
<entero-con-signo> ::= − <entero-sin-signo> / + <entero-sin-signo>
<entero-sin.signo> ::= <dígito> / <dígito> <entero-sin-signo>
<fracción> ::= • <entero-sin-signo>
<dígito> ::= 0 / 1 / 2 / 3 / 4 / 5 / 6 / 7 / 8 / 9

Por lo tanto, la derivación para determinar si la palabra –153.87 pertenece al lenguaje es como sigue:

<real> ::= <entero> <fracción>
::= <entero-con-signo> <fracción>
::= – <entero-sin-signo> <fracción>
::= – <dígito> <dígito> <dígito> <fracción>
::= – <dígito> <dígito> <dígito> • <entero-sin-signo>
::= – <dígito> <dígito> <dígito> • <dígito> <dígito>
::= –153.87

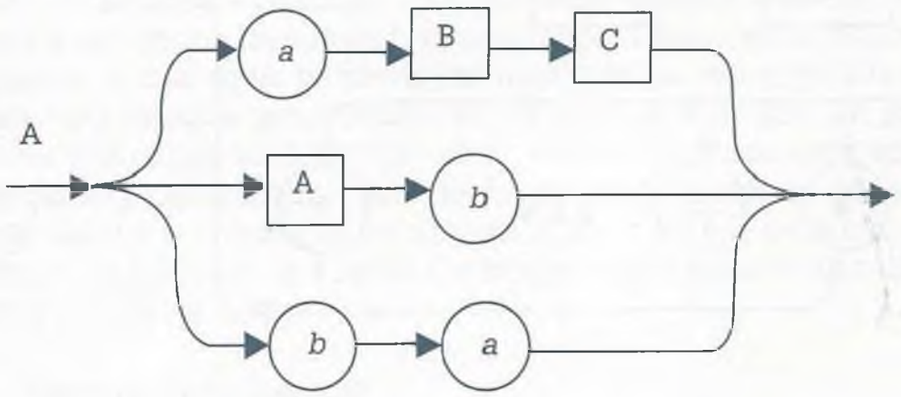
Con lo cual se concluye que la palabra –153.87 pertenece al lenguaje L(G).

Cuando se compila un programa, lo que se hace es determinar si cada una de las líneas del programa forma parte del lenguaje. Es obvio que cada una de las líneas se somete a la verificación de la gramática correspondiente, y en caso de que alguna instrucción no esté bien escrita el compilador mandará el mensaje correspondiente para que el programador lleve a cabo la corrección respectiva.

Diagramas sintácticos. Ésta es una forma gráfica para representar una gramática por medio de gráficas dinámicas que permiten determinar en forma más ilustrativa si una palabra pertenece a un lenguaje; a esta gráfica se le conoce como *diagrama sintáctico* o *diagrama de sintaxis*. Toda palabra reservada de un lenguaje formal es susceptible de ser representada por medio de diagramas sintácticos.

En un diagrama sintáctico los símbolos no terminales se representan dentro de un cuadro, y los símbolos terminales se encierran dentro de un círculo.

El diagrama sintáctico que representa a la composición A ::= *a*BC / A*b* / *ba* es el siguiente:



Un identificador de un lenguaje cualquiera está integrado por una sola "letra" o bien una "letra seguida de una combinación de letras o números", como se muestra en el siguiente ejemplo.

Ejemplo 9.9. Considérese la siguiente gramática:

$T = \{0, 1, 2, \ldots, 9, a, b, \ldots, z\}$

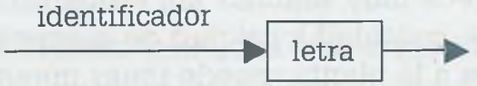
$N = \{$identificador, letra, dígito$\}$

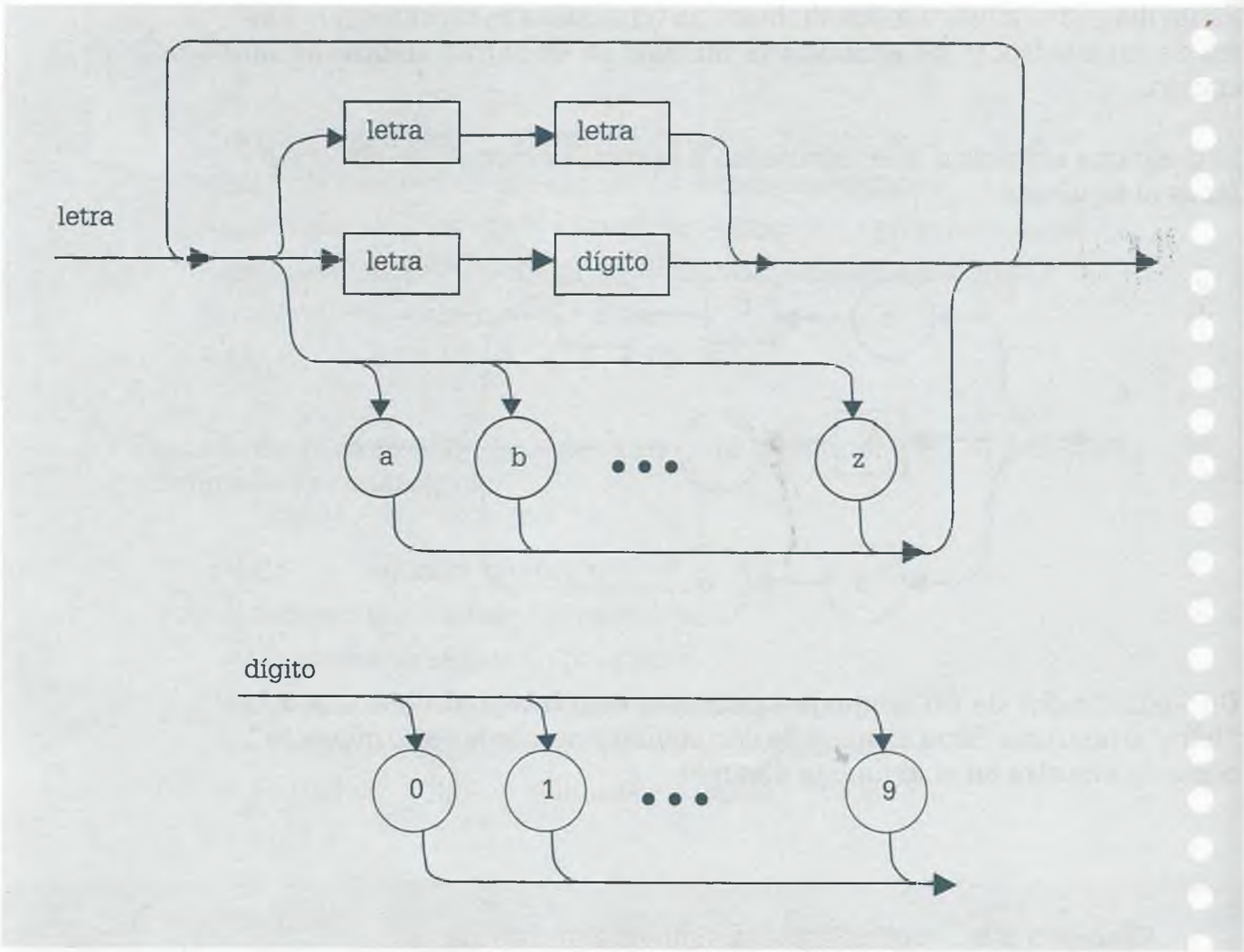
Composiciones:

<identificador> ::= <letra>

<letra> ::= <letra> <letra> / <letra> <dígito> / *a* / *b* / *c* / ... / *z*

<dígito> ::= 0 / 1 / 2 / ... / 9





Además de los árboles de derivación, representación BNF y diagramas sintácticos, las gramáticas libres de contexto se pueden representar por medio de un autómata de pila y máquinas de Turing.

9.3 Autómatas finitos

Todo proceso que recibe una o varias entradas, que las transforma y que después emite una salida recibe el nombre de *sistema*. Existen sistemas muy complejos, como los sistemas de los seres vivos; por ejemplo una planta que recibe como entrada agua, sales minerales, oxígeno y luz solar, procesa esas entradas y emite como salida hojas, tallos, flores y frutos. Este sistema parece muy sencillo, sin embargo no lo es ya que de acuerdo con las entradas, cantidad y calidad de elementos, así como el medio ambiente que rodea a la planta, puede tener mejores flores y frutos.

Un sistema aún más complejo es el comportamiento del ser humano, el cual recibe infinidad de información de varios lados que se procesa de acuerdo

con su estructura cognitiva y que en función de esto toma una decisión de lo que debe hacer. Su comportamiento o reacción a la información proporcionada no se puede determinar con exactitud, ya que depende además de la información de entrada, del entorno socioeconómico, edad, valores morales entre otras muchas cosas más. Este tipo de sistemas recibe el nombre de sistemas infinitos.

Los autómatas finitos también reciben como entrada información que procesan y en función de ello emiten una salida. Un autómata finito recibe una palabra, la cual debe procesar por medio de un recorrido a través de los diferentes estados que integran el autómata, y si al final del procesamiento de ésta el recorrido termina en un estado o posición de aceptación, se dice que la palabra forma parte del lenguaje. A diferencia de los sistemas infinitos, un autómata es un sistema finito y por eso se le llama *autómata finito*, en donde sí es posible determinar con exactitud la salida que se tendrá con cierta información.

9.3.1 Terminología básica

Las gramáticas regulares son parte esencial de los lenguajes regulares y los autómatas finitos (AF) son una representación gráfica de los lenguajes regulares. Es conveniente, antes de abordar el tema de AF, tener en cuenta algunos conceptos importantes que permiten la manipulación correcta de los lenguajes regulares y los autómatas finitos, por esta razón se tratarán algunos aspectos que se consideran importantes antes de entrar de lleno con el tema de autómatas finitos.

Una palabra que pertenece a un lenguaje L(G) realmente es una cadena de símbolos o caracteres, y por esto la relación existente entre símbolos, cadenas, lenguajes, alfabetos y gramáticas es importante.

Cadena.

Ésta consiste en una secuencia de símbolos yuxtapuestos (se coloca uno seguido del otro).

Sean

$$\Sigma = \{0, 1, 2\}$$

$$w = 0 \qquad x = 02 \qquad y = 011 \qquad z = 12012$$

En este caso w, x, y, z son cadenas formadas con símbolos del alfabeto Σ.

Cadena vacía (ε).

Es aquella cadena que no tiene símbolos. Aquí se tiene que $\varepsilon w = w\varepsilon$ y $\varepsilon\varepsilon = \varepsilon$.

Si w = pera, entonces

εw = pera = $w\varepsilon$.

Inversa de una cadena (w^I).

Es la cadena que se obtiene al escribir los caracteres en forma invertida.

Sea w = Hola, entonces

$$w^I = (\text{Hola})^I = (\text{ola})^I\text{H} = (\text{la})^I\text{oH} = (\text{a})^I\text{loH} = (\in)^I\text{aloH} = \text{aloH}$$

Cadena elevada a una potencia (w^n).

Si n es un elemento del conjunto de números naturales ($n \in N$) y w es una cadena, entonces:

$w^0 = \varepsilon$ (cadena vacía)

$w^1 = ww^0$

$w^2 = ww^1 = www^0$

$w^3 = ww^2 = www^1 = wwww^0$

$w^n = ww^{n-1}$

Sea w = Hola, entonces

$w^4 = ww^3 = www^2 = wwww^1 = wwwww^0$ = HolaHolaHolaHola$\in$ = HolaHolaHolaHola

Cerradura de Kleene o cerradura estrella (L^*).

Es el lenguaje con todas las cadenas que se pueden formar con el alfabeto (Σ).

Sea

$\Sigma = \{0, 1\}$

$\Sigma^* = \{0, 1, 00, 01, 10, 11, 000, 100, 101, 110, 111, 0000, 0001, 0010, \ldots\}$

Σ es un conjunto finito, pero Σ^* es infinito ya que el número de cadenas diferentes que se pueden formar es infinita.

Stephen Kleene
(1909-1994)

Fue un lógico y matemático estadounidense nació en Hartford, Connecticut, el de enero de 1909, fue director de los epartamentos de matemáticas y de análisis numérico de la Universidad de Wisonsin. Se especializó en el estudio de las unciones recursivas y la teoría de los autómatas, y entre sus numerosas obras destacan Introducción a la matemática que publicó en 1952 y Lógica matemática en 1967, además de que introdujo la operación clausura de Kleene denotada por el símbolo X*.

Cerradura estrella sobre un lenguaje (L^*).

Es la unión de todos los lenguajes potencia de L desde $n = 0$ hasta infinito ($n \in N$), que se pueden formar con el alfabeto (Σ).

Sea

$\Sigma = \{a, b\}$, $L = \{abb\}$;
$L^* = L^0 \cup L^1 \cup \ldots \cup L^\infty = \{\varepsilon\} \cup \{abb\} \cup \{abbabb\} \cup \ldots \cup \{abb\}^\infty$
$L^* = \{\varepsilon, abb, abbabb, abbabbabb, \ldots\}$

Cerradura positiva de un lenguaje (L^+).

Es la unión de todos los lenguajes potencia de L, desde $n = 1$ hasta infinito, que se pueden formar con el alfabeto (Σ).

Sea

$\Sigma = \{a, b\}$, $L = \{abb\}$;
$L^+ = L^1 \cup L^2 \cup \ldots \cup L^\infty = \{abb\} \cup \{abbabb\} \cup \ldots \cup \{abb\}^\infty$
$L^+ = \{abb, abbabb, abbabbabb, \ldots\}$

Inversa de un lenguaje (L^I).

Es el lenguaje que se obtiene al escribir los elementos de un lenguaje en forma invertida.

Sea

$L = \{\text{Morelia, Michoacán}\}$; $\Sigma = \{a, b, c, \ldots, z\}$
$L^I = \{\text{aileroM, nácaohciM}\}$

Operaciones.

Considerando que un lenguaje es un conjunto de símbolos o palabras sobre un alfabeto, se pueden llevar a cabo algunas operaciones de conjuntos como la unión, intersección, diferencia y complementación.

Sean L y M lenguajes, entonces

$(L \cup M) = \{x \mid x \in L \text{ o } x \in M \text{ o } x \in (L \cap M)\}$
$(L \cap M) = \{x \mid x \in L \text{ y } x \in M\}$
$(L - M) = \{x \mid x \in L \text{ y } x \notin M\}$
$L' = \Sigma^* - L = \{x \mid x \in \Sigma^* \text{ y } x \notin L\}$

Sean

$$\Sigma = \{a, b, c\}$$
$$L = \{\varepsilon, abc, aab, aabb, bbc, cb, cab\};\ M = \{aa, ab, aabb, bab, cb, cc\}$$

Entonces

$$(L \cup M) = \{\varepsilon, aa, aab, aabb, ab, abc, bab, bbc, cab, cb, cc\}$$
$$(L \cap M) = \{aabb, cb\}$$
$$(L - M) = \{\varepsilon, abc, aab, bbc, cab\}$$
$$L' = \Sigma^* - L = \{x \mid x \in \Sigma^* \text{ y } x \notin \{\varepsilon, abc, aab, aabb, bbc, cb, cab\}\}$$

Lenguajes regulares.

Con el alfabeto Σ es posible formar una gran cantidad de lenguajes, pero no existe un método efectivo para saber cuántos de ellos son regulares. Todos los lenguajes sobre Σ son sublenguajes del lenguaje universo Σ^*, en lugar de ello se utilizarán algunas propiedades de los lenguajes para determinar cuáles de ellos son regulares. El conjunto de lenguajes regulares sobre Σ son:

- El lenguaje vacío $\varnothing$.
- El lenguaje que contiene la cadena vacía como único elemento $\{\varepsilon\}$.
- El lenguaje unitario $\{a\}$ es regular $\forall\ a \in \Sigma$.
- Si L y M son lenguajes regulares, entonces también lo son el lenguaje que resulta de la unión $(L \cup M)$, la concatenación (LM), la potenciación $(L^n$ o $M^n)$ y la cerradura de Kleene (L^*).
- Ningún otro lenguaje sobre Σ es regular.

Se puede concluir que a partir del alfabeto Σ los lenguajes obtenidos con las operaciones unión, concatenación, potenciación y cerradura de Kleene son lenguajes regulares, además de los lenguajes de un solo elemento, incluyendo la cadena vacía y el lenguaje vacío.

Expresiones regulares.

Es una nueva forma de expresar los lenguajes regulares y tiene como finalidad facilitar la manipulación y simplificación de los mismos. La equivalencia entre lenguajes regulares y expresiones regulares es como se indica en la siguiente tabla:

Lenguaje regular	Expresión regular
$\{a\}$	a
$\{\varepsilon\}$	ε
$\varnothing^*$	ε
$\{a\}^+$	a^+
$\{a\}^*$	a^*
$\{ab\}$	ab
$\{a\} \cup \{b\} = \{a, b\}$	$a \cup b$

De esta manera el lenguaje $\{a, bc\}^*\{\varepsilon, aab\}\varnothing\{c\}$ sobre $\Sigma = \{a, b, c\}$, se puede indicar por medio de una expresión regular:

$$(a \cup bc)^*(\varepsilon \cup aab)\, \varepsilon c$$

Los paréntesis solamente se usarán en casos extremos, ya que la finalidad de las expresiones regulares es la simplificación en la representación de los lenguajes regulares.

Autómatas finitos (AF).

Los AF constan de cinco elementos fundamentales AF = $(\Sigma, E, F, s, \delta)$. Un alfabeto (Σ), un conjunto de estados (E), estado inicial (s), un conjunto de estados finales (F) y una función δ: $E \times \Sigma \rightarrow E$ que permite determinar cuál es el estado siguiente.

Las expresiones regulares se pueden representar por medio de autómatas finitos, a su vez los autómatas finitos pueden expresarse en forma gráfica por medio de un *diagrama de transición* o bien por medio de una tabla que recibe el nombre de *tabla de transición*.

Diagrama de transición.

En los diagramas de transición, los estados se representan por medio de un círculo con el nombre del estado dentro de él. Los estados de aceptación o finales se distinguen porque tienen doble círculo, las transiciones se representan por aristas y se etiquetan con un símbolo del alfabeto. El estado inicial se distingue porque se hace incidir sobre él una flecha. Por ejemplo, el diagrama de transición que permite representar a la expresión regular $(0 \cup 1)^*01$ es el siguiente:

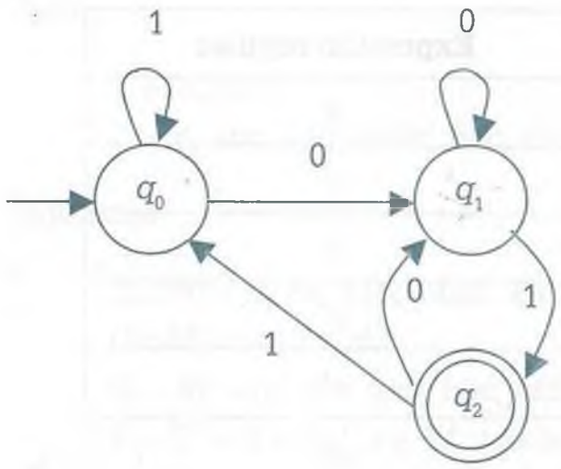


Diagrama de transición del autómata finito que reconoce cadenas de ceros y unos y cuya expresión regular es $(0 \cup 1)^*01$.

Aquí se tiene que

$\Sigma = \{0, 1\}$ es el alfabeto.

$E = \{q_0, q_1, q_2\}$ es el conjunto de estados.

$S = q_0$ es el estado inicial, que se reconoce por la flecha que incide en él.

$F = \{q_2\}$ conjunto de estados finales, que se distinguen por el doble círculo.

$\delta: E \times \Sigma \rightarrow E$ es la función para determinar el estado siguiente.

En el diagrama de transición se puede observar que pertenecen al lenguaje indicado por la expresión regular $(0 \cup 1)^*01$ cadenas de caracteres que terminan con 01. Por ejemplo, las cadenas 10001 y 101 pertenecen al lenguaje.

La función de transición $\delta: E \times \Sigma \rightarrow E$ permite determinar el estado siguiente, partiendo del par estado (E) y un símbolo del alfabeto (Σ) para acceder a otro estado del conjunto E. Por ejemplo, si la cadena es 10001, la ruta que se sigue para reconocer la cadena es:

$\delta(q_0, 1) = q_0$ Se inicia en el estado inicial q_0 y con el símbolo 1, nuevamente el estado siguiente es q_0, como se puede ver en el diagrama de transición.

$\delta(q_0, 0) = q_1$ De q_0 y con el símbolo 0 se traslada a estado q_1.

$\delta(q_1, 0) = q_1$ De q_1 y con el símbolo 0 permanece en q_1.

$\delta(q_1, 0) = q_1$ De q_1 y con el símbolo 0 permanece en q_1.

$\delta(q_1, 1) = q_2$ De q_1 y con el símbolo 1 se traslada a q_2.

Como q_2 es un estado de aceptación, se dice que la cadena 10001 pertenece al lenguaje.

Algunas veces es más fácil indicar la ruta que se sigue para evaluar la cadena de la siguiente manera:

$$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_1 \xrightarrow{0} q_1 \xrightarrow{1} q_2$$

De esta misma forma la cadena 1011010 no pertenece al lenguaje, como se muestra a continuación:

$$q_0 \xrightarrow{1} q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{1} q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_2 \xrightarrow{0} q_1$$

ya que el estado q_1 donde queda finalmente, no es un estado de aceptación.

Tabla de transición.

La información de un autómata, así como los valores que puede tomar la función δ, también se puede representar por medio de una tabla de transición. La tabla de transición que representa al AF anterior es:

δ	0	1
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0

$s = q_0$ y $F = \{q_2\}$

Tabla de transición

También por medio de esta tabla de transición, el estado inicial y los estados de aceptación, es posible determinar si alguna cadena pertenece o no a un lenguaje.

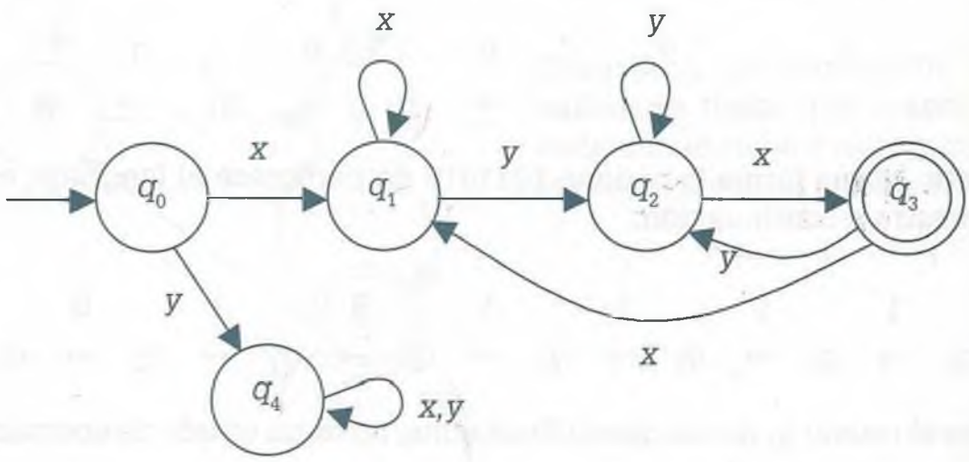
9.3.2 Autómatas finitos deterministicos (AFD)

Se dice que un autómata finito es determinístico si por medio de la función de transición δ: $E \times \Sigma \rightarrow E$ es posible determinar claramente cuál es el estado siguiente. El autómata anteriormente visto es un AFD, ya que cuando se está en un estado cualquiera y se tiene un símbolo del alfabeto, es posible determinar claramente cuál es el estado siguiente.

También es un AFD el siguiente, en donde:

$\Sigma = \{x, y\}$
$E = \{q_0, q_1, q_2, q_3, q_4\}$
$s = q_0$
$F = \{q_3\}$

cuyo diagrama de transición es:



Se observa que partiendo de cualquier estado del conjunto E y con un símbolo del alfabeto Σ, es posible acceder al estado siguiente por medio de la función de transición δ: $E \times \Sigma \rightarrow E$.

Con la tabla de transiciones, dicho AFD queda indicado completamente de la siguiente manera:

δ	x	y
q_0	q_1	q_4
q_1	q_1	q_2
q_2	q_2	q_3
q_4	q_4	q_4

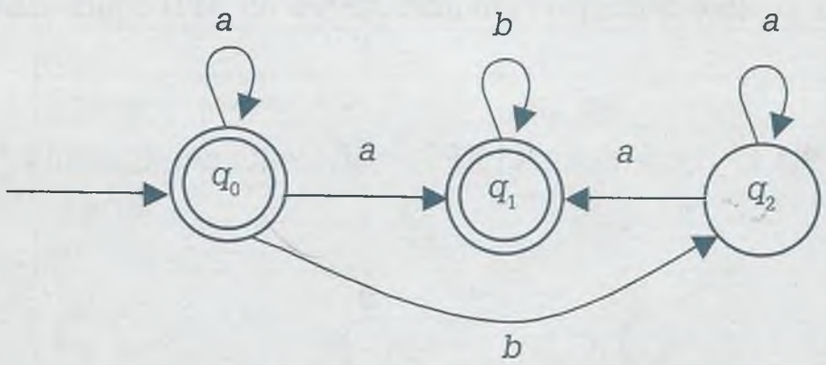
$s = q_0$ y $F = \{q_3\}$

Dicho AFD acepta cadenas de caracteres que comienzan con x y terminan con yx. Aquí el lenguaje regular está representado por la expresión regular $x(x \cup y)^* yx$, de forma que cadenas como *xxxyxyyyx*, *xyx*, *xyyyx*, pertenecen al lenguaje pero las cadenas yyxyx, xyxxy no son reconocidas por el AFD.

9.3.3 Autómatas finitos no deterministicos (AFN)

La diferencia fundamental entre un autómata finito determinístico (AFD) y un autómata finito no determinístico (AFN), es que en el AFN la función de estado siguiente no conduce a un estado único determinado.

Ejemplo 9.10. Considérese el siguiente diagrama:



Como se ve, se trata de un AFN porque no está determinado en todos los casos cuál es el estado siguiente para los diferentes símbolos del alfabeto. Por ejemplo en los estados q_0 y q_2 para un símbolo de *a* hay dos posibilidades de cambiar de estado o quedarse en el mismo. En el estado q_1 también hay problemas ya que no se sabe qué pasa en caso de que se presente el símbolo *a*. Con una indeterminación que se presente es suficiente para decir que se trata de un AFN.

En este caso

$\Sigma = \{a, b\}$
$E = \{q_0, q_1, q_2\}$
$s = q_0$
$F = \{q_0, q_1\}$

Este mismo AFN se puede representar por medio de una tabla de transición de la siguiente manera:

δ	a	*b*
q_0	$\{q_0, q_2\}$	q_2
q_1	$\varnothing$	q_1
q_2	$\{q_1, q_2\}$	$\varnothing$

$s = q_0$ y $F = \{q_0, q_1\}$

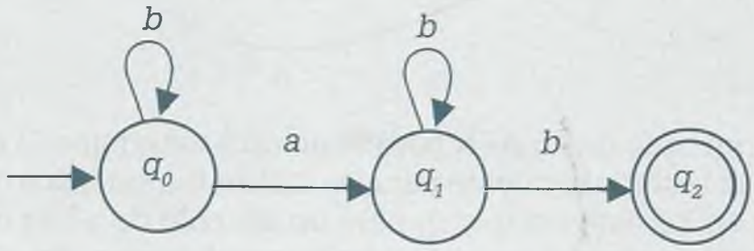
Una cadena es aceptada por un AFN si existe algún camino para esa cadena que comience en el estado inicial y termine en el estado aceptado.

Cadena	**¿Se acepta?**	**Ruta**
aabaabbb	Sí	$q_0, q_0, q_2, q_2, q_1, q_1, q_1, q_1$
bbba	No	q_0, q_2, indeterminado
bbaba	No	q_2, indeterminado
ε	Sí	Ya que el estado inicial q_0 es también un estado de aceptación

9.3.4 Conversión de un AFN a un AFD

Es posible convertir cualquier AFN a un AFD equivalente.

Ejemplo 9.11. Determinar el AFD equivalente del siguiente AFN.



Se observa que:

$\Sigma = \{a, b\}$
$E = \{q_0, q_1, q_2\}$
$s = q_0$
$F = \{q_2\}$

Se puede representar por medio de la tabla de transición, pero tomando como base el conjunto potencia $P(E)$.

Se sabe que el conjunto potencia $P(E)$ está integrado por todos los subconjuntos del conjunto E, y que el número de subconjuntos del conjunto potencia está dado por:

$$| P(E) | = 2^n$$

donde n es el número de elementos de E.

En este caso el conjunto E tiene 3 elementos, por lo tanto $| P(E) | = 2^3 = 8$, los cuales son:

$$P(E) = \{\varnothing, \{q_0\}, \{q_1\}, \{q_2\}, \{q_0, q_1\}, \{q_0, q_2\}, \{q_1, q_2\}, \{q_0, q_1, q_2\}\}$$

Tomando en cuenta que la tabla de transiciones debe integrarse con todos los elementos de $P(E)$, por lo tanto ésta queda de la siguiente manera

Elemento	*a*	*b*
$\varnothing$	$\varnothing$	$\varnothing$
$\{q_0\}$	$\{q_1\}$	$\{q_0\}$
$\{q_1\}$	$\varnothing$	$\{q_1, q_2\}$
$\{q_2\}$	$\varnothing$	$\varnothing$
$\{q_0, q_1\}$	$\{q_1\}$	$\{q_0, q_1, q_2\}$
$\{q_0, q_2\}$	$\{q_1\}$	$\{q_0\}$
$\{q_1, q_2\}$	$\varnothing$	$\{q_1, q_2\}$
$\{q_0, q_1, q_2\}$	$\{q_1\}$	$\{q_0, q_1, q_2\}$

Hay que observar las aristas en cada uno de los nodos del diagrama de transición para poder llenar la tabla de transiciones, por ejemplo, el elemento $\{q_0, q_1\}$ tiene $\{q_1\}$ en la columna *a*, lo cual significa que saliendo de q_0 y símbolo *a* se puede acceder solamente al mismo $\{q_1\}$ y saliendo de q_1 y símbolo *a* no se puede acceder a ningún otro elemento, por lo tanto se queda únicamente $\{q_1\}$. El elemento $\{q_0, q_1\}$ tiene en la columna *b* $\{q_0, q_1, q_2\}$, lo cual indica que saliendo de $\{q_0\}$ y símbolo *b*, es posible acceder al mismo $\{q_0\}$, saliendo de $\{q_1\}$ y símbolo *b* es posible acceder $\{q_1\}$ o $\{q_2\}$, de forma que uniéndolos se tiene que partiendo de $\{q_0, q_1\}$ y símbolo *b*, se puede acceder a $\{q_0, q_1, q_2\}$.

El estado siguiente en la tabla de transiciones se llena tomando en cuenta que el estado siguiente del conjunto vacío ($\varnothing$) siempre será $\varnothing$, de forma que:

$$\varnothing = \delta(\varnothing, a) = \varnothing$$

$$\varnothing = \delta(\varnothing, b) = \varnothing$$

Los estados siguientes q_0, q_1 y q_2 se toman del diagrama de transiciones y los restantes estados se pueden obtener como sigue:

$\{q_0, q_1\} = \delta(\{q_0, q_1\}, a) = \delta\{q_0, a\} \cup \delta\{q_1, a\} = \{q_1\} \cup \varnothing = \{q_1\}$

$\{q_0, q_1\} = \delta(\{q_0, q_1\}, b) = \delta\{q_0, b\} \cup \delta\{q_1, b\} = \{q_0\} \cup \{q_1, q_2\} = \{q_0, q_1, q_2\}$

$\{q_0, q_2\} = \delta(\{q_0, q_2\}, a) = \delta\{q_0, a\} \cup \delta\{q_2, a\} = \{q_1\} \cup \varnothing = \{q_1\}$

$\{q_0, q_2\} = \delta(\{q_0, q_2\}, b) = \delta\{q_0, b\} \cup \delta\{q_2, b\} = \{q_0\} \cup \varnothing = \{q_0\}$

$\{q_1, q_2\} = \delta(\{q_1, q_2\}, a) = \delta\{q_1, a\} \cup \delta\{q_2, a\} = \varnothing \cup \varnothing = \varnothing$

$\{q_1, q_2\} = \delta(\{q_1, q_2\}, b) = \delta\{q_1, b\} \cup \delta\{q_2, b\} = \{q_1, q_2\} \cup \varnothing = \{q_1, q_2\}$

$\{q_0, q_1, q_2\} = \delta(\{q_0, q_1, q_2\}, a) = \delta\{q_0, a\} \cup \delta\{q_1, a\} \cup \delta\{q_2, a\} = \{q_1\} \cup \varnothing \cup \varnothing = \{q_1\}$

$\{q_0, q_1, q_2\} = \delta(\{q_0, q_1, q_2\}, b) = \delta\{q_0, b\} \cup \delta\{q_1, b\} \cup \delta\{q_2, b\} = \{q_0\} \cup \{q_1, q_2\} \cup \varnothing$
$= \{q_0, q_1, q_2\}$

Estos últimos también se pueden obtener de la siguiente forma:

$\{q_0, q_1, q_2\} = \delta(\{q_0, q_1, q_2\}, a) = \delta\{q_0, a\} \cup \delta\{q_1, q_2, a\} = \{q_1\} \cup \varnothing = \{q_1\}$

$\{q_0, q_1, q_2\} = \delta(\{q_0, q_1, q_2\}, b) = \delta\{q_0, q_2, b\} \cup \delta\{q_1, b\} = \{q_0\} \cup \{q_1, q_2\} = \{q_0, q_1, q_2\}$

Haciendo

$\{q_0\} = \{e_0\}$ $\quad \{q_0, q_1\} = \{e_3\}$ $\quad \{q_0, q_1, q_2\} = \{e_6\}$

$\{q_1\} = \{e_1\}$ $\quad \{q_0, q_2\} = \{e_4\}$

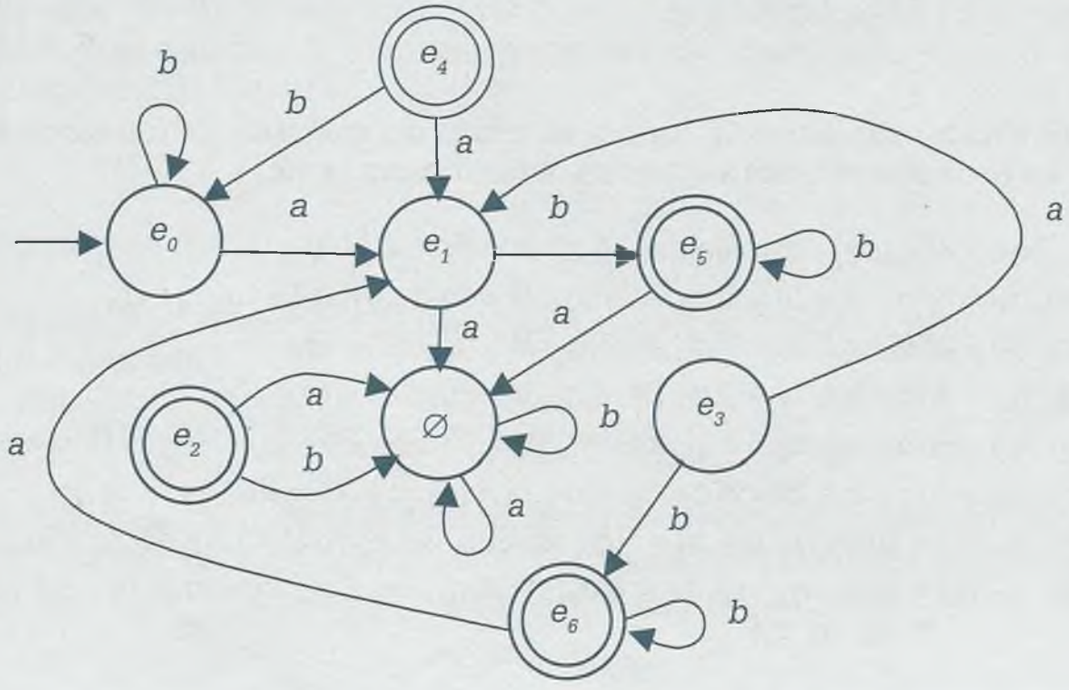
$\{q_2\} = \{e_2\}$ $\quad \{q_1, q_2\} = \{e_5\}$

y considerando además que los estados aceptados son aquellos que contienen a q_2, se tiene la siguiente tabla de transiciones:

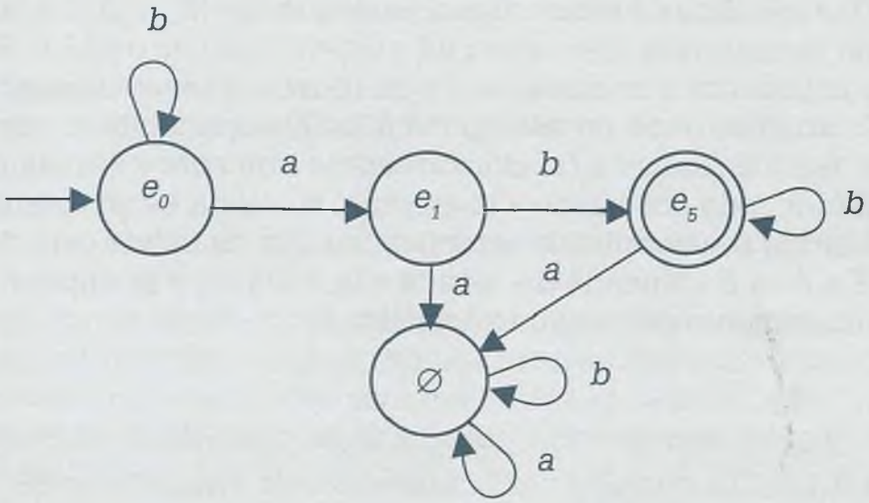
Elemento	*a*	*b*
$\varnothing$	$\varnothing$	$\varnothing$
$\{e_0\}$	$\{e_1\}$	$\{e_0\}$
$\{e_1\}$	$\varnothing$	$\{e_5\}$
$\{e_2\}$	$\varnothing$	$\varnothing$
$\{e_3\}$	$\{e_1\}$	$\{e_6\}$
$\{e_4\}$	$\{e_1\}$	$\{e_0\}$
$\{e_5\}$	$\varnothing$	$\{e_5\}$
$\{e_6\}$	$\{e_1\}$	$\{e_6\}$

En esta tabla se tiene que e_0 es estado inicial, e_2, e_4, e_5 y e_6 son estados de aceptación.

Así el diagrama de transición queda de la siguiente forma:



Eliminando los estados que no se tocan, se tiene el siguiente AFD equivalente:



Esto no significa que los AFD sean mejores que los AFN, porque en muchos casos es mejor usar un AFN para representar lenguajes, además de que son más compactos. Sin embargo en este caso lo que se hizo fue mostrar que es posible convertir un AFN en un AFD.

9.4 Máquinas de estado finito

Una máquina de estado finito es una forma especial de representar los autómatas finitos, en donde no existen estados aceptados y donde los símbolos de salida se colocan junto con los símbolos de entrada en cada una de las aristas de la máquina. Las máquinas de estado finito son muy utilizadas para desarrollar innumerables tareas. Una de ellas, quizá la más popular y compleja es la computadora, la cual recibe información de algún dispositivo de entrada, la procesa y más tarde la manda a un dispositivo de salida.

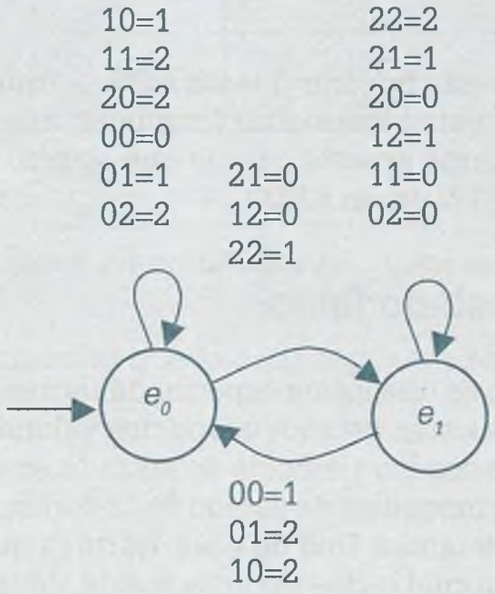
Existen máquinas de estado finito más sencillas, por ejemplo el funcionamiento de un elevador, el cual con base en la información que recibe, cierra o abre la puerta, se traslada a cierto piso, antes de cerrar la puerta espera un cierto tiempo, etc. La información se le da a través de teclas, sensores o impulsos de reloj y las salidas se observan claramente. Sin embargo, lo que no se puede ver es cómo toma la decisión, que es en sí la esencia de los autómatas y máquinas de estado finito.

Las máquinas tragamonedas es otro ejemplo típico de las máquinas de estado finito, las cuales funcionan solamente con cierta señal y permiten seleccionar el juego y número de jugadores. En algunas de ellas el juego

se interrumpe cuando ha trascurrido cierto tiempo, en otras porque el usuario llegó a la meta. Todo esto y otros estados se deberán considerar en una máquina de estado finito.

Una máquina de estado finito, en donde $M = \{E, A, B, s, \delta, \sigma\}$ se define como un sistema que cuenta con un número finito de estados E, que acepta un conjunto finito de entradas A y que puede tener un número finito de salidas B, además tiene un estado inicial s. Cuenta también con dos funciones: δ, también llamada *función de estado siguiente* y se define con σ: $E \times A \rightarrow E$ (depende del estado y la entrada, su salida es un estado siguiente) y la función δ denominada también *función de salida* está definida como δ: $E \times A \rightarrow B$ (depende del estado y la entrada, y se obtiene como resultado un elemento del conjunto de salida B).

Ejemplo 9.12. La máquina para sumar dos cantidades de base 3 es como se muestra a continuación:



Esta máquina de estado finito está integrada por:

$E = \{e_0, e_1\}$	Conjunto de estados.
$A = \{00, 01, 02, 10, 11, 12, 20, 21, 22\}$	Conjunto finito de entradas.
$B = \{0, 1, 2\}$	Conjunto de salidas.
$s = e_0$ Estado inicial.	

Se observa claramente que ninguno de los estados del conjunto E es de aceptación, porque las máquinas de estado finito no tienen estados aceptados. El conjunto $E = \{e_0, e_1\}$ tiene dos estados, e_0 que a su vez es el esta-

do inicial y el estado en el que no hay ningún acarreo. El estado e_1 es la posición en que se le suma el acarreo a los nuevos bits sumados y permanece ahí, siempre y cuando la suma de los bits nuevos y el acarreo generen un nuevo acarreo, en caso contrario regresará al estado e_0.

El conjunto *A* contiene todas las combinaciones posibles que se pueden presentar cuando se suman dos cantidades en binario {00, 01, 02, 10, 11, 12, 20, 21, 22}.

El conjunto *B* contiene todos aquellos símbolos de salida posibles {0, 1, 2}.

La máquina de estado finito también se puede representar por medio de su tabla de transición, de igual forma a como se hizo con autómatas finitos. En este caso para sumar dos cantidades de base 3 la tabla de transiciones que representa a la máquina de estados finitos es como se muestra a continuación:

Edo.	δ									σ								
	00	01	02	10	11	12	20	21	22	00	01	02	10	11	12	20	21	22
e_0	e_0	e_0	e_0	e_0	e_0	e_1	e_0	e_1	e_1	0	1	2	1	2	0	2	0	1
e_1	e_0	e_0	e_1	e_0	e_1	e_1	e_1	e_1	e_1	1	2	0	2	0	1	0	1	2

Como ejemplo considérese si los números que se suman en el sistema tercianario son 10222 y 1201. Lo primero que hay que tener en cuenta es que las cadenas deben ser de la misma longitud y en caso de no ser así se completa con ceros a la izquierda la cadena de menor longitud. Después se le agrega un 0 a la izquierda de cada cantidad, con la finalidad de que la máquina regrese al estado e_0 después de sumar las cantidades de forma que las cadenas de ceros y unos quedan de la siguiente manera: 010222 y 001201.

Tomando los caracteres que están más a la derecha de cada una de las cadenas, se tienen los siguientes valores para las funciones de estado siguiente δ y de salida σ:

$\delta(e_0, 21) = e_1$ $\quad \sigma(e_0, 21) = 0$ $\quad$ 0 y se lleva 1
$\delta(e_1, 20) = e_1$ $\quad \sigma(e_1, 20) = 0$ $\quad$ 0 y se lleva 1
$\delta(e_1, 22) = e_1$ $\quad \sigma(e_1, 22) = 2$ $\quad$ 2 y se lleva 1
$\delta(e_1, 01) = e_0$ $\quad \sigma(e_1, 01) = 2$
$\delta(e_0, 10) = e_0$ $\quad \sigma(e_0, 10) = 1$
$\delta(e_0, 00) = e_0$ $\quad \sigma(e_0, 00) = 0$

De esta forma se tiene que

$$010222_{(3)} + 001201_{(3)} = 012200_{(3)}$$

9.4.1 Equivalencia entre autómatas finitos y máquinas de estado finito

Un autómata finito se puede considerar como una máquina de estado finito. Recordando que los autómatas tienen los elementos AF = (Σ, E, F, s, δ) y que los de las máquinas de estado finito son M = {E, A, B, a, δ, σ}, se puede observar cierta equivalencia entre ellos que permite determinar la máquina de estado finito equivalente de un autómata finito y viceversa. La equivalencia entre elementos es como sigue:

a) El alfabeto finito de símbolos de un autómata, equivale al conjunto de símbolos finitos de entrada de una máquina de estado finito $\Sigma = A$.

b) El conjunto de estados en un AF, equivale al conjunto de estados en una máquina de estado finito $E = E$.

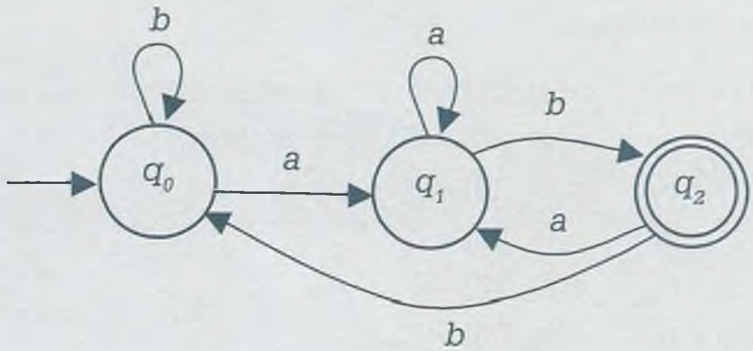
c) El conjunto de estados aceptados en un AF, equivale a las transiciones que inciden en un estado y cuyo símbolo de salida es 1.

d) La función de estado siguiente δ en un AF, equivale a la función de estado siguiente δ en una máquina de estado finito $\delta = \delta$.

e) El estado inicial s en un AF, equivale al estado inicial s de la máquina de estado finito $s = s$.

f) En los autómatas finitos no existe la función de salida σ de una máquina, ya que para determinar si una palabra pertenece a un lenguaje se tienen los estados aceptados.

Ejemplo 9.13. Considérese el siguiente autómata que acepta cadenas de caracteres que terminan en *ab* como *bab*, *bbab*, *aab*, *aababab*, cuyo diagrama de transiciones se indica a continuación



donde

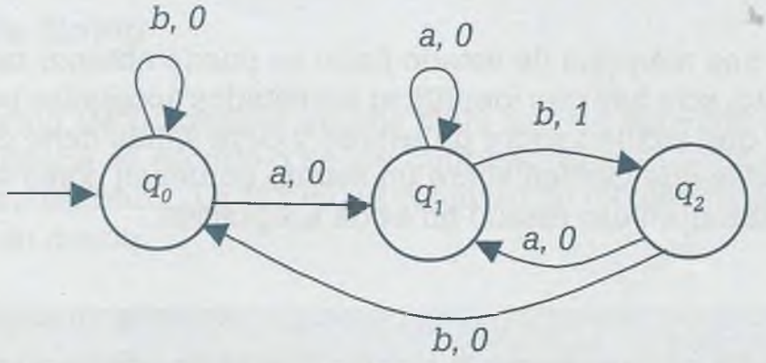
$\Sigma = \{a, b\}$ $F = \{q_2\}$

$E = \{q_0, q_1, q_2\}$ $s = q_0$

y cuya función de estado está expresada por la siguiente tabla de transiciones

δ	a	b
q_0	q_1	q_0
q_1	q_1	q_2
q_2	q_1	q_0

La máquina de estado finito equivalente es semejante al autómata finito, solamente desaparece el estado aceptado y se coloca en las aristas, además del símbolo propio del alfabeto, el símbolo de salida. Por lo general los símbolos de salida en una máquina de estado finito son 1 o 0; si sobre un estado todas las aristas que inciden tienen como símbolo de salida 1, entonces ese estado se considera aceptado:



donde

$A = \{a, b\}$

$E = \{q_0, q_1, q_2\}$

$s = q_0$

Las funciones de estado siguiente δ y de salida σ se pueden expresar por medio de la siguiente tabla de estados:

Edo.	δ		σ	
	a	b	a	b
q_0	q_1	q_0	0	0
q_1	q_1	q_2	0	1
q_2	q_1	q_0	0	0

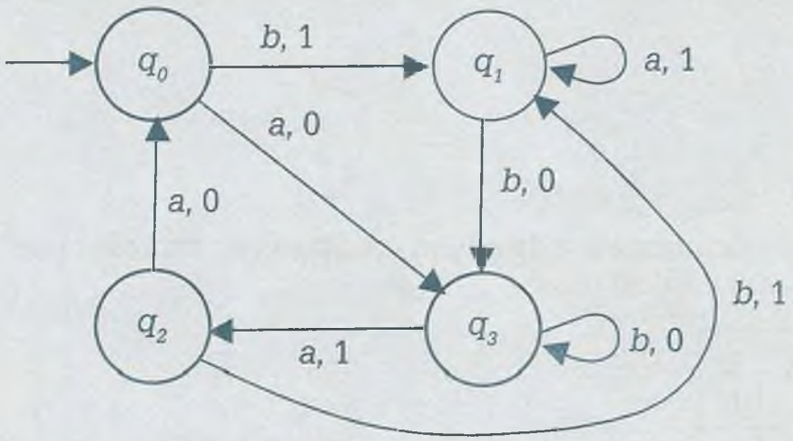
Para probar si la palabra *bbaabbab* pertenece al lenguaje, se puede usar el diagrama de transiciones haciendo pasar por la máquina de estado finito la cadena y observar que al final el símbolo de salida es un 1, lo cual significa que la palabra es aceptada por el lenguaje. O bien usando para ello las funciones de estado siguiente y de salida:

$$\begin{aligned}
&\delta(q_0, b) = q_0 \qquad \sigma(q_0, b) = 0\\
&\delta(q_0, b) = q_0 \qquad \sigma(q_0, b) = 0\\
&\delta(q_0, a) = q_1 \qquad \sigma(q_0, a) = 0\\
&\delta(q_1, a) = q_1 \qquad \sigma(q_1, a) = 0\\
&\delta(q_1, b) = q_2 \qquad \sigma(q_1, b) = 1\\
&\delta(q_2, b) = q_0 \qquad \sigma(q_2, b) = 0\\
&\delta(q_0, a) = q_1 \qquad \sigma(q_0, a) = 0\\
&\delta(q_1, b) = q_2 \qquad \sigma(q_1, b) = 1
\end{aligned}$$

Como el símbolo de salida al procesar toda la cadena es un 1, entonces se considera que la palabra *bbaabbab* pertenece al lenguaje.

De una máquina de estado finito se puede obtener también un autómata finito, sólo hay que identificar los estados aceptados por medio de las aristas que inciden sobre un estado y cuya salida debe de ser 1. Si todas las aristas que inciden sobre un estado no tienen como símbolo de salida un 1, entonces ese estado no es de aceptación.

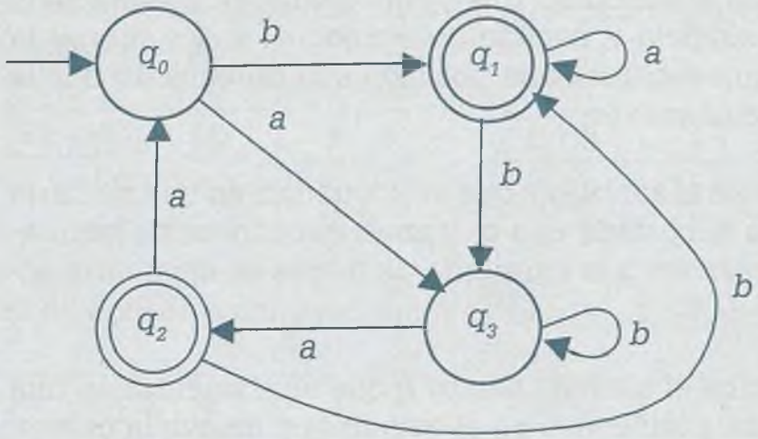
Ejemplo 9.14. Considérese la siguiente máquina de estado finito:



Aquí:
$A = \{a, b\}$
$E = \{q_0, q_1, q_2, q_3,\}$
$s = q_0$

Aquí hay que observar cómo sobre los estados q_1 y q_2 inciden solamente aristas cuya salida es 1, de forma que ésos son los estados aceptados en

el autómata finito. El autómata finito equivalente es como se muestra a continuación:



Aquí:
$\Sigma = \{a, b\}$
$E = \{q_0, q_1, q_2, q_3,\}$
$s = q_0$

Se puede decir que los autómatas finitos y las máquinas de estado finito son semejantes, y por lo tanto factibles de usarse para representar lenguajes regulares.

9.4.2 Máquinas de Turing

Alan Mathison Turing
(1912-1954)

Fue un Matemático, informático teórico, criptógrafo y filósofo inglés que es considerado uno de los fundadores de la ciencia de la computación y uno de los precursores de la informática moderna. Con su paradigma conocido como máquina de Turing proporcionó una influyente formalización de los conceptos de algoritmo y computación. También formuló su propia versión de la hoy ampliamente aceptada tesis de Church-Turing, la cual postula que cualquier modelo computacional existente tiene las mismas capacidades algorítmicas, o un subconjunto, de las que tiene una máquina de Turing.

Una máquina de Turing (MT) consiste en una cinta que se extiende de manera infinita, en donde se escribe o se lee información por medio de una cabeza de lectura-escritura. La MT es un conjunto de elementos MT $= (E, \Sigma, A, s, \flat, F, \delta)$ en donde:

E: Conjunto finito de estados.

Σ: Conjunto de símbolos de entrada.

A: Alfabeto de la cinta.

s: estado inicial $s \in E$.

$\flat$: Símbolo blanco, el cuál es un elemento del alfabeto de la cinta, pero no del conjunto de símbolos de entrada; $\flat \in A, \flat \notin \Sigma$.

F: Conjunto de estados de aceptación.

δ: Función de transición en donde δ: $E \times A \rightarrow E \times A \times \{D, I\}$.

En la función de transición entra el estado actual en donde se encuentra la MT (e) y un símbolo de la cinta (σ) para obtenerse con dicha función una terna conformada por el estado siguiente (f), el símbolo escrito en la cinta (σ) y un movimiento de la cinta (m)

$$\delta(e, \sigma) = (f, \sigma, m)$$

en donde m es un movimiento I = izquierda, D = derecha.

Ejemplo 9.15.

$\delta(e_0, x) = (e_1, y, D)$: cambia el símbolo x que se encuentra en una celda de la cinta por el símbolo y, cambia del estado e_0 a e_1 y mueve la cabeza de lectura-escritura una posición a la derecha de donde se encuentra actualmente.

$\delta(e_1, x) = (e_2, x, I)$: reescribe el símbolo x que se encuentra en una celda de la cinta, cambia del estado e_1 a e_2 y mueve la cabeza de lectura-escritura una posición a la izquierda de donde se encuentra actualmente.

$\delta(e_1, \not b) = (e_1, \not b, I)$: reescribe el símbolo blanco $\not b$ que se encuentra en una celda de la cinta, permanece en el estado e_1 y mueve la cabeza de lectura-escritura una posición a la izquierda de donde se encuentra actualmente.

Considérese que se tiene una MT con los siguientes elementos:

$E = \{e_0, e_1, e_2\}$	Conjunto de estados de la MT.
$\Sigma = \{0, 1\}$	Alfabeto de entrada.
$A = \{0, 1, \not b\}$	Alfabeto de la cinta.
$F = \{e_2\}$	Estado de aceptación.
$s = e_0$	Estado inicial.

La función de transición δ tiene las siguientes composiciones:

$\delta(e_0, 0) = (e_0, 1, D)$	$\delta(e_1, 0) = (e_1, 0, I)$
$\delta(e_0, 1) = (e_0, 0, D)$	$\delta(e_1, 1) = (e_1, 1, I)$
$\delta(e_0, \not b) = (e_1, \not b, I)$	$\delta(e_1, \not b) = (e_2, \not b, D)$

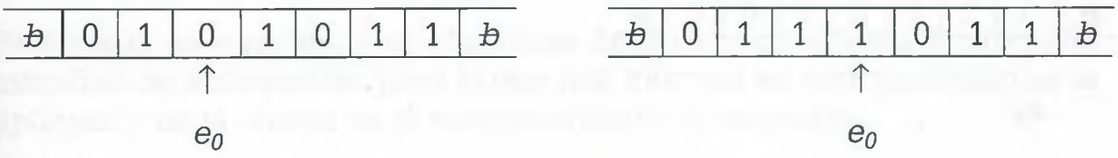
En este caso la MT encuentra el complemento de una cadena de unos y ceros y regresa al estado inicial e_0.

Considérese que en la cinta se encuentra la cadena 1001011, que el estado inicial es e_0 y que la cabeza de lectura-escritura está sobre el símbolo que indica la flecha en la siguiente figura

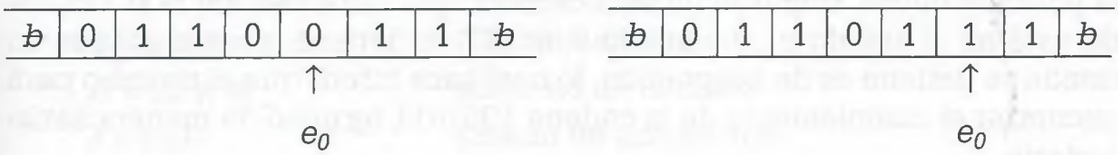
$\not b$	1	0	0	1	0	1	1	$\not b$
	↑							
	e_0							

$\not b$	0	0	0	1	0	1	1	$\not b$
		↑						
		e_0						

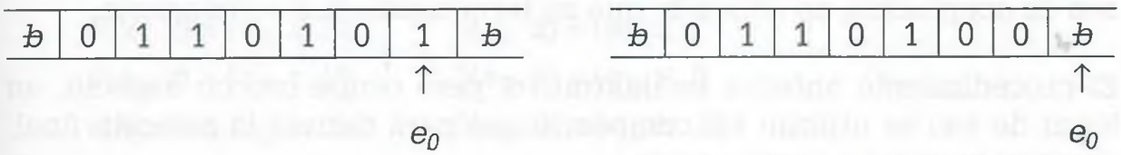
Como la cabeza de lectura-escritura se encuentra sobre el símbolo de la cinta "1" y la MT se encuentra en el estado e_0, entonces se utiliza la composición $\delta(e_0, 1) = (e_0, 0, D)$ que cambia el 1 por 0 y la cabeza de lectura-escritura se mueve una posición a la derecha de la cinta. Observar también cómo el estado no cambia.



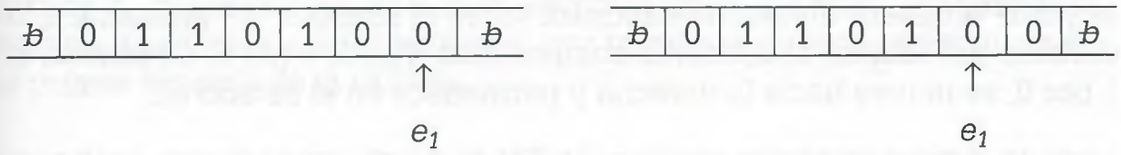
Después de aplicar la composición $\delta(e_0, 0) = (e_0, 1, D)$ en dos ocasiones



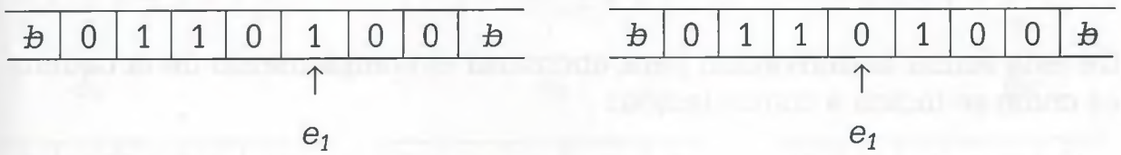
Después de aplicar las composiciones $\delta(e_0, 1) = (e_0, 0, D)$ y $\delta(e_0, 0) = (e_0, 1, D)$



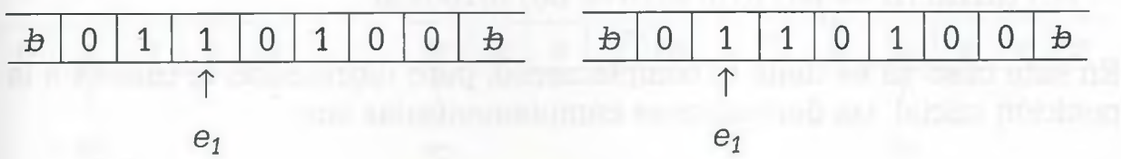
Después de aplicar la composición $\delta(e_0, 1) = (e_0, 0, D)$ en dos ocasiones

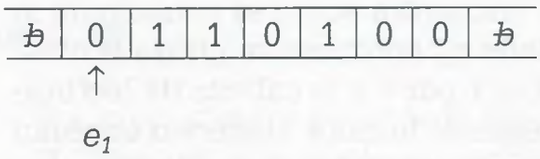


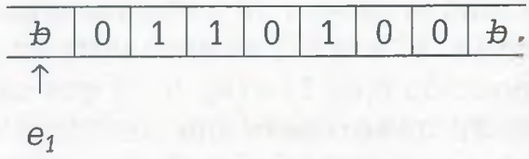
Después de aplicar las composiciones $\delta(e_0, b) = (e_1, b, I)$ y $\delta(e_1, 0) = (e_1, 0, I)$

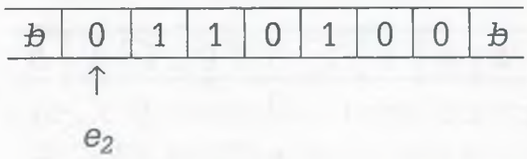


Después de aplicar las composiciones: $\delta(e_1, 0) = (e_1, 0, I)$ y $\delta(e_1, 1) = (e_1, 1, I)$









Hay que observar cómo al final se aplicó la composición $\delta(e_1, \not b) = (e_2, \not b, D)$, que reescribe el símbolo blanco y mueve la cabeza de lectura-escritura a la posición inicial. Como ya no está definido qué debe realizar la MT cuando está en el estado e_2, y símbolo 0, la MT se detiene, pero el estado en donde se detiene es de aceptación, lo cual hace inferir que el proceso para encontrar el complemento de la cadena 1001011 terminó de manera satisfactoria.

No es necesario que la cabeza regrese a la posición inicial, ya que podría haber quedado al final de la cadena, pero siempre y cuando el estado final sea de aceptación, se entiende que su terminación fue satisfactoria.

El procedimiento anterior es ilustrativo, pero ocupa mucho espacio, en lugar de eso se utilizan las composiciones para derivar la posición final, como se muestra a continuación.

$\not b e_0 1001011 \not b \mapsto \not b 0 e_0 001011 \not b$. Lo cual significa que si se está en el estado e_0 y con la cabeza de lectura-escritura sobre el símbolo "1" (que está a la derecha del estado) al aplicar la composición $\delta(e_0, 1) = (e_0, 0, D)$ cambia el 1 por 0, se mueve hacia la derecha y permanece en el estado e_0.

$\not b 0 e_0 001011 \not b \mapsto \not b 01 e_0 01011 \not b$: Si se está en el estado e_0 con la cabeza de lectura-escritura sobre el símbolo "0" (que está a la derecha del estado) al aplicar la composición $\delta(e_0, 0) = (e_0, 1, D)$ cambia el 0 por 1, se mueve hacia la derecha y permanece en el estado e_0.

De esta forma la derivación para encontrar el complemento de la cadena es como se indica a continuación:

$\not b e_0 1001011 \not b \mapsto \not b 0 e_0 001011 \not b \mapsto \not b 01 e_0 01011 \not b \mapsto \not b 011 e_0 1011 \not b \mapsto \not b 0110 e_0 011 \not b$

$\mapsto \not b 01101 e_0 11 \not b \mapsto \not b 011010\ e_0 1 \not b \mapsto \not b 0110100 e_0 \not b$

En este caso ya se tiene el complemento, pero regresando la cabeza a la posición inicial, las derivaciones complementarias son:

$\not b0110100e_0\not b \mapsto \not b011010e_10\not b \mapsto \not b01101e_100\not b \mapsto \not b0110e_1100\not b \mapsto \not b011e_10100\not b$

$\mapsto \not b01e_110100\not b \mapsto \not b0e_1110100\not b \mapsto \not be_10110100\not b \mapsto e_1\not b0110100\not b$

$\mapsto \not be_20110100\not b$

Realmente se pueden crear Máquinas de Turing para llevar a cabo una infinidad de actividades, pero lo que nos interesa en este momento es la aplicación de la misma en el reconocimiento de lenguajes.

Considérese que se tiene una MT con los siguientes elementos:

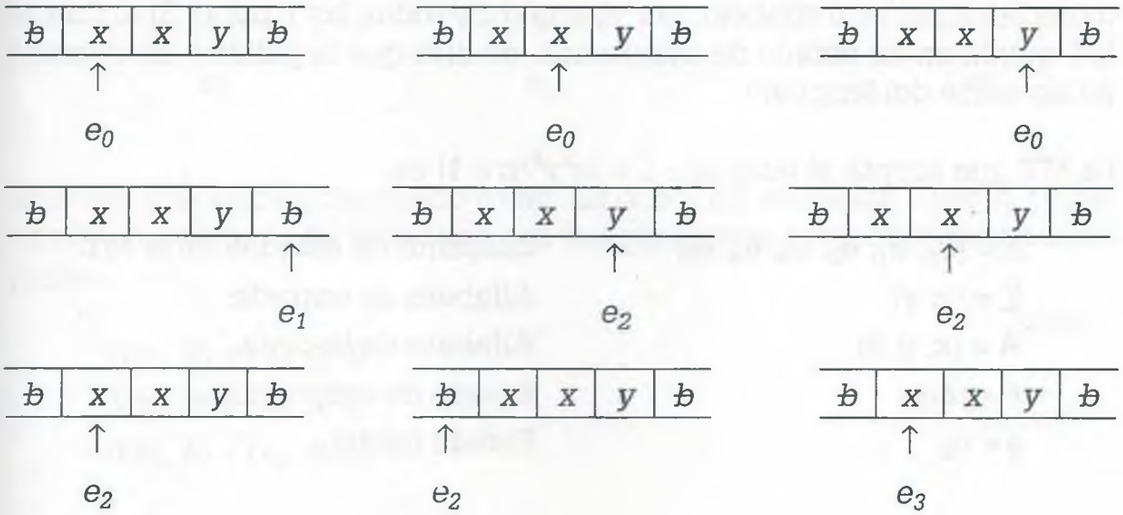
$E = \{e_0, e_1, e_2, e_3\}$	Conjunto de estados de la MT.
$\Sigma = \{x, y\}$	Alfabeto de entrada.
$A = \{x, y, \not b\}$	Alfabeto de la cinta.
$F = \{e_3\}$	Estado de aceptación.
$s = e_0$	Estado inicial.

La función de transición δ tiene las siguientes composiciones:

$\delta(e_0, x) = (e_0, x, D)$	$\delta(e_2, x) = (e_2, x, I)$
$\delta(e_0, y) = (e_1, y, D)$	$\delta(e_2, y) = (e_2, y, I)$
$\delta(e_1, \not b) = (e_2, \not b, I)$	$\delta(e_2, \not b) = (e_3, \not b, D)$

Dicha MT permite reconocer palabras del lenguaje representado por la expresión regular x^*y, tales como: y, xy, xxy, $xxxy$, si se comienza en el estado e_0, con la cabeza de lectura-escritura sobre el primer símbolo de la cadena y se termina en el estado e_3 con la cabeza de lectura-escritura en el primer símbolo de la cadena.

La cadena xxy es aceptada por la MT y la palabra xyx no se acepta, ya que no pertenece al lenguaje, como se muestra a continuación:



Como e_3 es un estado de aceptación, se dice que *xxy* pertenece al lenguaje $L = \{x^n y / n \geq 0\}$ representado por la expresión regular $x^* y$. Esto mismo se puede demostrar por medio de derivaciones como se muestra a continuación:

$\not b e_0 xxy \not b \mapsto \not b x e_0 xy \not b \mapsto \not b xx e_0 y \not b \mapsto \not b xxy e_1 \not b \mapsto \not b xx e_2 y \not b \mapsto \not b x e_2 xy \not b \mapsto \not b e_2 xxy \not b$

$\mapsto e_2 \not b xxy \not b \mapsto \not b e_3 xxy \not b$

Sin embargo, la cadena *xyx* no se acepta por el lenguaje usando la MT, como se muestra a continuación:

$\not b$	x	y	x	$\not b$
	↑			
	e_0			

$\not b$	x	y	x	$\not b$
		↑		
		e_0		

$\not b$	x	y	x	$\not b$
			↑	
			e_1	

porque la función de transición (e_1, x) no está definida, por lo tanto la MT se detiene en un estado e_1 que no es de aceptación y se concluye que la cadena *xyx* no pertenece al lenguaje $L = \{x^n y / n \geq 0\}$.

Realmente las MT son más potentes que los autómatas finitos, ya que admiten lenguajes regulares, pero también lenguajes que no son regulares, como el lenguaje $L = \{x^n y^n / n \geq 1\}$ el cual es un lenguaje libre de contexto y que solamente es reconocido por autómatas de pila ya que se requiere registrar cuántas veces se repite cada uno de los símbolos para posteriormente comprobar si el número de símbolos es igual, como ocurre con las palabras (*xy, xxyy, xxxyyy, xxxxyyyy,*...) que pertenecen al lenguaje $L = \{x^n y^n / n \geq 1\}$.

Para determinar si una palabra *w* pertenece al lenguaje anterior, se puede hacer de diferente manera, pero una de las formas más fáciles es cambiar todas las *x* por otro símbolo, por ejemplo *c* y todas las *y* por *d*. Si al final la MT queda en un estado de aceptación, se dice que la palabra en cuestión forma parte del lenguaje.

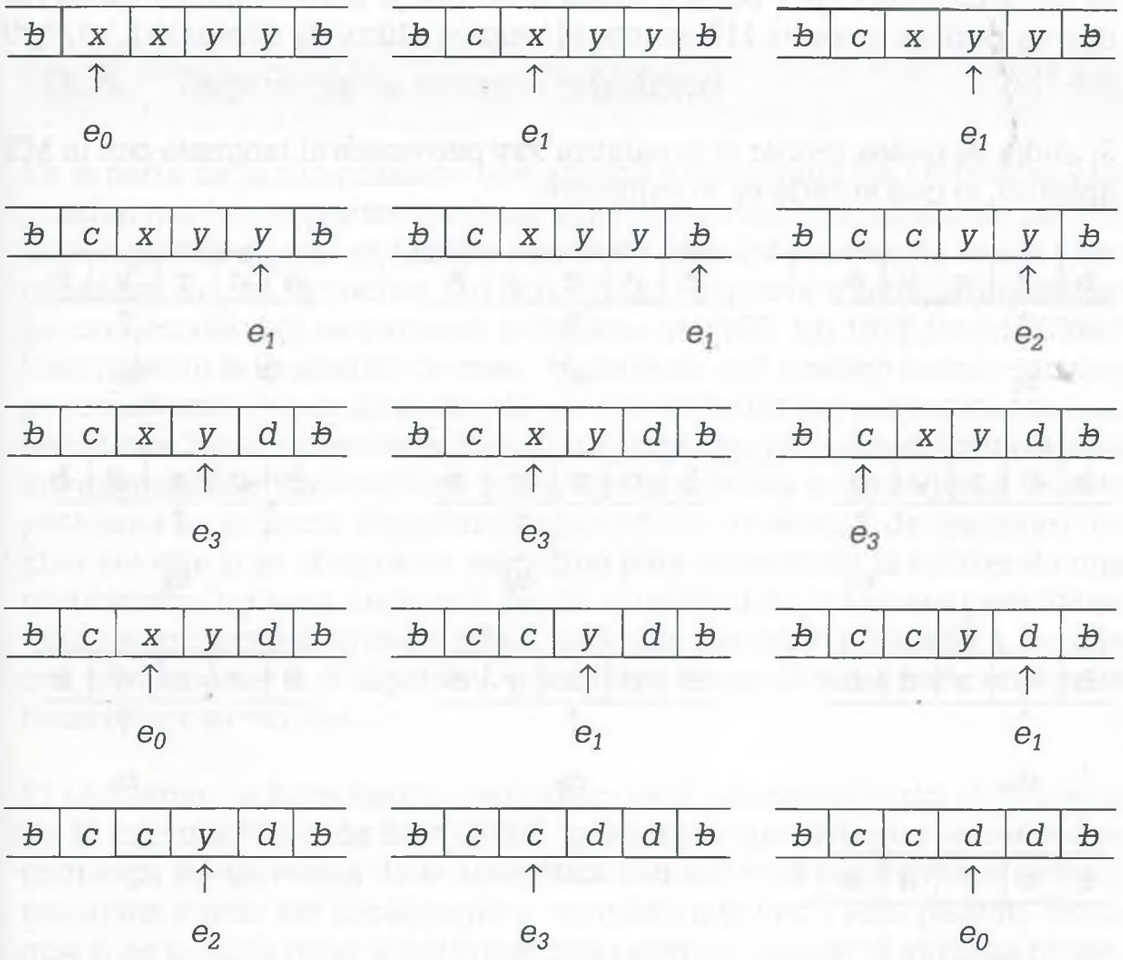
La MT que acepta el lenguaje $L = \{x^n y^n / n \geq 1\}$ es:

$E = \{e_0, e_1, e_2, e_3, e_4, e_5\}$	Conjunto de estados de la MT.
$\Sigma = \{x, y\}$	Alfabeto de entrada.
$A = \{x, y, \not b\}$	Alfabeto de la cinta.
$F = \{e_5\}$	Estado de aceptación.
$s = e_0$	Estado inicial.

La función de transición δ tiene las siguientes composiciones:

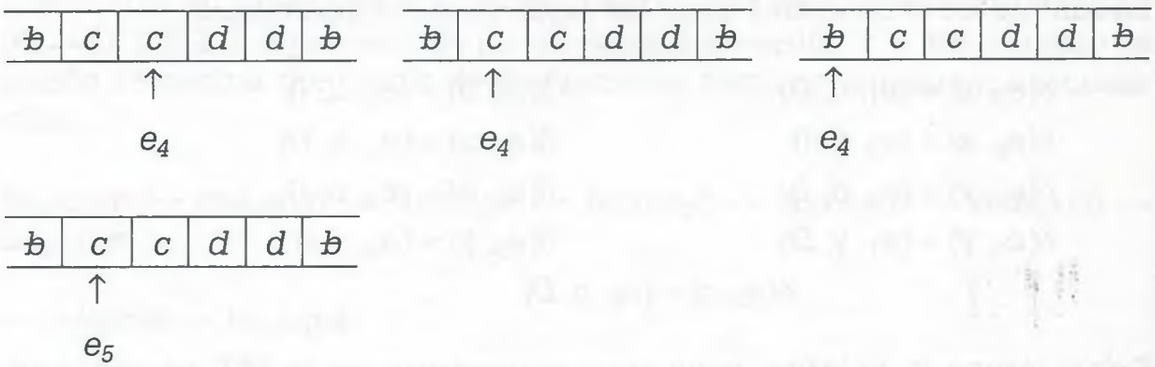
$\delta(e_0, x) = (e_1, c, D)$ $\quad$ $\delta(e_1, d) = (e_2, d, I)$
$\delta(e_3, x) = (e_3, x, I)$ $\quad$ $\delta(e_1, x) = (e_1, x, D)$
$\delta(e_2, y) = (e_3, d, I)$ $\quad$ $\delta(e_1, b) = (e_2, b, I)$
$\delta(e_1, y) = (e_1, y, D)$ $\quad$ $\delta(e_3, y) = (e_3, y, I)$
$\delta(e_3, c) = (e_0, c, D)$

Considérese la palabra *xxyy* cuya trayectoria en la MT es como se muestra:



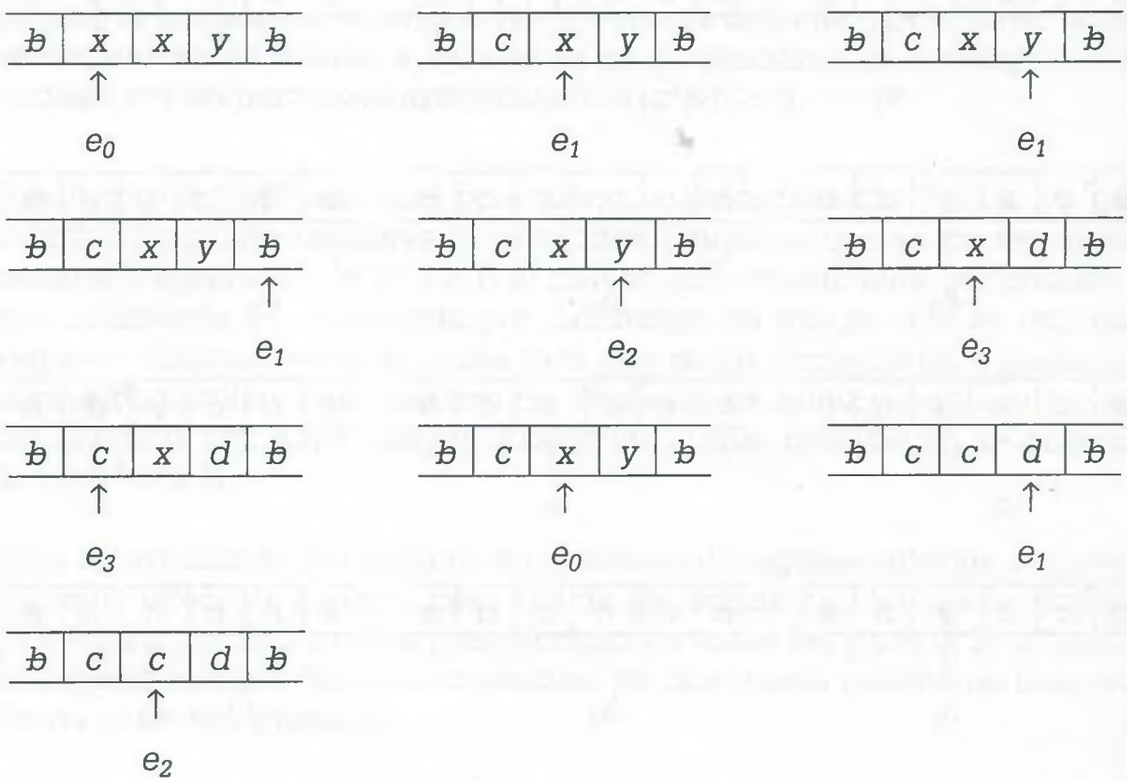
Una vez que se han cambiado todas las *x* por *c* y todas las *y* por *d*, regresamos al principio de la cadena con las siguientes composiciones adicionales:

$\delta(e_0, d) = (e_4, d, I)$
$\delta(e_4, c) = (e_4, c, I)$
$\delta(e_4, b) = (e_5, b, D)$



Si se requiere no perder la cadena inicial *xxyy*, ahora lo que se debe hacer es cambiar todas las *c* por *x* y todas las *d* por *y*, pero se puede observar que es posible que una MT acepte el lenguaje libre de contexto $L = \{x^n y^n / n \geq 1\}$.

Si ahora se desea probar si la palabra *xxy* pertenece al lenguaje con la MT anterior, lo que sucede es lo siguiente:



Pero $\delta(e_2, c)$ no está definida y el estado e_2 no es de aceptación, por lo tanto, la cadena *xxy* no pertenece al lenguaje $L = \{x^n y^n / n \geq 1\}$.

Lenguajes como $L = \{x^n y^n z^n / n \geq 1\}$ no son aceptados por autómatas de pila ya que las limitaciones de la pila hacen imposible el procesamiento. Cuando se guardan en la pila todas las *x*, para después sacar una a una cada vez que llega una *y*, hacen que cuando se desea comprobar si efectivamente el número de *x*, *y* y *z* son iguales, el registro de las dos primeras ya no existe, sin embargo una MT sí acepta este tipo de lenguajes, con lo cual se

ratifica la superioridad de las MT sobre los autómatas. Se pueden construir MT de una forma sencilla y para diferentes necesidades y no solamente en el reconocimiento de lenguajes. Hay máquinas de dos, tres o más cintas que fortalecen aún más la filosofía de las máquinas de Turing. Existen MT integradas por otras máquinas de Turing más sencillas, de tal manera que una MT es sin lugar a dudas, la equivalencia más cercana a una computadora que permite llevar a cabo procesos complicados, compuestos a su vez por actividades más simples, de la misma manera en que un programa está integrado por varios procedimientos o funciones, en el caso de la programación estructurada o bien, compuesto por una serie de métodos aplicables para diferentes situaciones, como ocurre en la programación orientada a objetos.

9.5 Teoría de la computabilidad

Es la parte de la computación que analiza y determina los problemas que pueden resolverse por medio de un algoritmo, o bien por alguna de las MT. Esto significa que no es factible resolver todos los problemas con la computadora, aun si se cuenta con la memoria (espacio) y tiempo ilimitados. La computabilidad se inicia en la década de 1930. En 1928 David Hilbert hizo patente la inquietud de crear un sistema matemático formal-complejo-consistente, capaz de probar la validez de todas las proposiciones matemáticas. Esto implicaba encontrar un algoritmo para determinar si todas las expresiones matemáticas conocidas son falsas o verdaderas. A este problema se le llamó *Entscheidungsproblem* (problema de decisión). La idea era que si se obtenía un algoritmo para determinar la validez de una proposición, no sería necesario seguir quebrándose la cabeza para determinar si una expresión matemática cualquiera es falsa o verdadera, porque con sólo someterla al algoritmo que Hilbert sugería sería suficiente para determinar su validez.

El problema de Entscheidungsproblem está relacionado con el *Teorema de la incompletitud*, de Kurt Gödel, quien dice que "Ningún teorema que contenga los teoremas de la aritmética con axiomas recursivamente enumerables puede ser consistente y completo a la vez", esto permite intuir que sí es posible tener algoritmos que permitan probar si algunas proposiciones matemáticas son verdaderas, pero es imposible obtener un algoritmo que permita determinar la falsedad o validez de todos los teoremas, como lo proponía Entscheidungsproblem.

En 1932 Alonzo Church utilizó una notación formal con la finalidad de transformar todas las fórmulas matemáticas a una forma estándar, de tal manera que la demostración de un teorema matemático pudiera probarse por medio de un algoritmo general, pero no fue posible la transformación de todos los teoremas matemáticos conocidos, sino solamente una parte de ellos que llamó funciones λ-*computables*.

Alonzo Church
(1903-1995)

Fue Matemático y lógico norteamericano creador de la base de la computación teórica. Nació en la ciudad de Washington, se diplomó en la Universidad de Princeton en 1924 y obtuvo su doctorado en 1927. Su obra más conocida es el desarrollo del cálculo lambda y su trabajo de 1936 que muestra la existencia de problemas indecibles. Este trabajo precedió el famoso trabajo de su alumno Alan Turing sobre el problema de parada que también demostró la existencia de problemas irresolubles por dispositivos mecánicos. Luego de revisar la tesis doctoral de Turing, demostraron que el cálculo lambda y la máquina de Turing utilizada para expresar el problema de parada tenían igual poder de expresión.

Posteriormente demostraron que una variedad de procesos mecánicos alternos para realizar cálculos tenían poder de cómputo equivalente y como resultado se postuló la tesis de Church-Turing.

Apoyándose en estudios de Herbrand sobre la recursividad, Gödel demuestra que una función *f* puede definirse como un conjunto de términos y símbolos matemáticos previamente definidos, incluyendo a la misma función *f*, dando origen a las funciones recursivas de Herbrand-Gödel.

En 1936 Church demuestra la equivalencia entre las funciones λ-*computables* y las funciones recursivas de Herbrand-Gödel, y establece que solamente estas funciones son factibles de probar por medio de un algoritmo, surgiendo de esta forma la *tesis de Church*, que establece que "La clase de funciones que se pueden calcular mediante un algoritmo, coinciden con las funciones recursivas". Con base en esto, Church mostró ejemplos de problemas no computables y demostró que el problema Entscheidungsproblem es no computable.

También Stephen Kleene coincide con Church al demostrar formalmente que las funciones recursivas de Herbrand-Gödel y las funciones λ-*computables* son equivalentes y da ejemplos de funciones no computables, usando para ello el concepto de función recursiva.

En ese mismo año Alan Turing publicó el trabajo "Números computables, una aplicación al Entscheidungsproblem", en donde Turing argumentó que dicho problema podría atacarse de un modo matemático y preciso por medio de sus MT, usando para ello un algoritmo. De esta forma postuló lo que hoy se conoce como la *Tesis de Turing*, que establece que "Las funciones que pueden ser calculadas mediante un algoritmo, son las funciones que pueden ser calculadas por medio de una Máquina de Turing", además demostró que el Entscheidungsproblem no era computable, y dio suficientes argumentos al respecto.

Church y Turing trabajaron en el mismo problema, pero en forma independiente, y llegaron a los mismos resultados en cuanto al Entscheidungsproblem y las conclusiones de ellos forman lo que hoy se conoce como la tesis de *Church-Turing* que establece que "Si una máquina de Turing no puede resolver un problema, ninguna otra computadora podrá hacerlo, puesto que no existe algoritmo para resolver dicho problema".

9.5.1 Teoría de la complejidad

La complejidad es la cantidad de recursos necesarios para resolver un problema, como son tiempo y espacio. El *tiempo* es el número de pasos de ejecución de un algoritmo para resolver un problema y el *espacio* la cantidad de memoria utilizada para resolver dicho problema. De tal manera que se puede decir que algoritmos que tardan mucho tiempo (días o meses) en ejecución o requieren gran cantidad de memoria principal (gigabytes) no es conveniente utilizarlos.

A pesar de que cada día se tienen computadoras más veloces, con mayor memoria principal y dispositivos con más espacio de almacenamiento secundario, es importante tener presente la inconveniencia de ejecutar algoritmos que no proporcionan resultados en un tiempo razonable, por tal razón los algoritmos se han clasificado, de acuerdo con el tiempo de ejecución, en dos tipos: polinomiales y no polinomiales.

Algoritmos polinomiales (P)

Prácticamente todos los algoritmos que se ejecutan en una computadora tienen una complejidad polinomial, es decir, la relación entre el tamaño del problema (número de datos o valor de *n*) y el tiempo de ejecución se puede encontrar por medio de una expresión polinomial. Los métodos para ordenar información (sorts) más comunes tienen una complejidad polinomial en donde el número de comparaciones que se deben hacer para ordenar un conjunto es un polinomio de segundo grado, como se muestra en la siguiente tabla:

Inicialmente	4 comparaciones	1ª. pasada	3 comparaciones	2ª. pasada	2 comparaciones	3ª. pasada	1 comparación	4ª. pasada	0 comparaciones
8		4		−1		−1		−1	
4		−1		4		2		2	
−1		8		2		4		4	
10		2		8		8		8	
2		10		10		10		10	

En cada paso el algoritmo lleva a cabo $(n - 1)$ comparaciones de tal manera que el total de comparaciones para ordenar el conjunto de datos resulta de multiplicar $(n - 1)$ por el número de datos: $n(n - 1) = n^2 - n$. Esto indica que el número de comparaciones está dado por un polinomio de grado 2, o bien $O(n^2)$. Es conveniente recordar que no todos los sorts tienen una complejidad polinomial, como ocurre con el quick sort. Por supuesto, el número exacto de comparaciones depende del algoritmo utilizado y la forma en que están colocados inicialmente los datos en el arreglo.

Se puede decir que aquellos algoritmos que tienen una complejidad de $O(n)$ y $O(Ln\ n)$ son algoritmos de complejidad lineal o logarítmica, ya que su crecimiento en cuanto al número de pasos para obtener resultados con un algoritmo determinado es menor que un algoritmo polinomial, como se muestra a continuación.

Algoritmos lineales (L)

Son aquellos algoritmos en donde el número de pasos está en relación directa con el número de datos. Un algoritmo que se utiliza para buscar un solo dato en un vector de *n* elementos, realizará en el peor de los casos *n* pasos.

A	80	–7	16	.....	43
	1	2	3	....	n

Por ejemplo, si se desea buscar el número 43 del vector anterior y considerando que se comienza la búsqueda a partir del dato que se encuentra en la primera posición, el algoritmo encontrará el dato, en el peor de los casos, en *n* iteraciones. Posiblemente el dato a buscar está antes, pero se toma en cuenta siempre el número máximo posible para obtener un resultado. En el caso anterior se dice que tiene una complejidad *n*, que es de orden *n* o bien O(*n*).

Complejidad logarítmica (*Ln n*)

Un algoritmo con una complejidad logarítmica es aquél en donde la información está segmentada en bloques de información, que a su vez también guardan un orden, como por ejemplo un directorio telefónico, en donde para buscar el teléfono de una persona cuyo apellido es "García", el algoritmo se remite al bloque de nombres donde sus apellidos comienzan con la letra "G", para de esa manera evitar pasar por los demás bloques. En este caso se dice que tiene una complejidad de orden O(*Ln n*).

No polinomiales (NP)

Son aquellos algoritmos en donde el número de pasos que se deben realizar para llegar al resultado tienen un crecimiento exponencial en relación con la cantidad de datos o valor de *n*. Esto ocurre con los algoritmos para encontrar combinaciones. Por ejemplo el número de combinaciones con repetición del 0 y el 1 en bloques de 3 posiciones es 2^3, como se muestra en la siguiente tabla:

Combinaciones
000
001
010
011
100
101
110
111

En forma generalizada, el número de combinaciones de 2 números y *n* posiciones es de complejidad 2^n, o bien complejidad exponencial.

En el caso anterior, si el valor de n es relativamente pequeño como $n = 150$, seguramente se tendrían problemas no solamente para encontrar 2^{150} combinaciones, sino que los recursos de la computadora no serían suficientes. Lo mismo pasaría para encontrar el número de combinaciones con repetición, de n números en bloques de n posiciones, en donde el crecimiento para obtener resultados es de $O(n^n)$, de tal manera que todos estos algoritmos, cuyo crecimiento en el número de pasos es exponencial, se agrupan como de complejidad NP. Lo mismo ocurre para encontrar el valor de $n!$, obtener los movimientos que se realizan para acomodar n discos con el algoritmo de las torres de Hanoi y la obtención de la expresión algebraica simplificada usando interpolación de Lagrange, en donde con un valor relativamente pequeño, el número de iteraciones y los recursos de la computadora son insuficientes.

Hay confusión para ubicar adecuadamente a los algoritmos en el grupo P o NP, ya que se dice que no es conveniente ejecutar algoritmos con complejidad NP, sin embargo, los algoritmos anteriores de combinaciones, factorial, torres de Hanoi y muchos más que no se citan aquí, corren sin problema en una computadora, pero solamente para un valor pequeño de n, de tal manera que este tipo de algoritmos son de complejidad NP "parcialmente computables" porque para valores medianamente grandes la computadora no puede con ellos, no solamente por el factor tiempo, sino también por el espacio.

9.6 Aplicación de los lenguajes formales

Las máquinas de estado finito y los AFD admiten *lenguajes regulares*, lenguajes simples que normalmente se observan funcionando en circuitos lógicos de control sencillos, en donde las operaciones a realizar están completamente determinadas por la información de entrada.

Considérese una máquina de estado finito expendedora de dos tipos de cerveza: Clara (C) y Obscura (O) de una misma marca. El costo de la cerveza es de $7.00 para ambos tipos de cerveza. La máquina sólo acepta monedas de 5, 2 y 1 pesos y está programada para dar cambio cuando sea necesario. Además, para seleccionar el tipo de cerveza cuenta con dos botones: Verde (V) para seleccionar cerveza clara y el botón naranja (N) para cuando se desea cerveza obscura.

En este caso la máquina de estado finito tiene los siguientes elementos:

$E = \{e_0, e_1, e_2, e_3, e_4, e_5, e_6, e_7\}$ (Conjunto de estados)
$A = \{1, 2, 5, V, N\}$ (Conjunto de entradas)
$B = \{1, 2, 5, C, O\}$ (Conjunto de salidas)

δ = La función para encontrar el estado siguiente, tiene como argumentos el par de elementos formados por un elemento del conjunto E y un elemento del conjunto A y arroja como resultado un elemento del conjunto E.

σ = La función de salida, tiene como argumentos el par formado por un elemento del conjunto E y un elemento del conjunto A y arroja como resultado un elemento del conjunto B.

De esta manera la tabla de estados queda de la siguiente manera:

Edo.	δ					σ				
	1	**2**	**5**	***V***	***N***	**1**	**2**	**5**	***V***	***N***
e_0	e_1	e_2	e_5	e_0	e_0	n	n	n	n	n
e_1	e_2	e_3	e_6	e_1	e_1	n	n	n	n	n
e_2	e_3	e_4	e_7	e_2	e_2	n	n	n	n	n
e_3	e_4	e_5	e_7	e_3	e_3	n	n	1	n	n
e_4	e_5	e_6	e_7	e_4	e_4	n	n	2	n	n
e_5	e_6	e_7	e_7	e_5	e_5	n	n	3	n	n
e_6	e_7	e_7	e_7	e_6	e_6	n	1	4	n	n
e_7	e_7	e_7	e_7	e_0	e_0	1	2	5	C	O

donde

n: Significa que la máquina de estado finito tiene salida nula (no regresa cambio ni se puede obtener de ella ninguna cerveza).

Suponer que una persona decide obtener de la máquina expendedora una cerveza obscura, para tal efecto deposita en la máquina dos monedas de $5.00, recibe un cambio de $3.00 y al presionar el botón N la persona recibe la cerveza obscura. Lo que ocurre es lo siguiente: inicialmente la máquina se encuentra en el estado e_0, cuando se hace el depósito de la moneda de $5.00 la función de estado siguiente y de salida generan los siguientes resultados

$$\delta(e_0, 5) = e_5 \qquad \sigma(e_0, 5) = n$$

lo cual significa que para el estado e_0 y una moneda de $5.00 el siguiente estado es e_5 y la salida es nula, aun si se presiona el botón N. Al depositar la segunda moneda de $5.00, se obtienen los siguientes resultados:

$$\delta(e_5, 5) = e_7 \qquad \sigma(e_5, 5) = 3$$

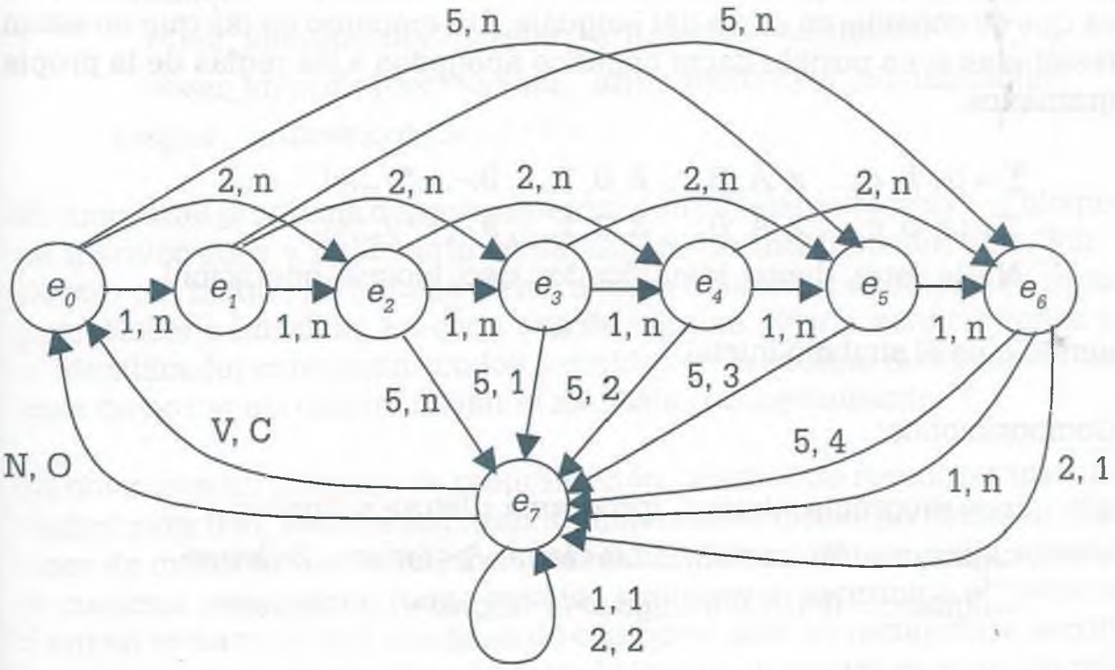
Esto significa que con la segunda moneda, la máquina se traslada al estado e_7 y proporciona un cambio de $3.00 (conformado por una moneda de

$1.00 y una de $2.00, o bien tres monedas de $1.00, ya que son los únicos tipos de monedas válidas).

En este estado si la persona presiona el botón *N*, obtendrá la cerveza obscura, o bien obtendrá una cerveza clara si presiona el botón *V* y la máquina de estado finito regresará al estado inicial.

$$\delta(e_7, N) = e_0 \qquad \sigma(e_7, N) = O$$

El diagrama de estados para este tipo de máquina de estado finito es como sigue:



Los *lenguajes libres* de contexto, por su parte, se pueden usar en la creación de intérpretes y compiladores que posteriormente se usarán en computadoras para crear programas, para lo cual es necesario llevar a cabo primeramente un buen análisis que contemple todos los elementos propios de un lenguaje de programación formal, como es el tipo de alfabeto (Σ), conjunto de símbolos no terminales (N), conjunto de símbolos terminales (T) y las composiciones que permitirán derivar las palabras válidas del propio lenguaje. Por ejemplo, supóngase que se desea desarrollar un intérprete para reconocer programas como el siguiente:

Programa nombre;

Variables

a,b,x: **entero;**

Inicio

Leer a;

Leer b;

x=3*a–b;

Imprimir x;

Fin.

Las palabras en negritas se consideran reservadas y no podrán cambiar, ya que se consideran parte del lenguaje, sin embargo en las que no están resaltadas sí es posible hacer cambios apegados a las reglas de la propia gramática.

Σ = {*a, b, c,..., z, A, B,..., Z*, 0, 1,..., 9,–,+,*,/,;,=}
T = {*a, b, c,..., z, A, B,..., Z*, 0, 1,..., 9,–,+,*,/,;,=}
N = {$\in$ letra, dígito, identificador, tipo, bloque, operación}

donde α es el símbolo inicial.

Composiciones:

α::= programa <*letra*>; /programa <*letra*> <*dígito*>;
<*letra*>::=a/b/...z/A/B/...Z/a<*letra*>/b<*letra*>...Z<*letra*>
<*dígito*>::= 0/1/...9/0<*dígito*>/1<*dígito*>/...9<*dígito*>

Esta parte permite observar que todo programa aceptado por este lenguaje debe comenzar con la palabra reservada "programa", seguida de un espacio, una letra y una cadena de letras o números, para terminar con un punto y coma.

α::= variables <*identificador*>:<*tipo*>; /variables <*letra*>:<*tipo*>;
<*identificador*>::= <*letra*>/<*letra*> <*digito*>, /<*letra*>,
<*letra*>::= a/b/...z/A/B/...Z/a<letra>/b<*letra*>.../Z<*letra*>
<dígito>::= 0/1/...9/0<dígito>/1<dígito>/...9<dígito>
<*tipo*>::= entero/real

El área donde se definen las variables deberá comenzar con la palabra "variables", seguida de los identificadores separados por comas. Después de la lista de identificadores se colocan dos puntos y el tipo de dato, que puede ser "entero" o "real", para terminar con punto y coma.

A todos los identificadores se les asigna una dirección de memoria principal, no solamente para guardar su valor cuando se lleven a cabo las operaciones, sino también para determinar si algún identificador no se encuentra definido y se pretende usar como variable en el cuerpo del programa.

α::= Inicio <*bloque*> Fin.

<*bloque*>::= Leer <*ver_identificador*>; Hacer <*ver_identificador*> = <*operación*>; / Imprimir <*ver_identificador*> / Leer <*ver_identificador*><*bloque*>; /Hacer <*ver_identificador*> = <*operación*><*bloque*>; / Imprimir <*ver_identificador*> <*bloque*>;

<*operación*>::= /<*digito*>/ <*digito*><*signo _ aritmético*><*ver_identificador*>

/<*ver_identificador*><*signo _ aritmético*><*digito*>

/<*ver_identificador*><*signo _ aritmético*><*ver_identificador*>

<*signo _ aritmético*>::= – / * / + / /

El cuerpo del programa deberá comenzar con la palabra "Inicio", el bloque de instrucciones y finalmente se deberá cerrar dicho bloque con "Fin". Dentro del bloque se pueden llevar a cabo diferentes operaciones como Leer, Hacer e Imprimir. En cada uno de ellos se deberá verificar antes si el identificador concuerda con los definidos en la sección de variables, en caso de no ser así deberá enviar el mensaje correspondiente.

Es obvio que un lenguaje de programación, además de reconocer instrucciones para leer, hacer e imprimir, también tiene ciclos, se manejan más tipos de datos, se tienen funciones estándar propias del lenguaje, maneja la memoria principal de forma estática y dinámica, determina el mensaje a enviar en caso de que una línea de código no esté correctamente escrita y varios elementos más. Sin embargo, la intención en este caso es dar una somera idea de los elementos mínimos a considerar en el diseño y desarrollo de un traductor, intérprete o compilador, que a su vez son aplicaciones ilustrativas de los lenguajes libres de contexto.

Los *lenguajes sensibles* al contexto se usan para llevar a cabo actividades más complejas, donde ya se han tenido algunos resultados, pero también en donde hay mucho que hacer, como es el área de inteligencia artificial (IA). Recordar que la inteligencia artificial es la ciencia que se dedica al desarrollo de programas para máquinas y computadoras, cuya finalidad es obtener un comportamiento y procesamiento de información semejante a la del ser humano, en otras palabras, la IA tiene como objetivo dotar a la computadora para que tome decisiones inteligentes, considerando diferentes elementos que no necesariamente se programan, sino que se deducen y aprenden. La inteligencia artificial abarca varias ramas como *sistemas expertos*, cuya finalidad es procesar una gran cantidad de información y con base en ello emitir una conclusión como lo haría un experto

en medicina, economía, política, etc. En las *Redes neuronales* la computadora aprende en función de las vivencias y situaciones que se le presentan, de la misma manera que lo hace el ser humano a lo largo de su vida. *Computación evolutiva*, como es el caso de algoritmos genéticos, en donde se determina la mejor opción de un gran volumen de respuestas posibles, asemejando con eso la supervivencia del más fuerte, como ocurre en la vida de todo ser vivo (de acuerdo con la teoría de Darwin), en donde solamente sobrevive el más fuerte. Pero también la IA trabaja en *reconocimiento de patrones*, como puede ser el reconocimiento de escritura y habla, en donde los lenguajes sensibles al contexto tienen aplicación cuando se desea relacionar palabras con sonidos semejantes, pero que su significado es diferente, construir palabras complejas a partir de palabras simples, estructurar frases y oraciones, y la determinación de su significado cuando están inmersas en contextos diferentes. El tratar de que el ser humano tenga una comunicación con la computadora de la misma manera a como lo hace con sus semejantes, es un ejemplo típico de la aplicación de los lenguajes sensibles al contexto.

9.7 Resumen

Un lenguaje es un conjunto de símbolos y métodos para estructurar y combinar dichos símbolos. Un lenguaje también recibe el nombre de idioma,el cual a su vez es un medio de comunicación. Esta clase de lenguaje recibe el nombre de *lenguaje natural.*

Existen lenguajes de menor capacidad para simular y modelar lenguajes naturales, como el lenguaje Binario, Java, C, Basic o Pascal, que se utilizan en la computación. A este tipo de lenguajes se les llama *lenguajes formales.*

Un lenguaje (L(G)) está basado en la gramática y las reglas o métodos para la creación de palabras propias del lenguaje. Las gramáticas están integradas por varios elementos que permiten la estructuración de palabras. La gramática G = {Σ, N, T, s, c} es el sistema que permite establecer las reglas que han de aplicarse a un lenguaje. Aquí se tiene que:

Σ: es el alfabeto o conjunto de símbolos con el cual se forman palabras de un lenguaje.

N: conjunto de símbolos no terminales en un lenguaje.

T: conjunto de símbolos terminales.

s: estado inicial.

c: conjunto de composiciones o reglas que se deben usar para la estructuración de las palabras válidas en el lenguaje.

Las gramáticas se pueden clasificar como:

Tipo 0: si no se pone ninguna restricción a las composiciones de G.

Tipo 1: si para cualquier composición $d_1 \rightarrow d_2$ de la gramática G, la longitud de símbolos de la izquierda de la composición (d_1) es menor o igual a la longitud de símbolos de la derecha (d_2).

Tipo 2: Si el lado izquierdo de cada composición es un símbolo no terminal y el lado derecho consta de uno o más símbolos terminales y/o no terminales.

Tipo 3: Si el lado izquierdo de la composición es un símbolo no terminal y el lado derecho tienen uno o más símbolos incluyendo a lo más un símbolo no terminal.

A las gramáticas 0 y 1 también se les conoce como *sensibles al contexto*. Este tipo de gramáticas son muy difíciles de analizar y estudiar, además de que es muy poco lo que se sabe acerca de ellas debido principalmente a que se tiene mucha libertad para la formación de palabras. La gramática tipo 2 recibe el nombre de gramática *libre de contexto*. Las gramáticas libres de contexto son las que se usan actualmente para la creación de lenguajes formales. La gramática tipo 3 o también llamada *regular* tiene reglas muy simples para la generación de palabras de un lenguaje. Por sus características una gramática regular es a su vez una gramática libre de contexto y una gramática sensible al contexto, de la misma manera que una gramática libre de contexto es una gramática sensible al contexto.

Las gramáticas regulares se usan en la operación de sistemas sencillos como juegos electrónicos y máquinas expendedoras, se pueden representar como autómatas finitos y máquinas de estado finito. Las gramáticas libres de contexto son las más usadas en la elaboración de compiladores, traductores e intérpretes, se pueden representar por medio de árboles de derivación, representación BNF y diagramas sintácticos.

Se dice que un autómata finito es determinístico si por medio de la función de transición δ: $E \times \Sigma \rightarrow E$ es posible determinar claramente cuál es el estado siguiente. Sin embargo, en un autómata finito no determinístico la función de estado siguiente, no conduce a un estado único determinado.

Una máquina de estado finito es una forma especial de representar los autómatas finitos, en donde no existen estados aceptados y donde los símbolos de salida se colocan juntamente con los símbolos de entrada en cada una de las aristas de la máquina. Una máquina de estados finitos $M = \{E, A, B, s, \delta, \sigma\}$ se define como un sistema que cuenta con un número finito de estados E, que acepta un conjunto finito de entradas A y que puede tener un número finito de salidas B, tiene un estado inicial s. Cuenta además con dos funciones: δ también llamada "función de estado si-

guiente" y se define con δ: $E \times A \rightarrow E$ y la "función de salida" σ que está definida como σ: $E \times A \rightarrow B$.

Una máquina de Turing (MT) consiste de una cinta que se extiende de manera infinita, en donde se escribe o se lee información por medio de una cabeza de lectura-escritura. La MT es un conjunto de elementos MT = (E, Σ, A, s, ƀ, F, δ) en donde:

E: Conjunto finito de estados.

Σ: Conjunto de símbolos de entrada.

A: Alfabeto de la cinta.

s: estado inicial $s \in E$.

ƀ: Símbolo blanco donde. ($ƀ \in A$; $ƀ \notin \Sigma$).

F: Conjunto de estados de aceptación.

δ: Función de transición en donde δ: $E \times A \rightarrow E \times A \times \{D,I\}$.

La *teoría de la computabilidad* es la parte de la computación que analiza y determina los problemas que pueden resolverse por medio de un algoritmo o bien por alguna de las MT. Esto significa que no todos los problemas son factibles de resolverse usando para ello la computadora, aún si se cuenta con la memoria y tiempo ilimitados.

La *teoría de la complejidad* es la cantidad de recursos necesarios para resolver un problema como son tiempo y espacio. El *tiempo* es el número de pasos de ejecución de un algoritmo para resolver un problema y el *espacio* la cantidad de memoria utilizada para resolver dicho problema.

A pesar de que cada día se tienen computadoras más veloces, con mayor memoria principal y dispositivos con más espacio de almacenamiento secundario es importante tener presente la inconveniencia de ejecutar algoritmos que no proporcionan resultados en un tiempo razonable. O bien problemas que no son computables usando para ellos los lenguajes formales actuales, de tal manera que se tiene el reto futuro de diseñar algoritmos cada vez más eficientes y computadoras con mejores recursos que las actuales.

9.8 Problemas

9.1 En cada uno de los incisos determinar si la gramática dada es regular, libre de contexto y/o sensible al contexto. Explique su respuesta.

a) Sea:

$T = \{x, y, z\}$ $N = \{s, A, B, C\}$

Composiciones:

$s \rightarrow xs$	$A \rightarrow xA$	$B \rightarrow xA$	$C \rightarrow xC$	$C \rightarrow y$
$s \rightarrow zB$	$A \rightarrow yC$	$B \rightarrow yC$	$C \rightarrow zA$	$A \rightarrow y$
$s \rightarrow yA$	$A \rightarrow zB$	$B \rightarrow zs$	$C \rightarrow yB$	$s \rightarrow z$

b) Sea:

$T = \{x, y, z\}$ $N = \{s, A, B\}$

Composiciones:

$s \rightarrow yA$	$A \rightarrow xBA$	$B \rightarrow Bx$
$BxA \rightarrow AzB$	$xAz \rightarrow CB$	$B \rightarrow Ayy$
$A \rightarrow yz$	$Cy \rightarrow zz$	$B \rightarrow xx$

c) Sea:

$T = \{x, y\}$ $N = \{s, A, B, C\}$

Composiciones:

$S \rightarrow xB$	$B \rightarrow yxC$	$C \rightarrow BxA$
$A \rightarrow xBy$	$B \rightarrow yyC$	$C \rightarrow xy$
$A \rightarrow CBx$	$B \rightarrow x$	$A \rightarrow y$

9.2 En cada uno de los incisos determinar si la gramática dada es regular, libre de contexto y/o sensible al contexto. Explique su respuesta.

a) Sea:

$T = \{x, y, z\}$ $N = \{s, A, B, C, D\}$

Composiciones:

$S \rightarrow BxA$	$B \rightarrow yCx$	$C \rightarrow BA$	$C \rightarrow yx$
$AA \rightarrow zBC$	$Ay \rightarrow xy$	$Bx \rightarrow DyA$	$BA \rightarrow yzz$
$Dyz \rightarrow CD$	$DB \rightarrow xAC$	$CC \rightarrow yyzx$	$Cxx \rightarrow zyB$

b) Sea:

$T = \{x, y\}$ $N = \{s, A, B, C\}$

Composiciones:

$s \rightarrow yA$	$A \rightarrow xAB$	$A \rightarrow Bxx$	$A \rightarrow yx$
$B \rightarrow xBy$	$B \rightarrow Ayy$	$C \rightarrow x$	
$A \rightarrow xC$	$B \rightarrow yC$	$B \rightarrow y$	

c) Sea:

$T = \{0, 1\}$ $N = \{s, A\}$

Composiciones:

$S \rightarrow 0A1$	$A \rightarrow 0A1$	$A \rightarrow 01$	$A \rightarrow 10$
$S \rightarrow 1A0$	$A \rightarrow 1A0$	$A \rightarrow 10A$	$A \rightarrow 01A$

9.3 En cada una de las gramáticas del problema 9.1 realizar lo siguiente:

a) Con la gramática del inciso a):

- Derivar las palabras: *zyxzzzz*, usando para ello las composiciones y también por medio de un árbol de derivación.
- Representar dicha gramática con notación BNF.

b) Con la gramática del inciso b):

- Derivar la palabra: *yzzzyyxx* usando para ello las composiciones.
- Representar dicha palabra con notación BNF.

c) Con la gramática del inciso c):

- Derivar la palabra: *xyyyyxyxxyxx* usando para ello las composiciones.
- Por medio de un árbol de derivación, probar que la palabra *xyyyyxyxxyxx* pertenece al lenguaje.
- Representar dicha gramática con notación BNF.
- Representar la gramática por medio de diagramas sintácticos.

9.4 En cada una de las gramáticas del problema 9.2 realizar lo siguiente:

a) Con la gramática del inciso a):
- Derivar usando composiciones las palabras siguientes: *zyyyzx, zyyyxxyyzx* con la finalidad de determinar si pertenecen al lenguaje.
- Representar dicha gramática con notación BNF.

b) Con la gramática del inciso b):
- Derivar usando para ello composiciones la palabra: *yxxxxyxxyyy*.
- Por medio de un árbol de derivación, probar que la palabra: *yxxxxyxxyyy* pertenece al lenguaje.
- Representar dicha gramática con notación BNF.
- Representar la gramática usando diagramas sintácticos.

c) Con la gramática del inciso c):
- Derivar usando composiciones las palabras: 010011, 11110000, 01011010
- Por medio de árboles de derivación, probar que las palabras 010011, 11110000, 01011010 pertenecen al lenguaje.
- Representar dicha gramática con notación BNF.
- Representar la gramática con diagramas sintácticos.

9.5 Sean:

$\Sigma = \{0, 1\}$; $L = \{\varepsilon, 0, 1, 00, 10, 111\}$; $M = \{1, 00, 01, 100, 111, 000\}$. Encontrar:

a) $L \cup M$	**d)** $L - M$	**g)** L^2	**j)** L'
b) $L \cap M$	**e)** M^I	**h)** L^+	
c) LM	**f)** L^I	**i)** L^*	

9.6 Sean:

$\Sigma = \{0, 1\}$; $L = \{0, 11, 100\}$; $M = \{0, 10, 11, 100, 101\}$. Encontrar:

a) L^I	**d)** M^+	**g)** M^3	**j)** LM
b) M^I	**e)** M^I	**h)** $(L \cup M^I)$	**k)** M'
c) L^+	**f)** L^2	**i)** $(M - L)$	**l)** $(M')^I$

9.7 Sean:

$\Sigma = \{a, b, \ldots, z\}$; L = {días}. Encontrar:

a) L^+ **d)** $(L^+)^2$ **g)** $(L^*)^I$ **j)** $(L')^I$
b) L^* **e)** $(L^+)^I$ **h)** $(L^I)^*$ **k)** $(L^*)'$
c) L^I **f)** $(L^I)^+$ **i)** L' **l)** $(L^+)^I$

9.8 Sean: $\Sigma = \{0, 1\}$; L = {110}. Encontrar:

a) L^* **d)** L^4 **g)** $(L^I)^+$ **j)** $(L^*)^*$
b) L^+ **e)** $(L^I)^3$ **h)** $(L^+)^*$ **k)** $(L^*)'$
c) L^I **f)** $(L^*)^I$ **i)** $(L^+)^+$ **l)** $(L^+)'$

9.9 Sean M y L lenguajes regulares sobre Σ. Demostrar que los siguientes lenguajes son equivalentes:

a) $(LM^*)^I \cup (M^0 \cup M^+)^I L^I = (M^*)^I L^I$
b) $((\varepsilon \cup M)^*((L^+)^+ \cup LL^*))^I = (L^+)^I (M^*)^I$

9.10 Sean M y L lenguajes regulares sobre Σ. Demostrar que los siguientes lenguajes son equivalentes:

a) $((\varnothing^* \cup L)^*)^*((M^+)^0 \cup (M^0 \cup MM^*)) = L^*M^*$
b) $(M^+)^I (L^*)^I \cup (((L^+)^+ \cup (L^+)^0)(M^*M \cup L^0)M)^I = (L^*M^+)^I$

9.11 Sean *r*, *s* y *t* expresiones regulares sobre Σ. Probar que las siguientes equivalencias de expresiones regulares son ciertas:

a) $(t^*t^* \cup r\varnothing)(s^*s \cup \varnothing^*) = t^*s^*$
b) $t\varnothing^*t^* \cup r\varepsilon t^*t^* = t^+ \cup r^*t$
c) $(t^* \cup \varnothing s^*)(t \cup r) = t^+ \cup t^*r$

9.12 Sean *r*, *s* y *t* expresiones regulares sobre Σ. Probar que las siguientes equivalencias de expresiones regulares son ciertas:

a) $(\varepsilon \cup r^+)(tr^*r \cup t\varnothing^*)^* = (r \cup t)^*$
b) $(\varepsilon \cup ss^*) \cup s\varnothing^*s^*tr = s^* \cup s^+tr$
c) $(r^*s)^*s^* \cup (r \cup s)^*t = (r \cup s)^*(s^* \cup t)$

9.13 Sean los lenguajes L = {*ab*, *baa*}; M = {*a*, *ab*, *bb*} sobre el alfabeto $\Sigma = \{a, b\}$. ¿Cuáles son los elementos de los siguientes lenguajes regulares?

a) $L^3 \cup L$
b) $(M \cup L)\{\varepsilon\}$
c) LM^2
d) $(LM)^2$

9.14 Sean los lenguajes L = {0, 00}; M = {10, 101} sobre el alfabeto $\Sigma = \{0, 1\}$. ¿Cuáles son los elementos de los siguientes lenguajes regulares?

a) $(L \cup M)\{\varepsilon\}^2$
b) M^2L
c) M^*
d) $(LM)^2$
e) L^3L^2
f) $\varnothing M^*$
g) $M \cup \varnothing^*$

9.15 Sea $\Sigma = \{a, b\}$. Para cada uno de los siguientes incisos:

a) Cadenas de caracteres con al menos una letra *b*.
b) Cadenas de caracteres con al menos dos letras *a*.
c) Cadenas de caracteres que comienzan con *aab*.
d) Cadenas de caracteres que comienzan con *ab* y terminan con *ba*.
e) Cadenas de caracteres en donde toda *a* está seguida de una *b*, tiene solamente *bs* o se trata de la cadena vacía.

Encontrar:

- La expresión regular.
- El diagrama de transición del autómata finito para representar dicha expresión regular.
- La tabla de transición.
- Los elementos de los conjuntos *E*, *F* y el estado inicial *s*.

9.16 Sea $\Sigma = \{0, 1\}$. Para cada uno de los siguientes incisos:

a) Cadenas de caracteres que comienzan con 100 (100, 10001, 100000, 1001110,...).

b) Cadenas de caracteres con un número par de unos (11, 00101, 1000100101,...).

c) Cadenas de caracteres en donde se concatena *i* veces la cadena 01 $\{(01)^i / i > 0\}$, (01, 0101, 01010101,...).

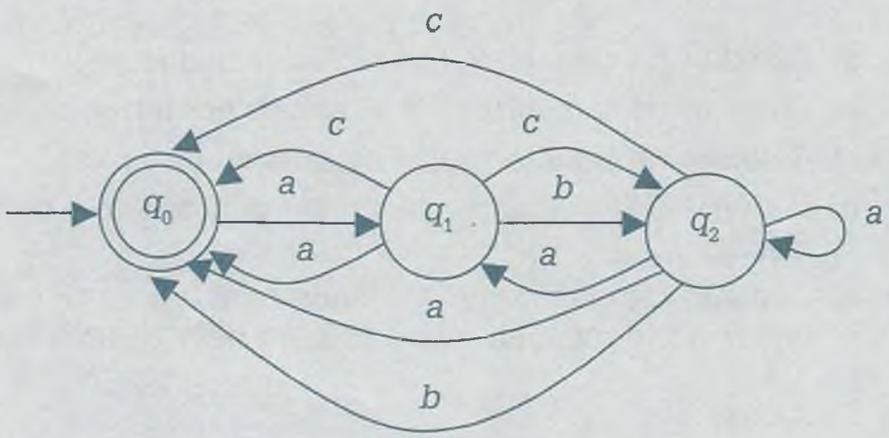
d) Cadenas de caracteres en donde todo 1 esté entre dos ceros (010, 0010100, 010010, 01010).

e) Cadenas de caracteres en donde nunca aparezcan las subcadenas 00 o 11.

Encontrar:

- La expresión regular.
- El diagrama de transición del autómata finito para representar dicha expresión regular.
- La tabla de transición.
- Los elementos de los conjuntos *E*, *F* y el estado inicial *s*.

9.17 Sea $\Sigma = \{a, b, c\}$ y el diagrama de transición del AFN.



a) Encontrar la tabla de transición del AFN, los elementos de los conjuntos *E* y *F*, además el estado inicial *s*.

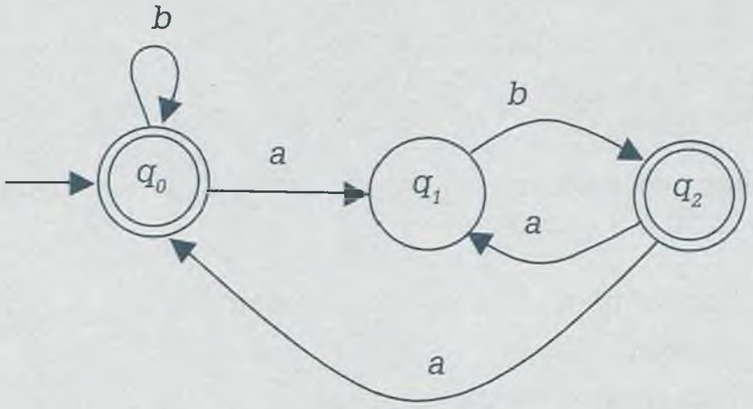
b) Convertir el AFN a un AFD.

c) ¿Cuáles son los elementos de los conjuntos *E* y *F*?, y ¿cuál es el estado inicial del AFD?

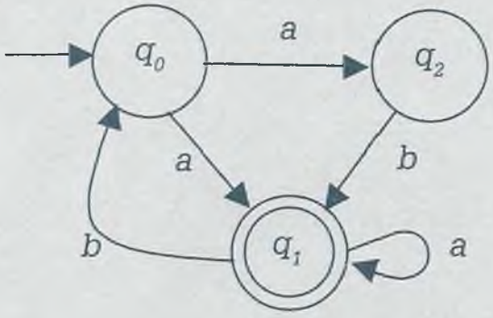
9.18 Sea $\Sigma = \{a, b\}$ para el AFN de cada inciso.

- Encontrar la tabla de transición del AFN, los elementos de los conjuntos E y F, además el estado inicial s.
- Convertir el AFN a un AFD.
- ¿Cuáles son los elementos de los conjuntos E y F?, y ¿cuál es el estado inicial del AFD.

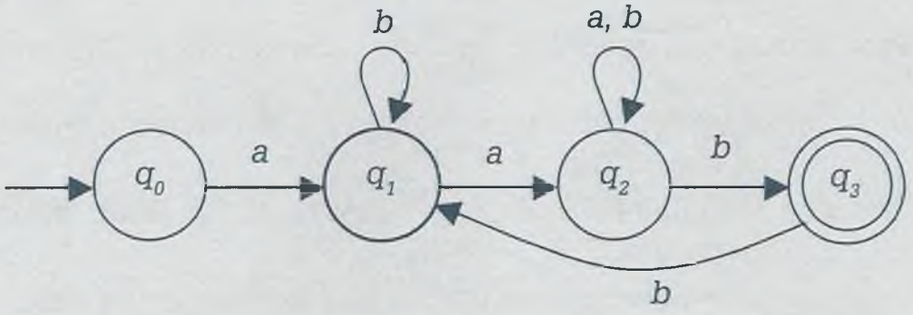
a)



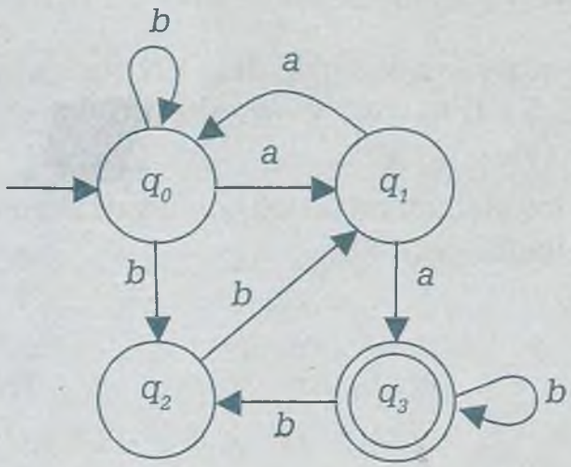
b)



c)



d)



9.19 Crear una máquina de estado finito, para restar dos cantidades en binario y encontrar:

a) Los elementos de los conjuntos *E*, *A* y *B*.
b) El estado inicial *s*.
c) El diagrama de transiciones.
d) La tabla de transiciones para las funciones de estado siguiente δ y salida δ.
e) La resta de $10001011_{(2)} - 1101101_{(2)}$.

9.20 Encontrar:

- Los elementos de los conjuntos *E*, *A* y *B*.
- El estado inicial *s*.
- El diagrama de transiciones.
- La tabla de transiciones para las funciones de estado siguiente δ y salida δ.

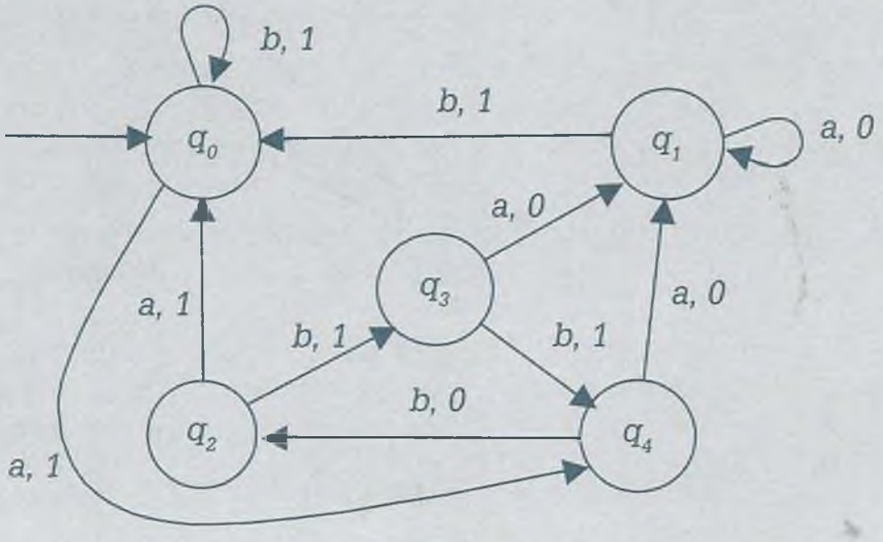
Para la máquina de estado finito de cada uno de los siguientes incisos:

a) Máquina para sumar dos cantidades en octal. Probar la máquina con $73012_{(8)} + 26347_{(8)}$.
b) Máquina para restar dos cantidades en octal. Probar la máquina para $60054_{(8)} - 7346_{(8)}$.
c) Máquina para multiplicar dos cantidades de base 5. Probar la máquina para $2304_{(5)} \times 32_{(5)}$.
d) Máquina para dividir dos cantidades de base 7. Probar la máquina para $560134_{(7)} \div 351_{(7)}$.

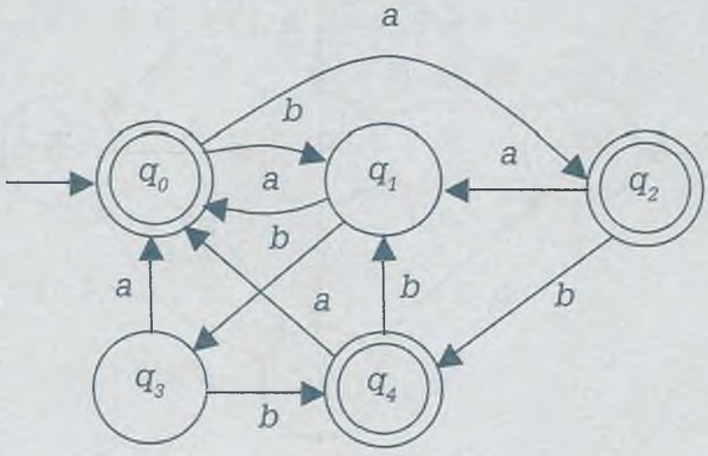
9.21 En cada uno de los siguientes incisos encontrar:

- El AF o la máquina de estado finito equivalente (según el caso).
- La gramática que corresponde.

a)



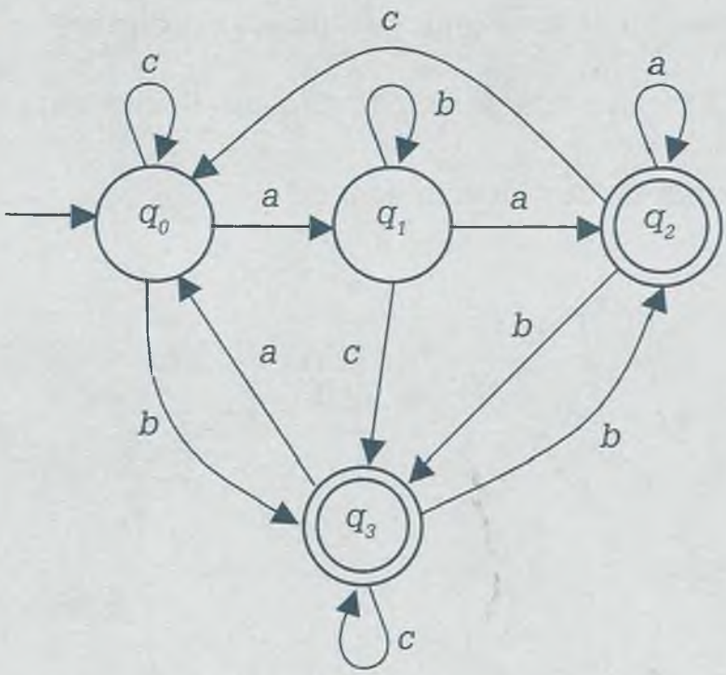
b)



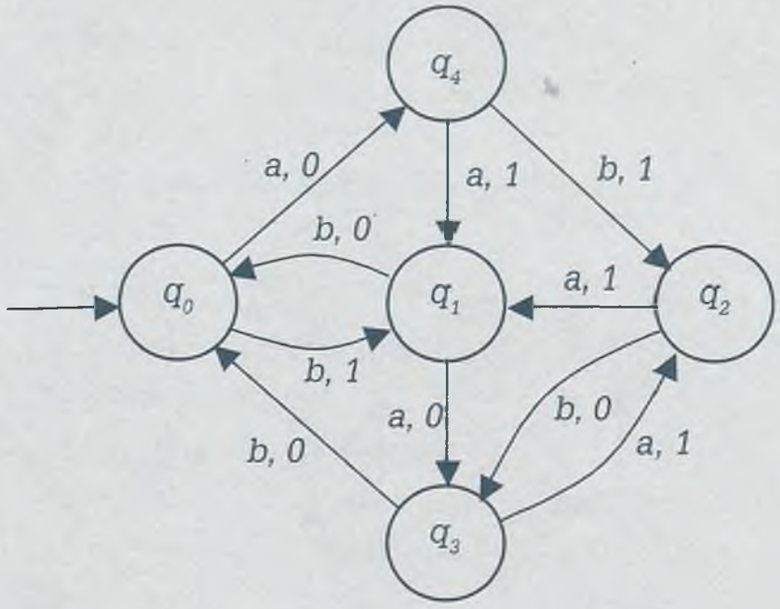
9.22 En cada uno de los siguientes incisos encontrar:

- El AF o la máquina de estado finito equivalente (según el caso).
- La gramática que corresponde.

a)



b)



9.23 Construir una MT para cada uno de los siguientes incisos:

a) Que acepte el siguiente lenguaje $L = \{x^n y^m / n \geq 1;\ m \geq 1\}$. Probar la MT para las palabras *xxy*, *xxyx*.

b) Que acepte palabras con un número par de *x*. Ejemplo: *xx*, *xxxx*, *xxxx*,.... Probar la MT para *xxxx* y *xxx*.

c) Que acepte palabras para el lenguaje L cuya expresión regular es $(x \cup y)^* z$. Esto implica que puede aceptar palabras que terminan con *z*, (*z*, *xz*, *yz*, *xyz*, *yxz*, *xxz*, *yxxz*,...). Probar la MT para las palabras *xyyz*, *xzz*.

9.24 Construir una MT para cada uno de los siguientes incisos:

a) Que acepte el siguiente lenguaje L = $\{yx^nzy\ n \geq 0\}$. Probar la MT para las palabras *yzy, yxxzy, yzyz*.

b) Que acepte palabras para el lenguaje L cuya expresión regular es $zx(z \cup y)^*y$. Esto implica que puede aceptar palabras que comienzan con *zx* y terminan con *y*. Probar la MT para: *zxy, zxxyy, zxyyx*.

c) Que acepte el siguiente lenguaje L = $\{x^ny^n\ n \geq 1\}$. Cambiar las *x* por *c* y las *y* por *d*, pero además regresar la cadena a sus símbolos originales en caso de ser aceptada la cadena como parte del lenguaje L.

d) Que acepte palabras para el lenguaje L cuya expresión regular es $x^+(y \cup x)^*z$.

Nota: En todos los casos la MT tiene inicialmente la cabeza de lectura-escritura en el primer símbolo de la cadena y debe terminar en ese mismo punto.

Bibliografía

[1] Brookshear, J. Glenn. "Teoría de la computación". Addison-Wesley Iberoamericana. Impreso en Estados Unidos. (1993).

[2] Grassmann, Winfried Karl. Tremblay, Jean-Paul. "Matemática Discreta y Lógica, una perspectiva desde la ciencia de la computación". Prentice Hall. Impreso en España. (1997)

[3] Grimaldi, Ralph. "Matemáticas discreta y combinatoria". Addison-Wesley Iberoaméricana. Impreso en México. (1989).

[4] Hopcroft, John E. Ullman Jeffrey D. "Introducción a la teoría de Autómatas Lenguajes y Computación". CECSA. Impreso en México. (1996).

[5] Jiménez Murillo, José Alfredo. "Matemáticas básicas para computación". Tecnológico de Morelia. (1996).

[6] Johnsonbaugh, Richard. "Matemáticas discretas". Grupo Editorial Iberoamérica. Impreso en México. (1988)

[7] Kelley, Dean. "Teoría de autómatas y lenguajes formales". Prentice Hall. Impreso en España. (1995).

[8] Kolman, Bernard. Busby, Robert C. Ross, Sharon. "Estructuras de Matemáticas Discretas para la Computación". Prentice Hall. 3ª Edición. Impreso en México. (1997).

[9] Liu, C. L. "Elements of Discrete Mathematics". 2ª Edición. Mc Graw Hill. Printed in Singapore. (1985).

[10] Martin, Jhon C. "Lenguajes formales y teoría de la computación". 3ª Edición. Mc Graw Hill. Impreso en México. (2004).

[11] Ross, Kenneth A. Wright Charles R. B. "Matemáticas Discretas" 2ª Edición. Prentice Hall. Impreso en México. (1990).

[12] Scheinerman, Edward R. "Metmáticas Discretas". Thomson Learning. Impreso en México. (2001)

[13] Tremblay, Jean-Paul. Manohar, Ram. "Matemáticas Discretas". CECSA. Impreso en México. (1986).

Respuestas de problemas seleccionados

Capítulo 1 Sistemas numéricos

1.1.

a)

001	001	000	111	010	100	100	010	.	010	$100_{(2)}$	a octal.
1	**1**	**0**	**7**	**2**	**4**	**4**	**2**	**.**	**2**	$\mathbf{4_{(8)}}$	**Resultado.**

b)

4	E	C	7	.	B	$5_{(16)}$
0100	**1110**	**1100**	**0111**	**.**	**1011**	**0101** $_{(2)}$ **Resultado**

c)

4	7	5	3	2	0	.	4	$7_{(8)}$
100	111	101	011	010	000	.	100	$111_{(2)}$ Binario
	0010	0111	1010	1101	0000	.	1001	$1100_{(2)}$ Separando en bloques de cuatro bits y completando con ceros en los extremos
	2	**7**	**A**	**D**	**0**	**.**	**9**	$\mathbf{C_{(16)}}$ Resultado

d)

			3	2	F	E	6	8	5	.	9	$C_{(16)}$	
			0011	0010	1111	1110	0110	1000	0101	.	1001	$1100_{(2)}$	
0	011	001	011	111	110	011	010	000	101	.	100	111	$000_{(2)}$
	3	**1**	**3**	**7**	**6**	**3**	**2**	**0**	**5**	**.**	**4**	$\mathbf{7_{(8)}}$	Resultado

1.3.

a) $730568.23_{(9)} = 433493.2593_{(10)} = B3D9B.38B5_{(14)}$

b) $6G5A.23_{(20)} = 54510.1075_{(10)} = 1101010011101110.0001_{(2)}$

c) $4A7E8.52_{(18)} = 480752.284_{(10)} = 976A2.43D4_{(15)}$

d) $93AF5.36_{(17)} = 769578.1972_{(10)} = 20C394.2744_{(13)}$

e) $558C5.3G_{(18)} = 556853.216_{(10)} = 1G6I5.54AF_{(24)}$

1.5.

a)

$$\begin{array}{rcccccccccccc} & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & . & 0 & 1 & 1 & 1_{(2)} \\ + & & 1 & 1 & 0 & 1 & 0 & 1 & 1 & . & 1 & 1 & 0 & 1_{(2)} \\ \hline 1 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & . & 0 & 1 & 0 & 0_{(2)} \end{array}$$

b)

$$\begin{array}{rcccccccccc} & 3 & A & 5 & 6 & 7 & B & . & 1 & 2_{(13)} \\ + & 9 & C & 0 & 1 & 7 & 2 & . & 3 & 4_{(13)} \\ \hline 1 & 0 & 9 & 5 & 8 & 2 & 0 & . & 4 & 6_{(13)} \end{array}$$

c)

$$\begin{array}{rcccccccccc} & 4 & 2 & 0 & 6 & 1 & 2 & 3 & 1 & . & 3 & 2 & 5_{(7)} \\ + & & 5 & 0 & 1 & 4 & 2 & 3 & 2 & . & 0 & 3_{(7)} & \\ \hline 5 & 0 & 1 & 0 & 5 & 4 & 6 & 3 & . & 3 & 5 & 5_{(7)} \end{array}$$

d)

$$\begin{array}{rcccccccccc} & 7 & H & 4 & G & 9 & A & . & E & 6_{(20)} \\ + & C & F & 7 & J & 7 & C & . & 8 & D_{(20)} \\ \hline 1 & 0 & C & C & F & H & 3 & . & 2 & J_{(20)} \end{array}$$

e)

$$\begin{array}{rcccccccccc} & 6 & 3 & 4 & 5 & 2 & 1 & 7 & . & 8 & 4 & 1_{(9)} \\ + & 4 & 7 & 2 & 8 & 4 & 3 & 6 & . & 2 & 8_{(9)} & \\ \hline 1 & 2 & 1 & 7 & 4 & 6 & 5 & 5 & . & 2 & 3 & 1_{(9)} \end{array}$$

f)

$$\begin{array}{rcccccccccc} & 5 & D & F & 0 & 8 & C & . & A & 3_{(17)} \\ + & 9 & D & B & G & 1 & E & 9 & . & 5 & C_{(17)} & \\ \hline A & 2 & 8 & E & 2 & 6 & 4 & . & F & F_{(17)} & & \end{array}$$

g)

$$\begin{array}{rcccccccccc} & 8 & A & 7 & 4 & 2 & 6 & 3 & B & . & 4 & 9_{(14)} \\ + & & C & A & 3 & D & C & 5 & 8 & . & 9 & C & 7_{(14)} \\ \hline 9 & 9 & 3 & 8 & 2 & 4 & 9 & 6 & . & 0 & 7 & 7_{(14)} \end{array}$$

h)

$$\begin{array}{rcccccccccc} & 5 & A & G & 8 & C & D & 3 & . & 2 & 7_{(19)} \\ + & & G & 7 & H & A & 4 & F & . & C & E_{(19)} \\ \hline & 6 & 8 & 5 & 7 & 3 & H & I & . & F & 2_{(19)} \end{array}$$

1.7.

a)

$$\begin{array}{rcccccccccc} & 1 & 5 & D & A & 8 & 4 & 3 & . & 2 & 3_{(14)} \\ - & & B & A & 2 & 5 & 4 & 4 & . & 2 & B_{(14)} \\ \hline & & 8 & 3 & 8 & 2 & D & C & . & D & 6_{(14)} \end{array}$$

b)

$$\begin{array}{rcccccccccc} & F & 4 & J & 3 & 0 & I & 9 & . & 7 & A_{(21)} \\ - & C & E & H & 4 & 8 & A & K & . & 2 & G_{(21)} \\ \hline & 2 & B & 1 & J & D & 7 & A & . & 4 & F_{(21)} \end{array}$$

c)

$$\begin{array}{rcccccccccc} & 1 & 0 & 0 & 0 & 1 & 1 & 0 & . & 0 & 0 & 1_{(2)} \\ - & & 1 & 1 & 0 & 0 & 1 & 1 & . & 1 & 0 & 1_{(2)} \\ \hline & & & 1 & 0 & 0 & 1 & 0 & . & 1 & 0 & 0_{(2)} \end{array}$$

d)

$$\begin{array}{rcccccccccc} & 5 & 3 & 0 & 7 & G & 4 & 9 & . & 5_{(17)} & & \\ - & 4 & C & 1 & F & C & A & 1 & . & 7 & C & 4_{(17)} \\ \hline & & 7 & F & 9 & 3 & B & 7 & . & E & 4 & D_{(17)} \end{array}$$

1.9.

a) $78205B05.3DA_{(14)}$

b) $45171712.610_{(8)}$

c) $2A355.85723_{(11)}$

d) $33F41GG4.G_{(17)}$

1.11.

a) Cociente = $32CE.3_{(15)}$ Resto = $408_{(15)}$

b) Cociente = $3D9.A_{(17)}$ Resto = $21C_{(17)}$

c) Cociente = $403.22_{(9)}$ Resto = $2747_{(9)}$

d) Cociente = $1562.4_{(13)}$ Resto = $2723_{(13)}$

1.13.

a) Convirtiendo las cantidades a binario, se tiene:

$+65508_{(10)} = 0\ 1111111111100100_{(2)}$
$+\ \ 103_{(10)} = 0\ \ 0000000001100111_{(2)}$

El número más grande que cabe en 16 bits es 2^{16}-1=65535 si se suman 65508+103= 65611 por lo tanto se debe aumentar un byte para evitar el desbordamiento, dado que dos bytes son insuficientes para guardar el resultado. De tal forma que la suma queda de la siguiente manera:

$508_{(10)} =$	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	1	0	0
$103_{(10)} =$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1
	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1

b) Como las cantidades tienen signo diferente ($-65508_{(10)} + 103_{(10)}$) no hay riesgo a desbordamiento por lo tanto no es necesario aumentar bytes. Pero se debe complementar a dos la cantidad negativa antes de realizar la suma.

Complementación.

$-65508_{(10)} =$	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	0	
	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1	Complemento a 1
																+	1	
	1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**1**	**1**	**0**	**0**	**Complemento a 2**

Realizando la suma:

$-65508_{(10)} =$	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
$+\ \ 103_{(10)} =$	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1
	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1

Como el resultado de la suma es negativo, se deberá complementar a dos dicho resultado.

1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	1	Resultado
1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	0	0	Complemento a 1
															+	1	
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**Complemento a 2 y Resultado**

c) Como las cantidades tienen signo diferente ($+ 65508_{(10)} - 103_{(10)}$)
No existe desbordamiento.

Complementado a dos la cantidad negativa se tiene:

$-103_{(10)} =$	1	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1	
	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	0	Complemento a 1
																+	1	
	1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**1**	**Complemento a 2**

Suma:

$+ 65508_{(10)} =$		0	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	0
$+ \quad 103_{(10)} =$		1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	1
	1	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**	**1**	**1**	**1**	**1**	**1**	**0**	**1** **Resultado**

Acarreo Signo

El acarreo es información que se desprecia y como el resultado de la suma es positivo por lo tanto ese es el valor buscado.

d) Como ambas cantidades tienen signo negativo ($-65508_{(10)} - 103_{(10)}$), es necesario agregar ceros para que no se presente un desbordamiento y como lo mínimo que se puede agregar en este problema es un byte se deberán agregar a la derecha de la cantidad ocho bits antes de encontrar el complemento a dos.

Complemento a dos después de agregar el byte.

$-65508_{(10)} =$	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	1	0	0
	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	1	1
																								+	1
	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0

$-103_{(10)} =$	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1	1
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	0
																									+	1
	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	1

Sumando:

$65508_{(10)}$ =		1	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0
$103_{(10)}$ =		1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0	1	1	0	0	1
	1	1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	1

El desbordamiento se desprecia y como el resultado es negativo se deberá complementar a dos dicho resultado.

1	1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1	0	1	1	0	1	0	1		
1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0		
																							+	1		
1	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**1**	**0**	**0**	**1**	**0**	**1**	**1**	**Resultado**	

Capítulo 2 Métodos de conteo

2.3.

a) Total = $27 \times 36 \times 35 \times 34 \times 33 \times 32 = 1221454080$

b) Una sola letra = 27

Letra y un solo carácter (ya sea L o D) = 27*37 = 999

Letra y dos caracteres = $27 \times 37 \times 37 = 36963$

Letra y tres caracteres = $27 \times 37 \times 37 \times 37 = 1367631$

Letra y cuatro caracteres = $27 \times 37 \times 37 \times 37 \times 37 = 50602347$

Letra y cinco caracteres = $27 \times 37 \times 37 \times 37 \times 37 \times 37 = 1872286839$

Total = 27 + 999 + 36963 + 1367631+ 50602347 + 1872286839 = 1924294806

2.5.

a)

Maneras diferentes = $4^9 = 262144$ (Para las 9 preguntas con cuatro opciones)

Maneras diferentes = $2^{11} = 2048$ (Para las 11 preguntas de F y V)

Total = $262144 \times 2048 = 536870912$

b)

Maneras diferentes = $3^9 = 19683$ (Para las 9 preguntas)

Maneras diferentes = $1^{11} = 1$ (Para las 11 preguntas de F y V)

Total = $19683 \times 1 = 19683$

c)

Maneras diferentes = $1^9 = 1$ (Para las 9 preguntas)

Maneras diferentes = $1^{11} = 1$ (Para las 11 preguntas de F y V)

Total = $1 \times 1 = 1$

2.7.

a) Maneras diferentes = $5^{10} = 9765625$

b) Maneras diferentes = $4^{10} = 1048576$

c) Maneras diferentes = $1^{10} = 1$

d) Maneras diferentes = $4^3 = 64$

2.13.

a) $P(n, r) = (n - 1)! = (10 - 1)! = 362880$ Maneras diferentes.

b) Considerando como un bloque a las computadoras A,B y C que se desea que siempre estén juntas, por lo tanto ahora se tendrán solamente 8 elementos distintos, 7 computadoras individuales y un pequeño bloque de 3 computadoras. Pero además entre esas 3 computadoras el orden en que pueden estar no siempre es el mismo, ya que pueden colocarse de las siguientes seis maneras (3!): ABC, ACB, BAC, BCA, CAB y CBA. Por lo tanto el número de formas en que se pueden colocar las computadoras con este nuevo diseño es.

$P(n, r) = 6\ (8 - 1)! = 30240$

2.17.

a) Permutaciones = $n! = 12! = 479001600$

b) Permutaciones = $4! \times (3! \times 3! \times 3! \times 3!) = 31104$

c) Permutaciones = $3! \times 4! \times 4! \times 4! = 3!\ (4!)^3 = 82944$

d) Permutaciones = $4! \times 4! \times 4! = (4!)^3 = 13824$

e) Permutaciones = $3! \times 3! \times 3! \times 3! = 1296$

2.29.

a) La instrucción **x =3*x-y** se ejecuta.

Primer ciclo
Desde a = 2 hasta a > 5 con incrementos de 3 = 2 veces

Segundo ciclo
Desde b = 13 hasta b < 4 con decrementos de 2 = 5 veces

Total = 2 × 5 = 10 veces

b) 6, 3, 233756, –18

Capítulo 3 Conjuntos

3.3.

a) A = {x / x es el nombre de una operación aritmética básica}
b) B = {x / x∈ Z^+; x es divisible entre 3; 3 ≤ x ≤ 18}
c) C = { x / x∈ Z^+; x es primo; x < 18}
d) D = {x / x es nombre de un continente}
e) E = { x = 2^n / x∈ Z^+; n ∈ Z^+; 0 ≤ n < 7}

3.5.

| P(A)| = 2^4 = 16

P(A) = {∅, {manzana}, {pera}, {fresa}, {sandía}, {manzana,pera}, {manzana,fresa}, {manzana,sandía}, {pera,fresa}, {pera,sandía}, {fresa,sandía}, {manzana,pera,fresa}, {manzana,pera,sandía}, {manzana,fresa,sandía} {pera,fresa,sandía}, {manzana,pera,fresa, sandía}}

3.9.

a) F ⊆ (C-D) (V)
b) E ⊆ D (V)
c) E ⊆ (C ∩ D) (V)
d) (A ∩ B) = ∅ (F)
e) (D – C) ⊆ (B – A) (F)
f) (C ∩ D) ⊆ U (V)

g) $D = \{1, 2, 3, 5, 6, 7, 8, 13, 14\}$ (V)
h) $B \subseteq A$ (F)
i) $U - (C \cap D) = \{4, 15, 16\}$ (F)
j) $E - (C \cap D) = \{6\}$ (F)
k) $(C \oplus D) = \{1, 2, 3, 5, 9, 10, 11, 12, 14\}$ (F)
l) $D - U = \varnothing$ (V)
m) $(B - A) = \{5, 8\}$ (V)
n) $3 \in (A \cup B)$ (V)
ñ) $11 \notin (C - D)$ (F)
o) $(F \cup E) \subseteq C$ (V)
p) $(C \cup D)' = \{4,15,16\}$ (V)
q) $(C \cap E) = \varnothing$ (F)
r) $(E - F) \subseteq D$ (V)
s) $(B - E) \not\subset (D\text{-}C)$ (F)

3.13.

a) $C' = \{x / x \in \mathbf{Z};\ x \notin \{6,7,8,9,15,17,20,21,22,23,25\}\}$
$(C' \cap A) = \{11,13,19,29\}$
$B \oplus (C' \cap A) = \{9,12,15,16,17,19,21,23,29\}$
$D' = \{ x / x \in \mathbf{Z};\ x \notin \{11,13,15,17,19\}\}$
$B \oplus (C' \cap A)) - D' = \{15,17,19\}$

b) $(B - C) = \{9,11,12,13,16\}$
$D' = \{ x / x \in \mathbf{Z};\ x \notin \{11,13,15,17,19\}\}$
$((B - C) - D') = \{11,13\}$
$B' = \{x / x \in \mathbf{Z};\ x \notin \{9,11,12,13,15,16,17,21,23\}\}$
$(A \oplus B') = \{x / x \in \mathbf{Z};\ x \notin \{7,9,12,15,16,19,21,29\}\}$
$((B - C) - D') \cup (A \oplus B') = \{x / x \in \mathbf{Z};\ x \notin \{7,9,12,15,16,19,21,29\}\}$

c) $C' = \{x / x \in \mathbf{Z};\ x \notin \{6,7,8,9,15,17,20,21,22,23,25\}\}$
$(C' \cup B) = \{x / x \in \mathbf{Z};\ x \notin \{6,7,8,20,22,25\}\}$
$((C' \cup B) \oplus D) = \{x / x \in \mathbf{Z};\ x \notin \{6,7,8,11,13,15,17,19,20,22,25\}\}$
$A' = \{x / x \in \mathbf{Z};\ x \notin \{7,11,13,17,19,23,29\}\}$
$((C' \cup B) \oplus D) - A' = \{23,29\}$

d) $A' = \{x / x \in \mathbf{Z};\ x \notin \{7,11,13,17,19,23,29\}\}$
$C' = \{x / x \in \mathbf{Z};\ x \notin \{6,7,8,9,15,17,20,21,22,23,25\}\}$
$(A' \cap C') = \{x / x \in \mathbf{Z};\ x \notin \{6,7,8,9,11,13,15,17,19,20,21,22,23,25,29\}\}$

$B' = \{x \ / x \in \mathbf{Z};\ x \notin \{9,11,12,13,15,16,17,21,23\}\}$

$(B' \oplus (A' \cap C')) = \{6,7,8,12,16,19,20,22,25,29\}$

$(B' \oplus (A' \cap C')) - D = \{6,7,8,12,16,20,22,25,29\}$

e) $D' = \{x \ / x \in \mathbf{Z};\ x \notin \{11, 13,15,17,19\}\}$

$(A \cap D') = \{7,23,29\}$

$C' = \{x \ / x \in \mathbf{Z};\ x \notin \{6,7,8,9,15,17,20,21,22,23,25\}\}$

$A' = \{x \ / x \in \mathbf{Z};\ x \notin \{7,11,13,17,19,23,29\}\}$

$(C' \oplus A') = \{6,8,9,11,13,15,19,20,21,22,25,29\}$

$((A \cap D') - (C' \oplus A')) = \{7,23\}$

$((A \cap D') - (C' \oplus A')) - B = \{7\}$

3.15.

a)

$A' \cap B' \cap \mathbf{C'} \cap A \cap \mathbf{B'} \cap \mathbf{C'} \cap \mathbf{A'} \cap \mathbf{B} \cap C \cup \mathbf{A'} \cap \mathbf{B} \cap C' \cup \mathbf{A} \cap \mathbf{B} \cap C \cup \mathbf{A} \cap \mathbf{B} \cap C' = B \cup C'$

$B' \cap C' \cap (A' \cup A) \cup A' \cap B \cap (C \cup C') \cup A \cap B \cap (C \cup C') = B \cup C'$ Ley distributiva 4a

$B' \cap C' \cap \mathbf{U} \cup A' \cap B \cap \mathbf{U} \cup A \cap B \cap \mathbf{U} = B \cup C'$ Propiedades del complemento 9a

$B' \cap C' \cup A' \cap \mathbf{B} \cup A \cap \mathbf{B} = B \cup C'$ Ley de identidad 10b

$B' \cap C' \cup B \cap (A' \cup A) = B \cup C'$ Ley distributiva 4a

$B' \cap C' \cup B \cap U = B \cup C'$ Propiedades del complemento 9a

$B' \cap C' \cup B = B \cup C'$ Ley de identidad 10b

$B \cap B' \cap C' = B \cup C'$ Ley conmutativa 2a

$B \cup C' = B \cup C'$ Equivalencia 7a

b)

$A' \cap B' \cap C \cup \mathbf{A} \cap B' \cap C' \cup \mathbf{A} \cap B' \cap C \cup \mathbf{A} \cap \mathbf{B} \cap C \cup \mathbf{A} \cap \mathbf{B} \cap C' = A \cup B' \cap C$

$A' \cap B' \cap C \cup A \cap B' \cap (C' \cup C) \cup \mathbf{A} \cap B \cap (C \cup C') = A \cup B' \cap C$ Ley distributiva 4a

$A' \cap B' \cap C \cup A \cap B' \cap U \cup A \cap B \cap U = A \cup B' \cap C$ Propiedades del complemento 9a

$A' \cap B' \cap C \cup A \cap B' \cup A \cap B = A \cup B' \cap C$ Ley de identidad 10b

$\mathbf{A} \cap B' \cup \mathbf{A} \cap B \cup A' \cap B' \cap C = A \cup B' \cap C$ Ley conmutativa 2a

$A \cap (B' \cup B) \cup A' \cap B' \cap C = A \cup B' \cap C$
Ley distributiva 4a

$A \cap U \cup A' \cap B' \cap C = A \cup B' \cap C$
Propiedades del complemento 9a

$A \cup \mathbf{A'} \cap B' \cap C = A \cup B' \cap C$ Ley de identidad 10b

$A \cup B' \cap C = A \cup B' \cap C$ Equivalencia 7a

c)

$(((\mathbf{A} \cup \mathbf{B'})' \cup C)' \cap (\mathbf{C} \cup \mathbf{B})')' = B \cup C$

$((A' \cap B \cup C)' \cap (C' \cap B'))' = B \cup C$
Ley de Morgan 6a

$(A' \cap B \cup C) \cup (C' \cap B')' = B \cup C$
Ley de Morgan 6b

$A' \cap B \cup \mathbf{C} \cup \mathbf{C} \cup B = B \cup C$ Ley de Morgan 6b

$A' \cap B \cup C \cup \mathbf{B} = B \cup C$ Ley de idempotencia 5a

$B \cup A' \cap B \cup C = B \cup C$ Ley conmutativa 2a

$B \cup \mathbf{B} \cap A' \cup C = B \cup C$ Ley conmutativa 2b

$B \cup C = B \cup C$ Ley de identidad 10e

3.19.

a) $2^n - 1 = 2^5 - 1 = 31$

b)

$|A \cup B \cup C \cup D \cup E| = |A| + |B| + |C| + |D| + |E| - |A \cap B| - |A \cap C| - |A \cap D| - |A \cap E| - |B \cap C| - |B \cap D| - |B \cap E| - |C \cap D| - |C \cap E| - |D \cap E| + |A \cap B \cap C| + |A \cap B \cap D| + |A \cap B \cap E| + |A \cap C \cap D| + |A \cap C \cap E| + |A \cap D \cap E| + |B \cap C \cap D| + |B \cap C \cap E| + |B \cap D \cap E| + |C \cap D \cap E| - |A \cap B \cap C \cap D| - |A \cap B \cap C \cap E| - |A \cap B \cap D \cap E| - |A \cap C \cap D \cap E| - |B \cap C \cap D \cap E|.$

Capítulo 4 Lógica matemática

4.1.

a) $[p \rightarrow q] \wedge [r \rightarrow (s \vee t)] \Rightarrow [(t' \wedge q' \wedge s) \rightarrow u]$

b) $[p \leftrightarrow (q \wedge r \vee s)] \wedge [(p \wedge q' \vee s') \rightarrow t] \Rightarrow [t \rightarrow q]$

c) $[(p \vee q) \rightarrow r] \wedge [r \leftrightarrow s] \Rightarrow [[(q \vee p)' \wedge r'] \rightarrow s']$

d) $[p \leftrightarrow q'] \wedge [(r' \wedge q') \rightarrow (s \wedge t)] \Rightarrow [(q \vee r) \rightarrow (p' \wedge t')]$

e) $[(a \wedge b) \rightarrow c] \wedge [b' \rightarrow d] \Rightarrow [(c' \wedge d) \rightarrow a']$

4.5.

a)

$[(p \rightarrow r) \wedge (q \rightarrow r)] \equiv [(p \wedge q) \rightarrow r]$

$[(p \wedge q) \rightarrow (r \wedge r)] \equiv [(p \wedge q) \rightarrow r]$ por 9b

$[(p \wedge q) \rightarrow r] \equiv [(p \wedge q) \rightarrow r]$ por 21b

b)

$[p \vee (q \wedge r)] \equiv [(p \wedge p) \vee (p \wedge r) \vee (p \wedge q) \vee (q \wedge r)]$

$[p \vee (q \wedge r)] \equiv [p \vee (p \wedge q) \vee (p \wedge r) \vee (q \wedge r)]$ Por 21b y 18a

$[p \vee (q \wedge r)] \equiv [p \vee p \wedge (q \vee r) \vee (q \wedge r)]$ Por 20b

$[p \vee (q \wedge r)] \equiv [p \vee (q \wedge r)]$ Por 27f

4.13.

El teorema es:

$$[(p \wedge q) \rightarrow r] \wedge \Box[q' \rightarrow s'] \Rightarrow [(r' \wedge s) \rightarrow p']$$

Demostración por el método directo.

1.-	$(p \wedge q) \rightarrow r$	Hipótesis
2.-	$q' \rightarrow s'$	Hipótesis
3.-	$r' \rightarrow (p \wedge q)'$	1; Contrapositiva; 23
4.-	$r' \rightarrow (p' \vee q')$	3; Ley de Morgan; 22b
5.-	$s \rightarrow q$	2; Contrapositiva; 23
6.-	$[r' \rightarrow (p' \vee q')] \wedge [s \rightarrow q]$	4,5; Conjunción; 14
7.-	$(r' \wedge s) \rightarrow [(p' \vee q') \wedge q]$	6; Dilema constructivo; 9b
8.-	$(r' \wedge s) \rightarrow [q \wedge (p' \vee q')]$	7; Ley conmutativa; 18b
9.-	$(r' \wedge s) \rightarrow [(q \wedge p') \vee (q \wedge q')]$	8; Ley distributiva; 20b
10.-	$(r' \wedge s) \rightarrow [(q \wedge p') \vee 0]$	9; Contradicción; 26
11.-	$(r' \wedge s) \rightarrow (q \wedge p')$	10; Ley de identidad; 27a
12.-	$(r' \wedge s) \rightarrow (p' \wedge q)$	11; Ley conmutativa; 18b
13.-	$(p' \wedge q) \rightarrow p'$	Simplificación; 2
14.-	$(r' \wedge s) \rightarrow p'$	12,13; Silogismo hipotético; 13

Demostración por contradicción:

1.-	$(p \wedge q) \rightarrow r$	Hipótesis
2.-	$q' \rightarrow s'$	Hipótesis
3.-	$[(r' \wedge s) \rightarrow p']'$	Negación de la conclusión.

4.-	$[[(r' \wedge s) \wedge p]']'$	3; Variantes de la condicional; 24b
5.-	$(r' \wedge s) \wedge p$	4; Doble negación; 17
6.-	$(r' \wedge s)$	5; Simplificación; 11
7.-	p	5; Simplificación; 11
8.-	$r' \rightarrow (p \wedge q)'$	1; Contrapositiva; 23
9.-	$s \rightarrow q$	2; Contrapositiva; 23
10.-	$[r' \rightarrow (p \wedge q)'] \wedge [s \rightarrow q]$	8,9; Conjunción; 14
11.-	$(r' \wedge s) \rightarrow [(p \wedge q)' \wedge q]$	10; Dilema constructivo; 9b
12.-	$(r' \wedge s) \rightarrow [q \wedge (p \wedge q)']$	11; Ley conmutativa; 18b
13.-	$(r' \wedge s) \rightarrow [q \wedge (p' \vee q')]$	12; Ley de Morgan; 22b
14.-	$(r' \wedge s) \rightarrow [(q \wedge p') \vee (q \wedge q')]$	13; Ley distributiva; 20b
15.-	$(r' \wedge s) \rightarrow [(q \wedge p') \vee 0]$	14; Contradicción; 26
16.-	$(r' \wedge s) \rightarrow (q \wedge p')$	15; Ley de identidad; 27a
17.-	$(q \wedge p')$	6,16; Modus ponens; 15
18.-	p'	17; Simplificación; 11
19.-	$p \wedge p'$	7,18; Conjunción; 14
20.-	0	19; Contradicción; 26

4.25.

Sean:

p: "Tiene alas".
q: "Vuelan"
r: "Tienen alas y vuelan"
s: "Cacarean"
t: "Ponen huevos"
u: "Es ave"
v: "Es gallina"
w: "Es mamífero"

a)	$\forall x\, p(x)$.	(falso)
b)	$\exists x\, q(x)$.	(cierto)
c)	$\exists x\, [p(x) \wedge q(x)]$ o bien $\exists x\, r(x)$	(cierto)
d)	$\exists x\, [p(x) \wedge q'(x)]$	(cierto)
e)	$\forall x\, u(x) \Rightarrow p(x)$	(cierto)
f)	$\forall x\, [u(x) \wedge t(x) \wedge s(x)] \Rightarrow v(x)$	(cierto)
g)	$\exists x\, [v(x) \wedge t'(x)]$	(cierto)
h)	$\forall x\, u(x) \Rightarrow w'(x)$.	(cierto)

4.29.

a) $\exists y\ \forall x\ [p(x,y) \wedge q(x,y) \Rightarrow r(x,y)]$ $\quad x,y \in U$

b) $\forall y\ \exists x\ [p'(x,y) \vee r(x,y) \Rightarrow q'(x,y)]$ $\quad x,y \in U$

c) $\exists x\ \exists y\ r(x,y) \vee \exists x\ q(x,y) \wedge \exists y\ p'(x,y)$ $\quad x,y \in U$

Capítulo 5 Álgebra booleana

5.3.

a) $F = A'B'D' + A'BD' + A'BD + ABD$

$F = A'D'(B' + B) + BD(A' + A)$

$F = A'D' + BD$

b) $F = A'CD + ACD + A'B'D + A'B'C + AB'D + AB'CD'$

$F = CD(A' + A) + B'D(A' + A) + BC'(A' + AD')$

$F = CD + B'D + A'B'C + B'CD'$

$F = C(D + D'B') + B'D + A'B'C$

$F = CD + B'C + B'D + A'B'C$

$F = B'C(1 + A') + CD + B'D$

$F = B'C + CD + B'D$

c) $F = A'B'C'D' + A'B'CD' + A'BC'D + A'BCD + ABC'D' + ABCD' + AB'C'D' + AB'CD'.$

$F = A'B'D'(C' + C) + A'BD(C' + C) + ABD'(C' + C) + AB'D'(C' + C)$

$F = A'B'D' + A'BD + ABD' + AB'D'$

$F = B'D'(A' + A) + A'BD + ABD'$

$F = B'D' + A'BD + ABD'$

$F = D'(B' + BA) + A'BD$

$F = B'D' + AD' + A'BD$

d) $F = A'B'C'D'E + A'B'C'DE + A'B'C'DE' + A'BC + ABC + ABC'D'E' + ABC'D'E + ABC'DE + AB'C'D'E + AB'C'DE + AB'CDE + AB'CD'E$

$F= A'B'C'E(D' + D) + AB'C'E(D' + D) + BC(A' + A) + ABC'D'(E'+E) + AB'CE(D + D') + A'B'C'DE' + ABC'DE$

$F = A'B'C'E + AB'C'E + BC + ABC'D' + AB'CE + A'B'C'DE' + ABC'DE$

$F = B'C'E(A' + A) + B(C + C'AD') + AB'CE + A'B'C'DE' + ABC'DE$

$F = B'C'(E + E'A'D) + AB(D' + DC'E) + BC + AB'CE$

$F = B'C'E + A'B'C'D + ABD' + ABC'E + BC + AB'CE$

$F = B'E(C' + CA) + A'B'C'D + ABD' + ABC'E + BC$

$F = B'C'E + AB'E + A'B'C'D + ABD' + ABC'E + BC$

$F = C'E(B' + BA) + AB'E + A'B'C'D + ABD' + BC$

$F = B'C'E + AC'E + AB'E + A'B'C'D + ABD' + BC$

e) $F = ((A + B)' + C' + D')\,((AC)' + (A + (BC)')' + D)$

$F = (A'B' + C' + D')\,(A' + C' + (A + B' + C')' + D)$

$F = (A'B' + C' + D')\,(A' + C' + A'BC + D)$

$F = (A'B' + C' + D')\,(A'(1 + BC) + C' + D)$

$F = (A'B' + C' + D')\,(A' + C' + D)$

$F = A'A'B' + A'B'C' + A'B'D + A'C' + C'C' + C'D + A'D' + C'D' + D'D$

$F = A'B' + A'B'C' + A'B'D + A'C' + C' + C'D + A'D' + C'D'$

$F = A'B'(1 + C' + D) + C'(A' + 1 + D + D') + A'D'$

$F = A'B' + C' + A'D'$

f) $F = A'B'C'D + A'B'CD + A'B'CD' + A'BCD + ABCD' + AB'C'D + AB'CD + AB'CD'$

$F = A'B'D(C' + C) + AB'C(D + D') + A'B'CD' + A'BCD + ABCD' + AB'C'D$

$F = A'B'D + AB'C + A'B'CD' + A'BCD + ABCD' + AB'C'D$

$F = A'B'(D + D'C) + AB'(C + C'D) + A'BCD + ABCD'$

$F = A'B'D + A'B'C + AB'C + AB'D + A'BCD + ABCD'$

$F = B'C(A' + A) + B'D(A' + A) + A'BCD + ABCD'$

$F = B'C + B'D + A'BCD + ABCD'$

$F = C(B' + BA'D) + B'D + ABCD'$

$F = B'C + A'CD + B'D + ABCD'$

$F = C(B' + BAD') + A'CD + B'D$

$F = B'C + ACD' + A'CD + B'D$

g) $F = A'B'C'D' + A'B'CD + A'B'CD' + ABC'D + ABCD + ABCD' + AB'C'D' + AB'CD + AB'CD'$

$F = A'B'C(D + D') + ABD(C' + C) + AB'C(D + D') + B'C'D'(A' + A) + ABCD'$

$F = A'B'C + ABD + AB'C + B'C'D'(A' + A) + ABCD'$

$F = B'C(A' + A) + AB(D + D'C) + B'C'D'$

$F = B'C + ABD + ABC + B'C'D'$

$F = B'(C + C'D') + ABD + ABC$

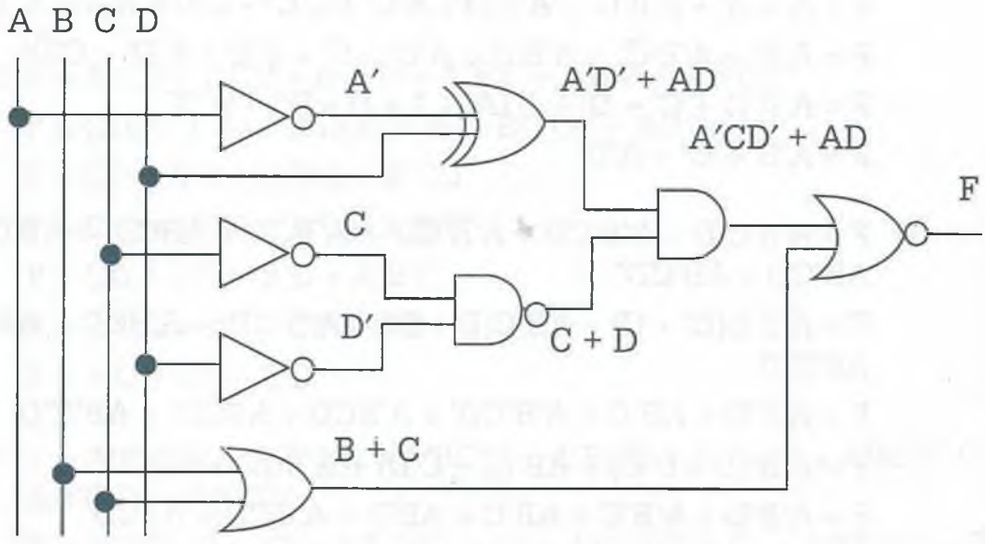
$F = B'C + B'D' ABD + ABC$

$F = C(B' + BA) + B'D' + ABD$

$F = B'C + AC + B'D' + ABD$

5.11.

a) La función booleana de salida es $F = [(A'CD'+AD)+(B+C)]'$ y los parciales de cada una de las compuertas se muestran en el siguiente diagrama (En algunas de las salidas se hicieron algunas operaciones para reducir un poco la expresión).



b) La función booleana simplificada en sumas de productos es $F = B'C'D' + A'B'C'$, la cual es posible encontrar usando la función de salida y simplificándola por medio de teoremas o bien mapas de Karnaugh.

c) La función booleana simplificada en productos de sumas, es $F = B'C'(A' + D')$.

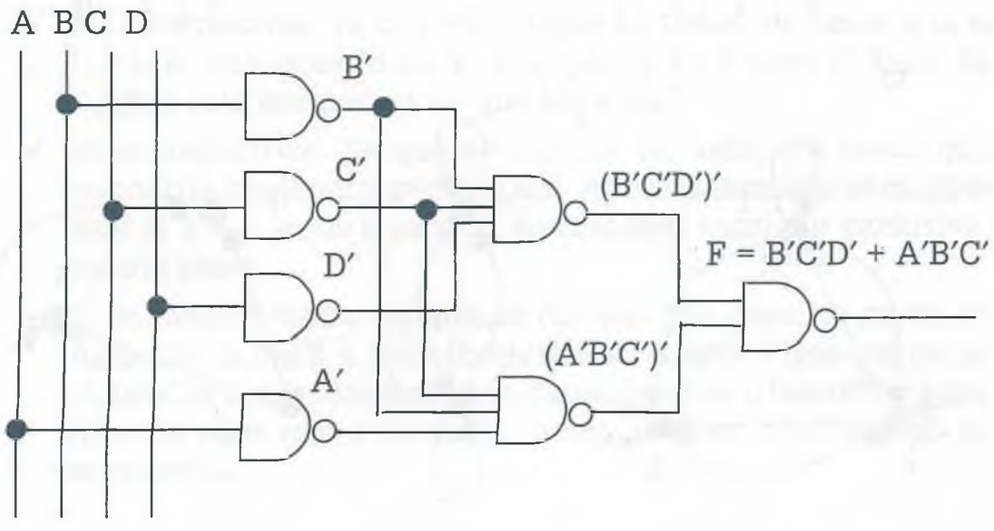
d)

A	B	C	D	A′	B′	C′	D′	B′C′D′	A′B′C′	B′C′D′ + A′B′C′	A′CD′	A′CD′ + AD	B + C	F
0	0	0	0	1	1	1	1	1	1	1	0	0	0	1
0	0	0	1	1	1	1	0	0	1	1	0	0	0	1
0	0	1	0	1	1	0	1	0	0	0	1	1	1	0
0	0	1	1	1	1	0	0	0	0	0	0	0	1	0
0	1	0	0	1	0	1	1	0	0	0	0	0	1	0
0	1	0	1	1	0	1	0	0	0	0	0	0	1	0
0	1	1	0	1	0	0	1	0	0	0	1	1	1	0
0	1	1	1	1	0	0	0	0	0	0	0	0	1	0
1	0	0	0	0	1	1	1	1	0	1	0	0	0	1
1	0	0	1	0	1	1	0	0	0	0	0	1	0	0
1	0	1	0	0	1	0	1	0	0	0	0	0	1	0
1	0	1	1	0	1	0	0	0	0	0	0	1	1	0
1	1	0	0	0	0	1	1	0	1	0	0	0	1	0
1	1	0	1	0	0	1	0	0	0	0	0	1	1	0
1	1	1	0	0	0	0	1	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	0	1	1	0

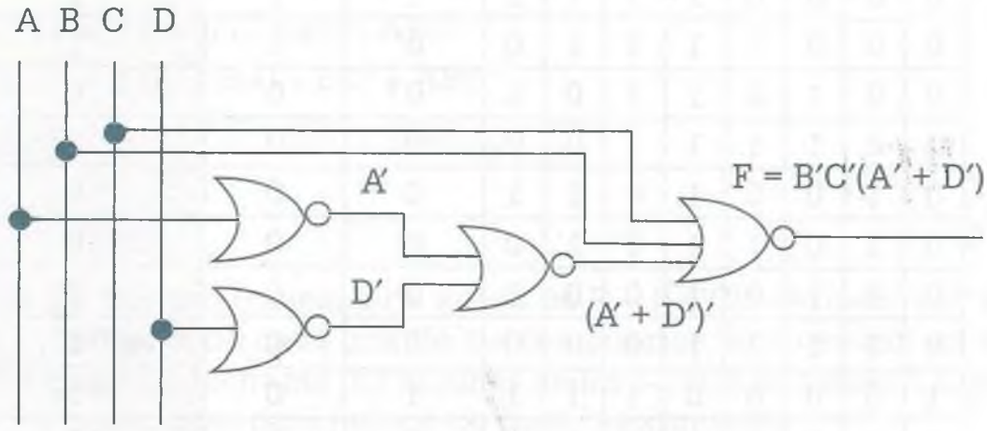
Donde F = [(A′CD′ + AD) + (B + C)]′

En la tabla de verdad anterior se puede observar que efectivamente las expresiones boolenas [(A′CD′ + AD) + (B + C)]′ y B′C′D′ + A′B′C′ son lógicamente equivalentes ya que coinciden en todas sus líneas.

e) Expresión F = B′C′D′ + A′B′C′ con compuertas Nand



f) Diagrama de la expresión booleana obtenida en el inciso c, usando para ello exclusivamente compuertas Nor.



Capítulo 6 Relaciones

6.1.

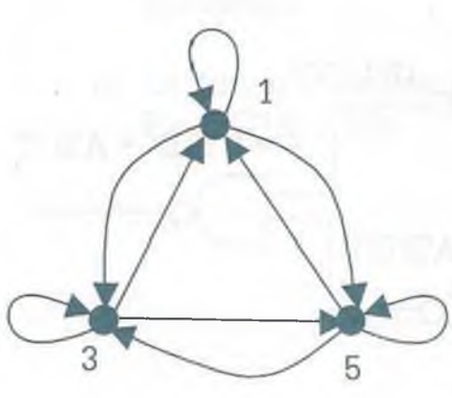
a)

$$S = \{(4,1), (4,3), (5,1), (5,3)\}$$

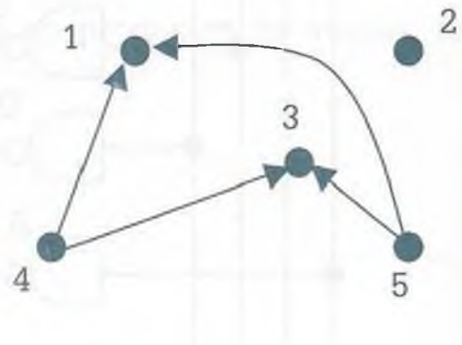
b)

$$M_R = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 1 & 0 & 1 & 0 & 1 \\ 2 & 0 & 1 & 0 & 1 & 0 \\ 3 & 1 & 0 & 1 & 0 & 1 \\ 4 & 0 & 1 & 0 & 1 & 0 \\ 5 & 1 & 0 & 1 & 0 & 1 \end{array} \qquad M_S = \begin{array}{c|ccccc} & 1 & 2 & 3 & 4 & 5 \\ \hline 1 & 0 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 \\ 3 & 0 & 0 & 0 & 0 & 0 \\ 4 & 1 & 0 & 1 & 0 & 0 \\ 5 & 1 & 0 & 1 & 0 & 0 \end{array}$$

c)



Grafo dirigido de R



Grafo dirigido de S

d)

Relación R

- Si es reflexiva. Ya que $\forall a \in A$, $(a,a) \in R$.
- No es irreflexiva. Ya que no se cumple que $\forall a \in A$, $(a,a) \notin R$. Ejemplo de ello es el par ordenado (1,1) que se encuentra en la relación.
- Si es simétrica. Ya que se cumple en todos los casos que si $(a,b) \in R$, entonces $(b,a) \in R$. O bien $M_R = M_R^T$.
- No es asimétrica. Ya que no se cumple que cuando $(a,b) \in R$ entonces $(b,a) \notin R$. Ejemplo de ello es que $(3,1) \in R$ pero también $(1,3) \in R$. Por otro lado; tampoco se cumple que si $a = b$ $(a,a) \notin R$. En otras palabras los pares colocados simétricamente deben ser contrarios y la diagonal debería contener solamente ceros, pero no ocurre así.
- No es antisimétrica. Ya que no ocurre con los pares simétricos que $(a,b) \notin R$ o bien $(b,a) \notin R$. Ejemplo de ello es que $(2,4) \in R$ y también $(4,2) \in R$.
- Si es transitiva. Ya que se cumple en todos los casos que si $(a,b) \in R$ y $(b,c) \in R$ entonces $(a,c) \in R$. La forma de llevar a cabo esta comprobación es demostrando que $M_R = (M_R \cup M_R^2)$, como se muestra a continuación. (Ver página Web)

Relación S

- No es reflexiva. Ya que no se cumple que $\forall a \in A$, $(a,a) \in R$. Ejemplo $(1,1) \notin S$
- Si es irreflexiva. Ya se cumple que $\forall a \in A$, $(a,a) \notin R$. Se observa porque la diagonal principal de M_S contiene exclusivamente ceros.
- No es simétrica. Ya que no cumple en todos los casos que si $(a,b) \in R$, entonces $(b,a) \in R$. Ejemplo $(4,1) \in S$ pero $(1,4) \notin S$. Se ratifica esta afirmación ya que $M_R \neq M_R^T$.
- Si es asimétrica. Ya que se cumple en todos los casos que cuando $(a,b) \in R$ entonces $(b,a) \notin R$. Además también se cumple que si $a = b$ $(a,a) \notin R$ ya que su diagonal contiene exclusivamente ceros.
- Si es antisimétrica. Ya que se cumple que para los pares simétricos $(a,b) \notin R$ o bien $(b,a) \notin R$. Esto implica que los pares colocados simétricamente son pares de ceros o bien contrarios (Uno de ellos es 0 y el otro 1), la diagonal en este caso no es importante.

- No es transitiva. Ya que no existe ningún caso en donde si $(a,b) \in R$ y $(b,c) \in R$ entonces $(a,c) \in R$. Además se cumple que $M_R = (M_R \cup M_R^2)$. Ya que $M_R^2 = \varnothing$ como se muestra a continuación. (Ver página Web)

e)

La relación R

Si es una relación de equivalencia ya que es Reflexiva, Simétrica y Transitiva.

- Las clases de equivalencia son:

  $[1] = [3] = [5] = \{1,3,5\}$
  $[2] = [4] = \{2,4\}$

- La partición es:

  $\lambda = \{[1], [2]\} = \{\{1,3,5\}, \{2,4\}\}$

La relación S

No es una relación de equivalencia ya que no es Reflexiva, tampoco es Simétrica ni Transitiva. Por tal razón se deberán aplicar las cerraduras correspondientes.

- Cerradura Reflexiva:
- $(S \cup I) = \{(1,1), (2,2), (3,3), (4,1), (4,3), (4,4), (5,1), (5,3), (5,5)\}$
- Cerradura Simétrica:
- $(S \cup S^{-1})$

  $= \{(1,1), (1,4), (1,5), (2,2), (3,3), (3,5), (4,1), (4,3), (4,4), (5,1), (5,3), (5,5)\}$
- Cerradura transitiva. (Ver página Web)

Como esta relación ahora es transitiva

- Las clases de equivalencia son:

  $[1] = [3] = [4] = [5] = \{1,3,4,5\}$
  $[2] = \{2\}$

- La partición es:

  $\lambda = \{[1], [2]\} = \{\{1,3,4,5\}, \{2\}\}$

g) No es relación de equivalencia ya que no es reflexiva, ni simétrica aunque si es transitiva.

6.17.

a) $R = \{(1,a),(2,b),(3,d),(4,c)\}$

- Es una función.
- $Dom(R)=Dom(f)=\{1,2,3,4\}$; $Cod(R)=Cod(f)=\{a,b,c,d\}$
- Es Inyectiva, ya que $\forall a \in A$ corresponde un elemento diferente $b \in B$. Es suprayectiva ya que $Cod(f)=B$. Por lo tanto se trata de una función Biyectiva.
- Es invertible, su inversa es $f^{-1} = R^{-1} = \{(a,1), (b,2), (d,3), (c,4)\}$. Por ser invertible su inversa también es una función.

b) $R = \{(1,a), (2,a), (3,a), (4,a)\}$

- Es una función.
- $Dom(R)=Dom(f)=\{1,2,3,4\}$; $Cod(R)=Cod(f)=\{a\}$
- No es Inyectiva, ya que no se cumple que $\forall a \in A$ corresponde un elemento diferente $b \in B$. Ejemplo: los elementos 1, 2, 3 y 4 tienen la misma imagen **a**. Tampoco es suprayectiva ya que $Cod(f) \neq B$, faltan los elementos b,c y d en el codominio. No es biyectiva porque no es inyectiva ni suprayectiva.
- No es invertible, porque no es una biyección.

c) $R = \{(1,b), (1,c), (2,a), (2,d)\}$

- No es función ya que $Dom(R) \neq A$. En otras palabras no se encuentran relacionados todos los elementos de A.
- $Dom(R)=\{1,2\}$; $Cod(R)=\{a,b,c,d\}$
- Como no es función no procede verificar si es inyectiva, suprayectiva o biyectiva.
- Como no es función entonces no procede verificar si se trata de una función invertible.

d) $R = \{(1,c), (2,c), (3,d), (4,d)\}$

- Si es función.
- $Dom(R)=Dom(f)=\{1,2,3,4\}$; $Cod(R)=Cod(f)=\{c,d\}$
- No es inyectiva ya que elementos del conjunto A tienen la misma imagen en B. Ejemplo: (1,c) y /2,c). Tampoco es suprayectiva ya que $Cod(f)\neq B$
- No es invertible ya que no es biyectiva.

6.21.

a) $g \circ f\,(2) = g(f(2)) = g(3(2) + 2^3) = g(14) = 14^5 = 537824$

b) $f \circ g\,(x - 1) = f(g(x - 1)) = f((x - 1)^5) = 3(x - 1)^5 + (x - 1)^{15}$

c) $g \circ f \circ h(x) = g(f(h(x))) = g(f(x + 1)) = g(3(x + 1) + (x + 1)^3) = (3(x + 1) + (x + 1)^3)^5$

d) $f \circ g \circ h(-x) = f(g(h(-x))) = f(g(1 - x)) = f((1 - x)^5) = 3(1 - x)^5 + (1 - x)^{15}$

Capítulo 7 Grafos

7.1.

a) No es grafo simple ya que los grafos simples no tienen lazos ni lados paralelos.

b) No es K_n, porque K_n no tiene lazos ni lados paralelos, además la valencia de cada uno de los vértices de un K_n es $(n - 1)$ y eso no ocurre en este caso.

c) No es un $K_{n,m}$, ya que $K_{n,m}$ no tiene lazos ni lados paralelos. Además en $K_{n,m}$ existen dos conjuntos en donde los elementos de uno de esos conjuntos está relacionado con los elementos del otro, pero entre elementos de un mismo conjuntos no existe ninguna relación, lo cual no se cumple en este caso.

d) Si es conexo ya que se puede encontrar un camino entre dos vértices cualquiera.

e) Si es plano, de hecho está dibujado en forma plana, ya que ninguna de sus aristas se cruzan. Si se cumple la ecuación de Euler $c = a - v + 2$. Ya que $c = 10$, $a = 18$ y $v = 10$, de tal manera que al sustituir en la ecuación se tiene que $10 = 18 - 10 + 2$

f) No tiene camino de Euler.

g) No tiene circuito de Euler. Ya que no todos su vértices tienen valencia par.

h) No tiene circuito de Hamilton.

i) Los elementos de cada conjunto son:

- V = {1,2,3,4,5,6,7,8,9,10}
- A = {a,b,c,d,e,f,g,h,i,j,k,l,m,n,ñ,o,p,q}.
- L = {f,p}
- P = {e,q}

k) La valencia de cada uno de los vértices es la que se indica en la última columna de la matriz de incidencia.

l) caminos, camino simple, circuito, circuito simple.

- (1,5,8,9,7,6,1,3). Camino, camino simple de longitud = 7.
- (5,8,4,10,10,7,6,5). Camino, circuito.
- (1,3,3,1). Camino, circuito
- (4,10,9,7,6,5,1,3,8,4).Camino, circuito simple.
- (2,6,5,8,9,10). Camino, camino simple de longitud = 5.

7.5.

a) Es un grafo bipartido donde A = {1,3,5,8} y B = {2,4,6,7}.

b) Es grafo bipartido completo $K_{5,2}$ en donde A = {a,b,d,f,g} y B = {c,e}

c) Grafo bipartido A = {e,b,d} y B = {a,c,f,g,h}

d) Es un grafo completo K_6 en donde A = {1,2,3,4,5,6}

Capítulo 8 Árboles

8.1.

a) Recorridos sin balancear el árbol.

- Primero: (a,b,e,i,n,r,s,t,u,w,v,x,y,j,ñ,o,c,f,g,k,l,p,q,d,h,m).
- Segundo: (i,r,n,t,s,w,u,x,v,y,e,ñ,j,o,b,a,f,c,k,g,p,l,q,d,m,h).
- Final: (r,t,w,u,x,y,v,s,n,i,ñ,o,j,e,b,f,k,p,q,l,g,c,m,h,d,a).

c) Recorridos con el árbol ya balanceado.

- Primero: (a,b,e,n,ñ,o,f,p,q,r,g,s,t,u,c,h,v,w,x,i,y,j,d,k,l,m).
- Segundo: (n,e,ñ,o,b,p,f,q,r,s,g,t,u,a,v,h,w,x,c,y,i,j,k,d,l,m).
- Final: (n,ñ,o,e,p,q,r,f,s,t,u,g,b,v,w,x,h,y,i,j,c,k,l,m,d,a).

8.3.

b) Recorrido en orden:

- Primero: /-+ba*bc+/ac/*dcb
- Segundo: b+a–b*c/a/c+d*c/b
- Final: ba+bc*–ac/dc*b/+/

8.7.

b)

- Primero: (m, d, c, a, a, c, b, d, f, g, m, z)
- Segundo: (a, a, b, c, c, d, d, f, g, m, m, z)
- Final: (a, b, c, a, d, c, m, g, f, d, z, m)

8.13.

b) Codificación de. **solounabuenailusion** es.
= 000111010001100010000111100100110100011101100000100010111100000

Capítulo 9 Introducción a los lenguajes formales

9.1.

a) **Es regular**. Ya que únicamente tienen un símbolo no terminal del lado izquierdo de todas las composiciones y del lado derecho tiene a lo más un solo símbolo no terminal. También **es libre de contexto y sensible al contexto**.

b) **No es regular**. Ya que tiene más de un símbolo no terminal del lado izquierdo de algunas composiciones.

No es libre de contexto. Ya que del lado izquierdo de algunas composiciones hay más de un símbolo no terminal, además algunas composiciones tienen mayor número de símbolos del lado izquierdo que del lado derecho.

Es sensible al contexto. Ya que no guarda ninguna restricción.

c) **No es regular**. Ya que algunas composiciones tienen del lado derecho más de un símbolo no terminal.

Es libre de contexto. Ya que del lado izquierdo de la composición tienen un solo símbolo no terminal.

Es sensible al contexto ya que toda gramática libre de contexto es también sensible al contexto.

9.5.

a) $(L \cup M) = \{\varepsilon, 0, 1, 00, 01, 10, 000, 100, 111\}$.

b) $(L \cap M) = \{00, 111\}$

c) $LM = \{01, 000, 001, 0100, 0111, 0000, 11, 100, 101, 1100, 1111, 1000, 0001, 00100, 00111, 00000, 1001, 10100, 10111, 10000, 11100, 11101, 111100, 11111, 111000\}$

d) $L - M = \{\varepsilon, 0, 10\}$

e) $M^I = \{1, 00, 10, 001, 111, 000\}$

f) $L^I = \{0, 1, 00, 01, 111\}$

g) $L^2 = \{00, 01, 000, 010, 0111, 10, 11, 100, 110, 1111, 001, 0000, 0010, 00111, 101, 1000, 1010, 10111, 1110, 11100, 11110, 111111\}$

h) $L^+ = L^1 \cup L^2 \cup L^3 \cup \ldots\ldots \cup L^\infty = \{0, 1, 00, 10, 111\} \cup \{00, 01, 000, 010, 0111, 10, 11, 100, 110, 1111, \ldots\ldots\ldots\ldots\} \cup \ldots\ldots = \{0, 1, 00, 10, 111, 01, 000, 010, 0111, 11, 100, 110, 1111, \ldots\ldots\ldots\ldots, 111111, \ldots\ldots\ldots\}$

i) $L^* = L^0 \cup L^1 \cup L^2 \cup \ldots\ldots \cup L^\infty = \{\varepsilon\} \cup \{\varepsilon, 0, 1, 00, 10, 111\} \cup \{00, 01, 000, 010, 0111, 10, 11, 100, 110, 1111, \ldots\ldots\ldots\ldots\} \cup \ldots\ldots = \{\varepsilon, 0, 1, 00, 10, 111, 01, 000, 010, 0111, 11, 100, 110, 1111, \ldots\ldots\ldots\ldots, 111111, \ldots\ldots\ldots\}$

j) $L' = L^* - L = \{x \mid x \text{ es una cadena de 0s y 1s; } x \in L^*; x \notin L\}$

9.9.

a) $(LM^*)^I \cup (M^0 \cup M^+)^I L^I = (M^*)^I L^I$
$(M^*)^I L^I \cup (M^*)^I L^I = (M^*)^I L^I$
$(M^*)^I L^I = (M^*)^I L^I$

b) $((\varepsilon \cup M)^*((L^+)^+ \cup LL^*))^I = (L^+)^I (M^*)^I$
$((M)^*(L^+ \cup L^+))^I = (L^+)^I (M^*)^I$
$(M^*L^+)^I = (L^+)^I (M^*)^I$
$(L^+)^I (M^*)^I = (L^+)^I (M^*)^I$

Índice analítico

Esta edición se terminó de imprimir en diciembre de 2008. Publicada por ALFAOMEGA GRUPO EDITOR, S.A. de C.V. Apartado Postal 73-267, 03311, México, D.F. La impresión y encuadernación se realizaron en FUENTES IMPRESORES, S.A, Centeno No. 109, Col. Granjas Esmeralda, Iztapalapa, 09810, México, D.F

Matemáticas para la computación

Uno de los pilares fundamentales de las ciencias de la computación son las matemáticas discretas que se exponen en este libro de texto, escrito para los primeros semestres de las carreras del área de la informática, y en el que se presentan los temas de Sistemas numéricos, Métodos de conteo, Conjuntos, Lógica matemática, Álgebra booleana, Relaciones, Grafos, Árboles e Introducción a los lenguajes formales.

Con el fin de destacar los principios matemáticos usados en la creación de herramientas computacionales, a lo largo de todo el libro se analizan ejemplos de aplicaciones específicas.

Por otro lado y como parte esencial de la obra, el material de la página Web de apoyo de este libro incluye:

- Diagramas de flujo.
- Simuladores.
- Hojas de cálculo.
- Software.
- Videos explicativos.
- Respuesta y solución de problemas seleccionados.
- Autoevaluaciones.
- Lecturas adicionales.
- Vínculos de interés.

Además se incluyen los siguientes recursos exclusivos para los docentes:

- Evaluaciones propuestas.
- Presentaciones.
- Respuesta y desarrollo de todos los problemas del libro.

www.alfaomega.com.mx

Apoyo en la

ISBN 978-970-15-1401-6

Alfaomega Grupo Editor