



UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

INGENIERÍA EN COMPUTACIÓN

ESTRUCTURAS DISCRETAS

GRUPO 6

ING. ORLANDO ZALDÍVAR ZAMORTEGUI

SEMESTRE 2024-1

SISTEMAS ALGEBRAICOS APLICADOS A FUNCIONES BOOLEANAS.
PROGRAMA DE CÓMPUTO DEL MÉTODO DE QUINEMCCLUSKEY PARA
MINIMIZAR FUNCIONES BOOLEANAS.

EQUIPO 7

BARRIOS AGUILAR DULCE MICHELLE
DOMÍNGUEZ PALACIOS JESÚS ALEJANDRO
GARCÍA VÁZQUEZ JAVIER ALEJANDRO
MARÍN MONTAÑO JOSUÉ
PÉREZ GONZÁLEZ SHARON LESLIE

FECHA DE ENTREGA: 9 DE NOVIEMBRE DEL 2023

Tema:

Sistemas Algebraicos aplicados a funciones booleanas. Programa de cómputo del método de QuineMcCluskey para minimizar funciones booleanas.

Temario

1. Objetivo
2. Introducción
3. Definiciones y Conceptos.
 - 3.1. Sistemas Algebraicos
 - 3.2. Algebra Booleana
 - 3.3. Métodos de minimización (menciones y breves definiciones)
 - 3.4. Método QuineMcCluskey
4. Ejemplos
5. Video
6. Cuestionario
7. Software/Programa
8. Bibliografía
9. Evaluación

Objetivos

1. Investigar acerca de los sistemas algebraicos aplicados en un entorno booleano, profundizando en el método de minimización de QuineMcCluskey.
2. Realizar una explicación clara para un interesado en diseñar algún circuito lógico, proporcionando los conceptos necesarios y algunos métodos de minimización para el álgebra booleana a usar, incluyendo ejemplos.
3. Proporcionar un software capaz de interpretar el método QuineMcClusky, como método de minimización principal del proyecto, y que el usuario sea capaz de presentar el tema con claridad al usuario, o bien, que le sirva como apoyo didáctico.

Introducción

Este trabajo es una búsqueda de implementación de un método de minimización de funciones booleanas, vista desde un punto teórico y llevado a uno práctico; haciendo para esta segunda parte mencionada un programa hecho en lenguaje JavaScript. Sin embargo, antes de profundizar de lleno al método que nos compete en esta ocasión, mencionemos un poco de los antecedentes que nos llevaron a la creación de este trabajo.

Para la materia Estructuras Discretas, impartida por el profesor Orlando Zaldívar Zamorategui, hemos abordado el manejo de la lógica proposicional, como una forma de demostración de la validez de ciertos razonamientos de interés. Si bien, en el lenguaje común los podemos encontrar en forma de presentación de un evento y efectuar una categorización de la información dada, como proposiciones que forman una premisa (o hipótesis) y llegar a una conclusión (que no siempre es válida); siendo este caso un método inductivo. O una oración en la cual se nos describe una situación general que va a la particular; un ejemplo podría ser: "Todos los perros ladran.", y "Lanudo es un perro, por lo tanto ladra."; de esto hay miles de ejemplos, mismos que son métodos de deductividad, método que a nosotros como parte del curso de Estructuras Discretas nos compete y trabajaremos de una forma distinta a lo que nos ofrecen los anteriores ejemplos.

Bien, de estas proposiciones, que deben ser entendidas como enunciados declarativos que pueden adquirir el valor de verdadero o falso; al momento de reproducirlas en dos categorías: Atómicas/primarias/simples y Compuestas/moleculares/secundarias; estas segundas derivadas de las atómicas, pues las atómicas representan enunciados declarativos (que solo presentan información tomada como verdadera, podríamos decir); son capaces de poseer conectivos (unarios o binarios). Quedando para nosotros trabajar con una simbología estricta, siendo las representaciones de las proposiciones letras mayúsculas (aquí se debe ser claro, solamente las atómicas pueden ser representadas por letras mayúsculas), y sus conectivos serán negación (representada por \neg), disyunción (representada por \vee) conjunción (representada por \wedge), condicional (representada por \rightarrow), y por último el bicondicional (representado por \leftrightarrow). De aquí se desglosa todo un tópico que denominado cálculo de predicados, que deriva en otros asuntos que refieren a su propio manejo como lo dictan sus leyes y propiedades, que son regidas por la parte más básica de toda la lógica proposicional que son las tablas de verdad que ayudan a verificar a su vez la validez de las leyes mencionadas.

Aunque, si bien este es un tema de arduo interés. Ya podemos ir mencionando que, de aquí, de la formación de las fórmulas proposicionales que se crean usando atómicas y compuestas que muestran una presunta validez, que en lenguaje de lógica proposicional se representa a la equivalencia con el símbolo \Leftrightarrow . Para el álgebra booleano, es distinto en cuanto a simbología se refiere y el uso de los últimos dos conectivos mencionados, la condicional y la bicondicional; además

de que la simbología de representación de lo que conocíamos como atómicas, de mayúsculas pasan a ser minúsculas. Y en resumen, tenemos que:

- Negación (7)
- Disyunción (\vee) +
- Conjunción (\wedge) ·
- Y para la equivalencia (\Leftrightarrow) =

Claro, lo anterior son ejemplos de expresiones booleanas simples. Pero, precisamente, por los casos en que se presentan expresiones complejas y extensas es que se necesitan aplicar ciertos métodos de minimización para obtener la expresión más simple. Y ese es el objetivo de este proyecto que se resumen en: investigar, comprender e implementar de forma teórica y por medio de un software el método de minimización de QuineMcCluskey para funciones booleanas.

Por lo que, para llegar a cumplirlo, se dividió el proyecto en varias secciones. La primera sección, se encargará de proporcionar las definiciones y conceptos del álgebra booleana contando con ejemplos y apoyándonos una vez más de la tabla VI. Leyes y propiedades (álgebra booleana); abordándolo desde la base que tenemos en la aritmética, el álgebra común que conocemos para realizar operaciones matemáticas; para seguir con el álgebra booleana que hemos mencionado brevemente en esta introducción (pero con ejemplos y aplicando las leyes para buscar minimizar sin ayuda de los métodos de minimización); mencionar los métodos de minimización existentes, para dar introducción al método de QuineMcCluskey.

Para la segunda sección, vamos a proporcionar ejemplos de la aplicación del método QuineMcCluskey en forma de imágenes y explicación por medio de notas agregadas para una mejor interpretación por parte del usuario. Seguido de un vídeo abordando la parte teórica necesaria, y un segundo para realizar un ejemplo. Y, como recomendación al usuario: repase el trabajo, pues este proyecto está acompañado de un breve cuestionario que le ayudará a repasar el tema. Si aún hay dudas, o quiere un apoyo para un problema más complejo: se adjunta un programa hecho con tal de proporcionar una solución al usted, el usuario.

DEFINICIONES Y CONCEPTOS

Sistemas algebraicos

Solemos llamarles sistemas algebraicos a las estructuras matemáticas que consisten en un conjunto de elementos unidos a una o más operaciones definidas. Estas operaciones son las principales que encontramos en la aritmética, como la suma y multiplicación, hasta operaciones más abstractas.

Sin embargo, usualmente se manejan datos de los tipos que representan cantidades; los números reales (π^4 , $\frac{e}{2}$, $\sqrt{2} + 1$, e , π , entre otros) en donde encontramos aún más tipos, como: los números racionales, enteros, naturales y por otro lado tenemos a los números imaginarios y luego los complejos. Mismos que van a ser representados de otra manera, en caso de desconocerlos. Por lo tanto, habrá que definir a los sistemas algebraicos como:

Los conjuntos de una o más ecuaciones que posean más de una incógnita y que conforman un problema matemático; para el cual nos compete encontrar los valores que van a tomar las incógnitas que satisfacen las operaciones que forman parte de las ecuaciones ya mencionadas; a esto nos referimos a que también se debe cumplir la igualdad del sistema en cuestión.

Dentro de los sistemas algebraicos, encontramos ciertas representaciones de elementos en las operaciones que conforman las ecuaciones, como letras del alfabeto latino (las últimas, como: u, v, w, x, y, z) o si son demasiadas representaciones usando una misma letra, se suelen utilizar los subíndices.

Bien, conocemos lo que son los sistemas algebraicos y la definición anterior es suficiente para lo que necesitamos, pero repasemos en forma de mención a las propiedades que regulan lo que se puede o no hacer con los sistemas.

PROPIEDADES

- **Cerradura.**
- **Asociativa.**
- **Identidad.**
- **Inverso.**
- **Distributiva.**
- **Cancelación.**
- **Elemento de Idempotente.**
- **Homomorfismo (morfismo):** una función que preserva operaciones definidas en objetos matemáticos con la misma estructura algebraica.

Ejemplo de ello: Sean $A = (A, {}^01, \dots, {}^0k)$ y $\beta = (B, {}^01, \dots, {}^0k)$ dos sistemas algebraicos de un mismo tipo, A y B son conjuntos y los elementos dentro de los paréntesis son las operaciones algebraicas definidas para esos conjuntos.

Como podemos observar, las propiedades de los sistemas algebraicos son casi los mismos que se aplican para el álgebra booleana que mencionamos en la introducción. Álgebra que en vez de manejar una cantidad desorbitante de números, solo va a trabajar con dos, basados en el trabajo de George Boole (1815 - 1864).

Álgebra Booleana

El álgebra booleana, o también conocida como álgebra de Boole, es un sistema algebraico basado en la lógica proposicional, utilizado para representar circuitos lógicos en forma de ecuaciones. Apoyándose a su vez, de la lógica binaria y la teoría de conjuntos. Fue desarrollada por el matemático y lógico británico George Boole durante el siglo XIX.

Como ya se mencionó, en este sistema algebraico de base binaria, nos indica que se van a manipular valores binarios, esto nos dice que son valores que pueden ser verdaderos (1) o falsos (0). Y al ser un sistema algebraico, se van a estipular ciertos reglamentos para su adecuado y lógico uso, utilizando operaciones como conjunción (AND), disyunción (OR) y la negación (NOT). Nos permite evaluar condiciones, tomar decisiones y controlar el flujo de ejecución de programas que queramos poner en marcha.

APLICACIONES DEL ÁLGEBRA DE BOOLE:

- Diseño de circuitos lógicos (digitales: procesadores, memorias y dispositivos lógicos programables).
- Programación y algoritmos (estructuras de control condicional).
- Diseño de hardware.

NOS PERMITE:

- Debido a sus leyes, simplificar expresiones booleanas complejas, facilitando su comprensión y análisis. (Como lo hemos visto en los distintos métodos de minimización, y ahora con Quine-McCluskey).
- Optimizar circuitos digitales. (Se reducen el número de compuertas lógicas, mejorando su eficacia.)

Y proporciona un marco matemático adecuado para el estudio de la lógica y el razonamiento lógico; asunto vital para la computación moderna.

LEYES

1. **Ley Comutativa** – Para la operación disyuntiva (OR) nos dice que el orden que ocupen las variables es indiferente. Lo representamos con como: \vee .
2. **Ley Asociativa** – Para la operación conjuntiva (AND) el orden que ocupen las variables es indiferente, Lo representamos como: \wedge .
3. **Ley Distributiva** – Para la operación disyuntiva (OR), en su resultado al tener un caso en que intervienen más de dos variables, va a ser independiente del modo en que se agrupen las variables.

Métodos de minimización

Ahora estamos teniendo un enfoque centrado a la búsqueda de representaciones algebraicas de lógica para representar desde circuitos hasta estructuras condicionales que por ende nos deja en un entorno de lógica con razón de dos respuestas en las que encontramos a verdadero (1) y falso (0). Por lo que encontraremos sistemas algebraicos en los que tendremos expresiones extensas y que complicarían la comprensión de sí mismas para uno. Por lo tanto, nos quedamos en la necesidad de recurrir a métodos que logren reducir las expresiones en álgebra de Boole, que ahora; por comodidad, vamos a llamar funciones booleanas.

Para nuestro curso hemos visto distintas formas de reducir una función booleana, siendo estas las más destacadas de las que se encuentran en el entorno computacional, en el área de lógica se refiere, las cuales serían reducciones por:

- Método algebraico
- Mapas de Karnaugh.
- Método de Quine-McCluskey

Claro, el criterio de minimización más usado es obteniendo una expresión en forma de suma de productos, que tenga un número mínimo de términos con el menor número de variables posible en cada uno de estos productos. Pero para ello, se ha de obtener la expresión en su forma ‘canónica’, como se hace referencia en algunos textos (Como en que consultamos: SISTEMAS ELECTRONICOS DIGITALES, de 1998). Ahora si podemos hablar del primer método de minimización.

Método algebraico

Para este método es a primera instancia, fácil de interpretar por su semejanza con los sistemas algebraicos a los que estamos acostumbrados. Sin embargo, aquí iremos aplicando los postulados (o propiedades) del álgebra de Boole.

Bueno, las propiedades que se estuvieron manejando durante el curso para la aplicación de este método se encuentran en la tabla VI que adjuntamos a continuación, en donde estamos manejando las representaciones de la conjunción, disyunción y negación en su forma (\cdot , $+$ y $'$, respectivamente).

Expresión	Equivalencia, ley o propiedad
$\neg\neg p = p$	Doble negación
$p + p = p$	Idempotencia
$p \cdot p = p$	
$p + q = q + p$	Commutativa
$p \cdot q = q \cdot p$	
$p + (q + r) = (p + q) + r$	Asociativa
$p \cdot (q \cdot r) = (p \cdot q) \cdot r$	
$p + (q \cdot r) = (p + q) \cdot (p + r)$	Distributiva
$p \cdot (q + r) = (p \cdot q) + (p \cdot r)$	
$p + \neg p = 1$	Tercero excluido Complemento Tautología
$p \cdot \neg p = 0$	Tercero excluido Complemento Contradicción
$p + 0 = p$	Identidad
$p \cdot 1 = p$	
$p + 1 = 1$	Dominancia o Dominante
$p \cdot 0 = 0$	
$\neg(p + q) = \neg p \cdot \neg q$ $\neg(p \cdot q) = \neg p + \neg q$	De Morgan
$p + (p \cdot q) = p$ $p \cdot (p + q) = p$	Absorción

Tabla VI. Leyes y propiedades (álgebra booleana)

1 corresponde a True

0 corresponde a False

Mapas de Karnaugh

Este método surge de tratar de agrupar a los términos de interés sin aplicar de forma directa el desarrollo algebraico que ofrece el método anterior. Y esto se constituye de desarrollar de forma gráfica la tabla de verdad de una función lógica.

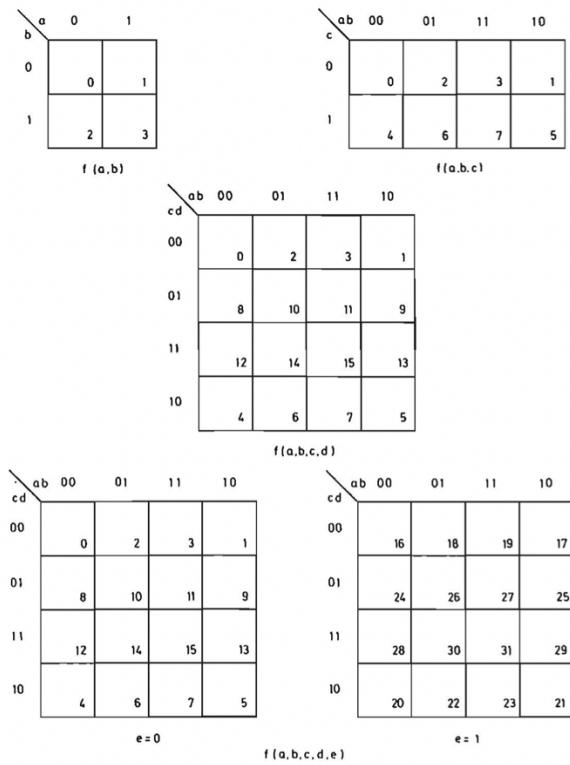


FIGURA 3.1.—Tablas de Karnaugh.

Imagen obtenida de la página 42 de Mandado, Enrique. (1998). SISTEMAS ELECTRONICOS DIGITALES.

Como podemos ver, se pueden tratar funciones de dos hasta cinco variables, para este método tabular cuyos términos canónicos adyacentes pueden ser agrupados para hacer de forma más sencilla la agrupación de los elementos.

Su algoritmo, en pocas palabras, consiste en:

1. Se escribe un 1 en los cuadros que tiene un número de los que forman la sumatoria de la función.
2. Se forman los grupos de minitérminos que se consideran vecinos (tienen un lado en común).
3. Se identifican a las variables para los conjuntos hechos por los grupos que hicimos de los minitérminos, estos serán los conjuntos.
4. Hacemos la sumatoria de los productos *Habrá que comprobar el resultado con el método gráfico de formar líneas dictadas por las variables que forman a los productos de los grupos*.

Método de minimización de Quine-McCluskey

Este método fue presentado originalmente por Willard Van Orman Quine y Edward J. McCluskey, denominado en su momento como método (Q-M), no limitado al número de variables de una función (como si lo pudiera llegar a ser para los mapas de Karnaugh), posibilita la implementación algorítmica y tiene la ventaja de realizar simplificaciones simultáneas para varias funciones que comparten un mismo conjunto de variables de entrada.

Este método entrega una solución minimizada global de la implementación, en vez de una solución mínima por función (los llamados mínimos locales).

Ejemplo de la implementación de Quine-McCluskey y mapas de Karnaugh.

Germán Andrés Holguín Londoño - Mauricio Holguín Londoño

5. Con base en los implicantes (o implicados) primos esenciales, identificar el total de mintérminos (o maxtérminos) cubiertos y no cubiertos por ellos.

6. Para los mintérminos (o maxtérminos) no cubiertos en el punto 5, identificar el mínimo conjunto de implicantes (o implicados) primos necesarios para cubrirlos.

7. La respuesta está compuesta del conjunto de implicantes (o implicados) primos esenciales identificados en el punto 4 y del conjunto de implicantes (o implicados) primos identificado en el punto 6.

EJEMPLO
Minimizar mediante el método de Quine-McCluskey la siguiente función:

$$f(A, B, C, D) = \sum m(2, 4, 6, 8, 9, 10, 12, 13, 15)$$

Para este ejemplo, además, se muestra la solución simultánea usando mapas VK con el método de I.P.E., con el fin de hacer el paralelo entre los dos algoritmos.

Partiendo con los pasos 1 y 2 del algoritmo de minimización por I.P.E., se obtienen todos los I.P., tal como se indica en la figura 4.22.

Por Q-M, los I.P. se obtienen producto de aplicar los pasos 1, 2 y 3 del respectivo algoritmo, así: inicialmente se lista en una columna los mintérminos agrupados según el número de unos. Para este caso, los mintérminos se agrupan en grupos de 1, 2, 3 y 4 unos. Seguidamente, se evalúan las vecindades, por ejemplo, el grupo 1 contra el grupo 2, donde m_2 es vecino

Figura 4.22: I.P. sobre mapa VK y Q-M

de m_6 formando la vecindad m_2m_6 , identificada en binario como 0 – 10. En esta vecindad, el guion identifica el bit que cambia y en la primera columna se identifica con el asterisco los mintérminos agrupados. En la tabla 4.5, la columna denominada 1-Cubo es el resultado final de comparación de vecindades de la columna de mintérminos. La columna denominada 2-Cubo, es el resultado de comparación de vecindades de la columna 1-Cubo. En la columna 2-Cubo se destaca que la vecindad $m_8m_9m_{12}m_{13}$ se obtiene dos veces, pero solo se anota una. Lo anterior sucede ya que, al agrupar en la columna 1-Cubo las vecindades m_8m_9 y $m_{12}m_{13}$, se obtiene lo mismo que al agrupar m_8m_{12} con m_9m_{13} .

Estructuras Discretas

Equipo 7

Principios y métodos combinatoriales en sistemas automáticos digitales

Tabla 4.5: Ejemplo Quine McCluskey por mintérminos, tabla de vecindades

Número de 1's	mintérminos	1-Cubo	2-Cubo
1	$m_2m_0010^*$ $m_4m_0100^*$ $m_8m_{1000}^*$	$m_2m_0 - 10IP2$ $m_2m_{10} - 010IP3$ $m_4m_01 - 01P4$ $m_4m_{12} - 100IP5$ $m_8m_{100} - *$ $m_8m_{10}m_{10} - 0IP6$ $m_8m_{12} - 00 *$	
2	$m_8m_{0110}^*$ $m_9m_{1001}^*$ $m_{10}m_{1010}^*$ $m_{12}m_{1100}^*$	$m_9m_{131} - 01^*$ $m_{12}m_{131} - 01^*$	$m_8m_9m_{12}m_{131} - 0 - IP1$
3	$m_{13}m_{1101}^*$		
4	$m_{15}m_{1111}^*$	$m_{13}m_{1511} - 1IP7$	

Al final del proceso anterior, se identifican las vecindades no agrupadas (sin asterisco) lo cual las convierte en implicantes primos. Estos IP. coinciden con los mismos identificados sobre el mapa VK de la figura 4.22.

Siguiendo con el paso 3 del algoritmo de minimización por I.P.E., se obtienen todos los implicantes primos que contienen un uno exclusivo, es decir, se identifican los I.P.E., tal como se muestra en la figura 4.23.

AB

	00	01	11	10
00		1	1	1
01			1	1
11			1	
10	1	1		1

CD

Figura 4.23: I.P.E. sobre mapa VK y Q-M

Este proceso de identificar los I.P.E. se realiza en los pasos 4 y 5 del algoritmo Q-M. Para lo anterior, se procede a realizar una nueva tabla con los implicantes primos en las filas y los mintérminos en las columnas. Se pone una "X" para identificar los mintérminos que cubre cada implicante primo. En la tabla 4.6, se puede observar que los mintérminos m_9 y m_{15} solo están cubiertos por un único implicante primo, IP1 e IP7 en este caso, lo cual los vuelve esenciales.

156

Estructuras Discretas

Equipo 7

Germán Andrés Holguín Londoño - Mauricio Holguín Londoño

Si IP1 e IP7 son esenciales, entonces hacen parte de la respuesta y se identifica con un asterisco los miníterminos que agrupan, que son m_8 , m_9 , m_{12} , m_{12} y m_{15} .

Tabla 4.6: Ejemplo Quine McCluskey por miníterminos, tabla de I.P.E.

IP	IPE	m_2	m_4	m_6	m_8	m_9	m_{10}	m_{12}	m_{13}	m_{15}
IP1, $m_8m_9m_{12}m_{13}1 - 0-$	IPE			*	*	X	X	X	X	*
IP2, $m_2m_60 - 10$		X		X						
IP3, $m_2m_{10} - 010$		X					X			
IP4, $m_4m_601 - 0$			X	X						
IP5, $m_4m_{12} - 100$			X					X		
IP6, $m_8m_{10}10 - 0$					X		X			
IP7, $m_{13}m_{15}11 - 1$	IPE								X	X

$f(A, B, C, D) = \sum m(1, 3, 5, 7, 11, 13, 15)$

Ahora, la figura 4.24 muestra la solución completa por mapa VK, la cual consta de los I.P.E. más el mínimo conjunto de I.P. necesarios para cubrir todos los miníterminos faltantes.

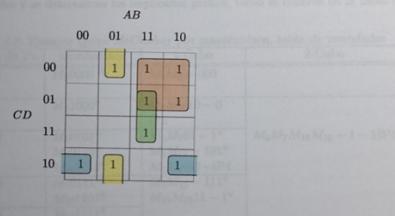


Figura 4.24: Solución sobre mapa VK Vs QM

Para identificar, por Q-M, los implicantes primos restantes, y obtener la solución respectiva, se siguen los pasos 6 y 7 del algoritmo. Este conjunto mínimo está formado por los implicantes primos IP3 e IP4, tal como se puede observar en la tabla 4.7, con los cuales se termina de cubrir todos los miníterminos.

Tabla 4.7: Ejemplo Quine McCluskey por miníterminos, miníterminos cubiertos

IP	IPE	m_2	m_4	m_6	m_8	m_9	m_{10}	m_{12}	m_{13}	m_{15}
IP1, $m_8m_9m_{12}m_{13}1 - 0-$	IPE	*	*	*	*	*	*	*	*	*
IP2, $m_2m_60 - 10$		X		X						
IP3, $m_2m_{10} - 010$	+		X				X			
IP4, $m_4m_601 - 0$	+		X	X						
IP5, $m_4m_{12} - 100$			X					X		
IP6, $m_8m_{10}10 - 0$					X		X			
IP7, $m_{13}m_{15}11 - 1$	IPE								X	X

Estructuras Discretas

Equipo 7

Principios y métodos combinatoriales en sistemas automáticos digitales

De la tabla anterior, se obtiene la solución completa así:

$$f = IP_1 + IP_7 + IP_3 + IPA = AC + ABD + BCD + \bar{A}BD$$

Una observación que se puede realizar sobre el algoritmo de minimización Q-M radica en que, en el paso 1, la numeración de min térmimos (o max térmimos) se puede hacer tanto por la cantidad de unos que contiene su representación binaria, así como por la cantidad de ceros. Lo anterior radica en el hecho que, numerar por la cantidad de ceros es único que hace es listar en sentido contrario al realizado por cantidad de unos.

A continuación, se muestra el procedimiento de minimización para una función en término de sus max térmimos.

EJEMPLO

Minimizar mediante el método de Quine-McCluskey la siguiente función:

$$f(A, B, C, D) = \prod M(0, 5, 6, 7, 8, 10, 13, 15)$$

Inicialmente, se lista en una columna los max térmimos según el número de unos. Para este caso, los max térmimos se agrupan en grupos de 0, 1, 2, 3 y 4 unos. Seguidamente, se evalúan las vecindades y se determinan los implicados primos, como se observa en la tabla 4.8.

Tabla 4.8: Ejemplo Quine McCluskey por max térmimos, tabla de vecindades

Número de 1's	min términos	1-Cubo		2-Cubo	
		$M_0M_8 - 000$ IP2	$M_6M_{10} - 0$ IP3	$M_5M_7M_{13}M_{15} - 1 - 1IP_1$	
0	M_0000^*				
1	M_81000^*				
2	M_50101^* M_60110^* $M_{10}1010^*$	$M_5M_701 - 1^*$ $M_5M_{13} - 101^*$ $M_6M_7011 - IP4$			
3	M_70111^* $M_{13}1101^*$	$M_7M_{15} - 111^*$ $M_{13}M_{15}11 - 1^*$			
4	$M_{15}1111^*$				

Ahora, se realiza una nueva tabla con los implicados primos en las filas y los max térmimos en las columnas. Se pone una "X" para identificar los max térmimos que cubren cada implicado primo. En la tabla 4.9 se puede observar que todos los implicados primos son esenciales.

Tabla 4.9: Ejemplo Quine McCluskey por max térmicos, tabla de cubierta

IP	IPE	M_0	M_5	M_6	M_7	M_8	M_{10}	M_{13}	M_{15}
$IP_1, M_5M_7M_{13}M_{15} - 1 - 1$	IPE	*	*	*	X			X	X
$IP_2, M_0M_8 - 000$	IPE	X				X			
$IP_3, M_8M_{10} - 0$	IPE					X	X		
$IP_4, M_6M_7011 -$	IPE			X	X				

158

Ejemplo obtenido de Holguín, Germán; et Holguín, Mauricio. (2021). *Principios y métodos combinatoriales en sistemas automáticos digitales*

A continuación, incluiremos un ejercicio con explicación escrita paso a paso para un mejor entendimiento de la implementación de este método de minimización, con base en la forma vista en clase.

EJEMPLO

En el siguiente ejemplo podremos observar cómo se lleva a cabo el método de Quine McCluskey para minimizar la siguiente expresión.

$f=(a,b,c,d) = \sum m(1,3,9,11)$ podemos observar que la función contiene 4 minitérminos

Nuestro paso **no.1** es generar nuestra Tabla 1:

minitérminos	1's	abcd	Comentario
1	1	0001	*
3	2	0011	*
9	2	1001	*
11	3	1011	*

Columna minitérminos: llenamos conforme a los minitérminos que nos muestra la función en este caso: 1,3,9,11

Columna 1's: es el número de 1's que contiene el número binario de ese minitérmino

abcd: el número binario que representa ese minitérmino
comentario:

Nuestro paso **no.2** es generar nuestra Tabla 2 que consiste en lo siguiente:

minitérminos	abcd		1's
1	0001	*	1
3	0011	*	2
9	1001	*	2
11	1011	*	3

En nuestra tabla 2 agruparemos nuestros minitérminos conforme al número de 1's que contengan en su número binario.

Nota: En la columna de comentarios de la tabla 1 iremos añadiendo un * a cada minitérmino ya utilizado en nuestra tabla 2 esto con la finalidad de tener un mejor control.

Nuestro paso **no.3** es generar nuestra Tabla 3 de la siguiente manera:

Combinaciones	abcd		1's
(1,3)	00-1	*	1
(1,9)	-001	*	1
(3,11)	-011	*	2
(9,11)	10-1	*	2

En nuestra tabla 3 haremos combinaciones con los minitérminos que tenemos en la tabla 2, es muy importante saber que solo pueden cambiar en una sola variable y que se debe de hacer de acuerdo con el número de 1's; primero los un 1's con los de dos 1's, los de dos 1's con los de tres 1's y sucesivamente.

Después intercambiaremos por un guion la variable que cambia en nuestra combinación y nuevamente llenamos la columna de 1's que hay en esa combinación.

Nota: importante marcar con * cada minitérmino utilizado en nuestra tabla 2

Estructuras Discretas
Equipo 7

El siguiente paso **no.4** es realizar nuestra Tabla 4 con las siguientes especificaciones:

Combinaciones	abcd	1's
(1,3,9,11)	-0-1	1

En este caso tenemos la combinación: (1,3,9,11) porque son combinaciones que ya existen (1,3) con (9,11) y también tenemos (1,9) con (3,11) entonces no las repetimos, y recordamos seguir la misma técnica que en la tabla 3 combinamos los un 1's con los 2 1's y así sucesivamente. Importante agregar un guion en la variable que cambie en nuestra combinación.

Nota: marcar con un * en nuestra tabla 3 los minitérminos que utilizamos en nuestras combinaciones

Nuestro paso **no.5** y último paso es generar nuestra Tabla 5 de la siguiente manera:

Combinaciones	abcd	1's	1 3 9 11	Producto
(1,3,9,11)	-0-1	1	x x x x	b'd

COLUMNAS TABLA 5

Combinaciones: Esta columna contiene las combinaciones que realizamos nuestra tabla 4.

abcd: Este es el número binario de nuestra combinación después de modificar la variable distinta con un guion.

1's: El número de 1's que contiene nuestro número binario dado por la combinación.

1,3,9,11: son todos los minitérminos de nuestra expresión, tiene una x para marcar que ya han sido utilizados.

Producto: Es el valor de las variables según nuestra columna abcd en la Tabla 5, recordemos que un guion nos indica que no es necesaria esa variable, un cero es el negado de la variable y un 1 es la misma variable.

En nuestro resultado final pondremos nuestra minimización dada por la suma de productos de la siguiente manera:

$$f = (a,b,c,d) = \sum m(1,3,9,11) = b'd$$

y listo 😊 .

CUESTIONARIO

- 1. ¿Cuáles son los dos métodos principales de minimización utilizados en la síntesis de circuitos lógicos?**
 - a) Mapa de Karnaugh y el algoritmo Quine-McCluskey (método tabula).
 - b) Método Simplex y Método algebraico
 - c) Método gráfico de mapas de Karnaugh y Algebra de Boole

- 2. ¿Cómo difieren en su aplicabilidad para funciones de más de 4 variables el mapa de Karnaugh y el método tabula (Quine-McCluskey) en la síntesis de circuitos lógicos?**
 - a) Mientras que el mapa de Karnaugh se vuelve complicado de analizar visualmente para funciones con más de 4 variables, el método tabula (Quine-McCluskey) es más adecuado y no presenta dificultades significativas al manejar funciones lógicas con un mayor número de variables.
 - b) Mientras más complejo el mapa de Karnaugh, más complejo se vuelve el método de Quine McCluskey.
 - c) No difieren en ningún momento

- 3. ¿Cuál es el Algoritmo Quine-McCluskey?**
 - a) El algoritmo Quine-McCluskey es un método de simplificación de funciones booleanas desarrollado por Willard Van Orman Quine y Edward J. McCluskey
 - b) Es un algoritmo de un método gráfico de simplificación de funciones logarítmicas desarrollado por Edward J. McCluskey y Bernhard Riemann
 - c) Es un método de simplificación de funciones desarrollado por Edward J. McCluskey y Grigori Perelman

- 4. ¿Cuáles son pasos principales en la simplificación de funciones booleanas? Elije 2**
 1. Involucra encontrar todos los implicants primos de la función.
 2. Consiste en identificar los implicants primos esenciales, que son necesarios y suficientes para generar la función simplificada.
 3. Consiste en encontrar todos los implicants irracionales que existen en la función
 4. Encontrar los elementos fraccionarios dentro de la función simplificada

5. ¿Qué es la representación cúbica de las funciones de Boole?

- a) La representación cúbica de las funciones de Boole es una técnica gráfica que utiliza cubos para visualizar las combinaciones posibles de variables booleanas. En esta representación, una variable se muestra como un punto en un segmento, dos variables en un cuadrado, y tres variables en un cubo.
- b) Es una técnica tabular que utiliza cubos para visualizar las combinaciones posibles de variables.
- c) Representación de una sola combinación de una variable booleana

6. ¿Qué son los implicantes primos en el contexto del método de Quine-McCluskey?

- a) Los implicantes primos son cubos que no están completamente contenidos en otros cubos más grandes de una función booleana. Estos cubos se utilizan para simplificar la función.
- b) Los implicantes primos son cubos que se encuentran contenidos en cubos más pequeños de una función booleana.
- c) Los implicantes primos son cubos que se encuentran contenidos en cubos más pequeños de una función booleana. Estos cubos se utilizan para simplificar la función

7. ¿Cómo se simplifican las funciones booleanas utilizando el método de Quine-McCluskey?

- a) El método de Quine-McCluskey simplifica funciones booleanas comparando mintérminos y combinándolos en cubos de orden superior. Se buscan los implicantes primos esenciales que cubren los mintérminos de la función, y luego se buscan implicantes primos secundarios de menor costo. La función simplificada se obtiene combinando estos implicantes primos.
- b) Se buscan implicantes compuestos al azar y luego implicantes primos esenciales, al combinarlos se obtiene la función simplificada
- c) Selecciona un grupo de implicantes primos de menor valor y otro grupo de estos mismos, al final se combinan y la función queda simplificada

8. ¿Qué papel juegan los cubos en la representación cúbica de funciones booleanas?

1. Los cubos se utilizan en la representación cúbica para visualizar las combinaciones posibles de variables booleanas. Cada cubo representa una combinación de variables y se utiliza para simplificar funciones booleanas mediante el método de Quine-McCluskey.

2. Cada puede representar distintas combinaciones de variables y se utiliza para simplificar funciones booleanas
 3. Son una representación cubica que visualiza una sola combinación de variables booleanas. Cada cubo es igual a diferentes combinaciones de variables y se utiliza en el método de Quine-McCluskey.
- 9. ¿Cuál es la ventaja de simplificar funciones booleanas utilizando el método de Quine- McCluskey?**
- a) Obtener expresiones más compactas y eficientes, lo que facilita su implementación en circuitos electrónicos y reduce la complejidad. Esto mejora el rendimiento y reduce el costo de los circuitos lógicos.
 - b) Obtener expresiones más complejas pero eficientes al implementarlas en circuitos eléctricos.
 - c) Un mejor rendimiento en los circuitos eléctricos, pero aumentando su costo
- 10. ¿Qué implicantes se consideran primos esenciales en el algoritmo de Quine- McCluskey?**
- a) Los implicantes primos esenciales son aquellos que no pueden ser reducidos más y que tienen solo un tache en la columna de números binarios.
 - b) Son aquellos que si se pueden reducir y que tienen múltiples taches en la columna de números binarios.
 - c) Aquellos que no pueden ser reducidos más y que tienen múltiples taches en la columna de números binarios
- 11. ¿Cuál es el propósito de la minimización de funciones booleanas en el diseño de circuitos lógicos?**
- a) La minimización de funciones booleanas es esencial en el diseño de circuitos lógicos ya que afecta la complejidad del sistema, su costo y su implementación. El objetivo es representar una función booleana como la suma del menor número de términos, lo que permite simplificar el diseño de circuitos lógicos.
 - b) Su objetivo es representar una función booleana como la resta del número mayor de términos, lo que nos daría un diseño más complejo, pero más útil.
 - c) No tienen ningún objetivo, simplemente es tener el circuito de manera visual.
- 12. ¿Qué ventajas se mencionan en la versión decimal del algoritmo Quine- McCluskey con la modificación propuesta?**
- a) Si importa con la necesidad de utilizar números adicionales en cada paso del algoritmo
 - b) La ventaja es que permite tratar aritméticamente los términos "No-Importa" sin la necesidad de utilizar números adicionales en cada paso del algoritmo.

- c) No tiene ninguna ventaja la versión decimal del algoritmo Quine McCluskey

13. ¿Cómo se lleva a cabo la simplificación de funciones booleanas en la versión decimal del algoritmo Quine-McCluskey?

- Lo términos se simplifican comparando sus valores decimales con los enteros y si la diferencia entre estos dos términos es múltiplo de 5, se genera un término reducido
- Los términos se simplifican comparando sus valores decimales, y si la diferencia entre dos términos es un múltiplo de 3, pueden generar un término reducido.
- Tomamos nuestro primer número decimal y si es múltiplo de 6, podemos generar su término reducido

13. ¿Cómo difieren en su aplicabilidad para funciones de más de 4 variables el mapa de Karnaugh y el método tabula (Quine-McCluskey) en la síntesis de circuitos lógicos?

- El método Quine Mccluskey presenta ciertas dificultades al manejar funciones lógicas con un menor número de variables.
- Mientras que el mapa de Karnaugh se vuelve complicado de analizar visualmente para funciones con más de 4 variables, el método tabula (Quine-McCluskey) es más adecuado y no presenta dificultades significativas al manejar funciones lógicas con un mayor número de variables.
- No difiere en ningún caso

14. ¿Qué es la representación cúbica de las funciones de Boole?

- La representación cónica de las funciones de Boole es una técnica gráfica que utiliza cubos para visualizar las combinaciones posibles de variables booleanas.
- Es una representación tabular de las funciones de Boole que visualiza las combinaciones posibles de diferentes variables
- Es una técnica tabular que utiliza cubos para visualizar una sola variable booleana

15. ¿Qué son los implicantes primos en el contexto del método de Quine-McCluskey?

- Los implicantes primos son cubos que no están completamente contenidos en otros cubos más grandes de una función booleana. Estos cubos se utilizan para simplificar la función.
- Los implicantes son cubos contenidos en cubos más pequeños de una función booleana. Se utilizan para simplificar la función
- Cubos contenidos en cubos más grandes de una función y estos cubos hacen más compleja nuestra función, pero con más opciones para trabajarla.

16. ¿Cómo se simplifican las funciones booleanas utilizando el método de Quine-McCluskey?

1. comparando minitérminos y combinándolos en cubos de orden superior. Se buscan los implicantes primos esenciales que cubren los mintérminos de la función, y luego se buscan implicantes primos secundarios de menor costo. La función simplificada se obtiene combinando estos implicantes primos.
2. No podemos simplificarlas, simplemente tenerlas de manera visual.
3. Combinando los implicantes primos.

17. ¿Qué papel juegan los cubos en la representación cúbica de funciones booleanas? Respuesta:

1. Los cubos se utilizan en la representación cónica para visualizar las combinaciones posibles de variables booleanas. Cada cubo representa una combinación de variables y se utiliza para simplificar funciones booleanas mediante el método de Quine-McCluskey.
2. Cada cubo representa distintas combinaciones de variables y nos muestra las funciones de manera más compleja.
3. Los cubos nos muestran funciones más complejas, pero más útiles para trabajarlas.

18. ¿Cuál es la ventaja de simplificar funciones booleanas utilizando el método de Quine- McCluskey?

1. La ventaja de simplificar funciones booleanas con el método de Quine-McCluskey es obtener expresiones más compactas y eficientes, lo que facilita su implementación en circuitos electrónicos y reduce la complejidad. Esto mejora el rendimiento y reduce el costo de los circuitos lógicos.
2. No hay ninguna ventaja, al contrario, es más complejo utilizar el método de Quine- McCluskey.
3. La ventaja de tener una función booleana más compleja pero más sencilla su implementación en circuitos electrónicos

19. ¿Qué implicantes se consideran primos esenciales en el algoritmo de Quine-McCluskey?

1. Son aquellos que no pueden ser reducidos más y que tienen solo un tache en la columna de números binarios.
2. Son aquellos que podemos reducir más y tienen múltiples taches en la columna de números binarios
3. Son aquellos que podemos reducir más y que tienen un solo tache en su columna de números binarios.

20. ¿Cuál es el propósito de la minimización de funciones booleanas en el diseño de circuitos lógicos?

1. La minimización de funciones booleanas es esencial en el diseño de circuitos lógicos ya que afecta la complejidad del sistema, su costo y su

implementación. El objetivo es representar una función booleana como la suma del menor número de términos, lo que permite simplificar el diseño de circuitos lógicos.

2. Su objetivo es representar una función booleana como el producto del mayor número de términos, lo que permite simplificar el diseño de los circuitos lógicos.
3. No existe ningún propósito

21. ¿Quiénes fueron los desarrolladores del Algoritmo de Quine-McCluskey?

Respuesta:

- a) El Algoritmo de Quine-McCluskey fue desarrollado por Willard Van Orman Quine y Edward J. McCluskey. (Correcta)
- b) El Algoritmo de Quine-McCluskey fue desarrollado por George Boole y John McCluskey.
- c) El Algoritmo de Quine-McCluskey fue desarrollado por Alan Turing y Charles Babbage.

22. ¿Cuál es el propósito principal del Algoritmo de Quine-McCluskey en la simplificación de funciones booleanas?

Respuesta:

- a) El propósito principal del Algoritmo de Quine-McCluskey es simplificar funciones booleanas, reduciendo la complejidad y el número de términos en sus expresiones. (Correcta)
- b) El propósito principal del Algoritmo de Quine-McCluskey es aumentar la complejidad de las funciones booleanas.
- c) El propósito principal del Algoritmo de Quine-McCluskey es generar funciones booleanas más largas y complejas.

23. ¿Cuál es la entrada típica para el Algoritmo de Quine-McCluskey?

Respuesta:

- a) La entrada típica para el Algoritmo de Quine-McCluskey son los mintérminos de una función booleana que se desea simplificar. (Correcta)
- b) La entrada típica para el Algoritmo de Quine-McCluskey son los números primos de una función booleana.
- c) La entrada típica para el Algoritmo de Quine-McCluskey son los números enteros de una función booleana.

24. ¿Qué es un mintérmino en el contexto del Algoritmo de Quine-McCluskey?

Respuesta:

- a) Un mintérmino es una expresión booleana que consiste en la productoria (AND) de todas las variables de entrada en su forma original o complementada. (Correcta)

- b) Un mintérmino es una expresión booleana que consiste en la suma (OR) de todas las variables de entrada en su forma original o complementada.
- c) Un mintérmino es una expresión booleana que consiste en la resta (RESTA) de todas las variables de entrada en su forma original o complementada.

25. ¿Cuál es el primer paso del Algoritmo de Quine-McCluskey?

Respuesta:

- a) El primer paso del Algoritmo de Quine-McCluskey implica encontrar todos los implicantes primos de la función booleana. (Correcta)
- b) El primer paso del Algoritmo de Quine-McCluskey consiste en simplificar la función booleana de inmediato.
- c) El primer paso del Algoritmo de Quine-McCluskey es buscar los implicantes primos esenciales.

26. ¿Qué son los implicantes primos en el contexto del Algoritmo de Quine-McCluskey?

Respuesta:

- a) Los implicantes primos son términos irreducibles que cubren una o varias combinaciones de mintérminos en la función booleana. (Correcta)
- b) Los implicantes primos son términos que no tienen relación con los mintérminos en la función booleana.
- c) Los implicantes primos son términos complementarios a los mintérminos en la función booleana.

27. ¿Cuál es el segundo paso del Algoritmo de Quine-McCluskey?

Respuesta:

- a) El segundo paso del Algoritmo de Quine-McCluskey consiste en identificar los implicantes primos esenciales, que son necesarios y suficientes para generar la función simplificada. (Correcta)
- b) El segundo paso del Algoritmo de Quine-McCluskey es combinar todos los términos de la función en una única expresión.
- c) El segundo paso del Algoritmo de Quine-McCluskey es buscar los mintérminos que no son necesarios para la función simplificada.

28. ¿Qué se hace después de encontrar los implicantes primos esenciales en el Algoritmo de Quine-McCluskey?

Respuesta:

- a) Después de encontrar los implicantes primos esenciales, se combinan para obtener la expresión simplificada de la función booleana. (Correcta)
- b) Después de encontrar los implicantes primos esenciales, se eliminan para reducir la complejidad de la función booleana.

- c) Después de encontrar los implicantes primos esenciales, se utilizan como entrada en otro algoritmo para simplificar aún más la función booleana

29. ¿Cuál es el beneficio de utilizar el Algoritmo de Quine-McCluskey en lugar de otras técnicas de simplificación de funciones booleanas?

Respuesta:

- a) El beneficio de utilizar el Algoritmo de Quine-McCluskey en lugar de otras técnicas de simplificación de funciones booleanas es que produce simplificaciones óptimas, lo que significa que proporciona la expresión más reducida y eficiente de una función booleana. (Correcta)
- b) El beneficio de utilizar el Algoritmo de Quine-McCluskey es que es más rápido que otras técnicas, aunque no siempre produce la simplificación más eficiente.
- c) El beneficio de utilizar el Algoritmo de Quine-McCluskey es que es fácil de aprender y aplicar, lo que lo hace adecuado para principiantes en electrónica digital.

30. ¿Qué ocurre si no se encuentran implicantes primos esenciales en el Algoritmo de Quine-McCluskey?

Respuesta:

- a) Si no se encuentran implicantes primos esenciales, se deben buscar implicantes primos secundarios para obtener una simplificación adecuada. (Correcta)
- b) Si no se encuentran implicantes primos esenciales, la función no se puede simplificar utilizando el Algoritmo de Quine-McCluskey.
- c) Si no se encuentran implicantes primos esenciales, se debe agregar más variables a la función para permitir la simplificación.

31. ¿Cuál es la complejidad computacional del Algoritmo de Quine-McCluskey en términos de tiempo?

Respuesta:

- a) La complejidad computacional del Algoritmo de Quine-McCluskey es exponencial en el peor de los casos, lo que significa que puede ser lento para funciones con un gran número de variables. (Correcta)
- b) La complejidad computacional del Algoritmo de Quine-McCluskey es lineal en términos de tiempo, lo que lo hace rápido y eficiente.
- c) La complejidad computacional del Algoritmo de Quine-McCluskey es constante, lo que significa que su rendimiento es constante independientemente del tamaño de la función.

32. ¿Qué modificaciones se han propuesto para mejorar el Algoritmo de Quine-McCluskey?

Respuesta:

- a) Se han propuesto modificaciones, como la versión decimal del algoritmo, que simplifican el tratamiento de los términos "No-Importa". (Correcta)
- b) Se han propuesto modificaciones para aumentar la complejidad del Algoritmo de Quine-McCluskey y hacerlo más eficiente.
- c) Se han propuesto modificaciones para eliminar completamente los términos "No-Importa" del algoritmo.

33. ¿Qué ventajas se mencionan en la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta?

Respuesta:

- a) La ventaja de la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta es que permite tratar aritméticamente los términos "No-Importa" sin la necesidad de utilizar números adicionales en cada paso del algoritmo. (Correcta)
- b) La ventaja de la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta es que simplifica la identificación de los implicantes primos esenciales.
- c) La ventaja de la versión decimal del Algoritmo de Quine-McCluskey con la modificación propuesta es que elimina por completo los términos "No-Importa".

34. ¿Cómo se lleva a cabo la simplificación de funciones booleanas en la versión decimal del Algoritmo de Quine-McCluskey?

Respuesta:

- a) En la versión decimal del Algoritmo de Quine-McCluskey, los términos se simplifican comparando sus valores decimales, y si la diferencia entre dos términos es un múltiplo de 3, pueden generar un término reducido. (Correcta)
- b) En la versión decimal del Algoritmo de Quine-McCluskey, los términos se simplifican eliminando los valores decimales y reduciendo la función directamente.
- c) En la versión decimal del Algoritmo de Quine-McCluskey, los términos se simplifican mediante una representación en base 10 de los valores booleanos, lo que facilita la simplificación.

35. ¿Cuál es la principal desventaja del Algoritmo de Quine-McCluskey?

Respuesta:

- a) La principal desventaja del Algoritmo de Quine-McCluskey es su complejidad exponencial en el peor de los casos, lo que lo hace ineficiente para funciones con muchas variables. (Correcta)
- b) La principal desventaja del Algoritmo de Quine-McCluskey es su falta de precisión en la simplificación de funciones booleanas.

- c) La principal desventaja del Algoritmo de Quine-McCluskey es su incompatibilidad con la representación decimal de funciones booleanas.

36. ¿En qué aplicaciones se utiliza comúnmente el Algoritmo de Quine-McCluskey?

Respuesta:

- a) El Algoritmo de Quine-McCluskey se utiliza en el diseño de circuitos lógicos, en la simplificación de funciones booleanas en electrónica digital y en la optimización de expresiones lógicas. (Correcta)
- b) El Algoritmo de Quine-McCluskey se utiliza en aplicaciones médicas para el análisis de datos clínicos.
- c) El Algoritmo de Quine-McCluskey se utiliza en la industria de la construcción para el cálculo de estructuras.

37. ¿Cuál es el objetivo final del Algoritmo de Quine-McCluskey?

Respuesta:

- a) El objetivo final del Algoritmo de Quine-McCluskey es encontrar una expresión simplificada que represente de manera eficiente una función booleana dada. (Correcta)
- b) El objetivo final del Algoritmo de Quine-McCluskey es encontrar una expresión que aumente la complejidad de una función booleana.
- c) El objetivo final del Algoritmo de Quine-McCluskey es encontrar todas las posibles combinaciones de variables en una función booleana.

38. ¿Qué es el principio fundamental detrás del Algoritmo de Quine-McCluskey?

Respuesta:

- a) El principio fundamental detrás del Algoritmo de Quine-McCluskey es encontrar cubos que cubran la máxima cantidad de mintérminos en la función y combinarlos para obtener la simplificación óptima. (Correcta)
- b) El principio fundamental detrás del Algoritmo de Quine-McCluskey es buscar la mayor cantidad de términos complementarios en la función para aumentar su complejidad.
- c) El principio fundamental detrás del Algoritmo de Quine-McCluskey es dividir la función en segmentos iguales para simplificarla de manera eficiente.

39. ¿Cómo se compara el Algoritmo de Quine-McCluskey con el mapa de Karnaugh en términos de aplicabilidad y eficiencia?

Respuesta:

- a) El Algoritmo de Quine-McCluskey es más adecuado para funciones con un gran número de variables, mientras que el mapa de Karnaugh es más eficiente para funciones con pocas variables. (Correcta)

- b) El Algoritmo de Quine-McCluskey es más adecuado para funciones con pocas variables, mientras que el mapa de Karnaugh es más eficiente para funciones con un gran número de variables.
- c) El Algoritmo de Quine-McCluskey y el mapa de Karnaugh tienen la misma aplicabilidad y eficiencia en todos los casos.

40. ¿Cuál es la importancia de la simplificación de funciones booleanas en el diseño de circuitos lógicos?

Respuesta:

- a) La simplificación de funciones booleanas es crucial para reducir la complejidad y el costo de los circuitos lógicos, lo que mejora su rendimiento y eficiencia. (Correcta)
- b) La simplificación de funciones booleanas no tiene importancia en el diseño de circuitos lógicos, ya que no afecta la complejidad ni el costo.
- c) La simplificación de funciones booleanas aumenta la complejidad y el costo de los circuitos lógicos, lo que mejora su rendimiento. y facilita

41. ¿Cuál es la función principal de una puerta lógica AND?

Respuesta

- a) La función principal de una puerta lógica AND es realizar la operación de multiplicación lógica.
- b) La función principal de una puerta lógica AND es realizar la operación de suma lógica.
- c) La función principal de una puerta lógica AND es realizar la operación de negación lógica.

42. ¿Cómo se representa una puerta lógica OR en un circuito eléctrico?

Respuesta:

- a) Una puerta lógica OR se representa mediante un símbolo con dos entradas y una salida que realiza la operación de suma lógica.
- b) Una puerta lógica OR se representa mediante un símbolo con una entrada y una salida que realiza la operación de multiplicación lógica.
- c) Una puerta lógica OR se representa mediante un símbolo con tres entradas y dos salidas.

43. ¿Cuál es la diferencia entre un circuito combinacional y un circuito secuencial?

Respuesta:

- a) Un circuito combinacional realiza operaciones lógicas basadas únicamente en las entradas actuales, mientras que un circuito secuencial tiene memoria y su salida depende de las entradas y el estado anterior.

- b) La diferencia entre un circuito combinacional y un circuito secuencial radica en la complejidad de las operaciones lógicas.
- c) Un circuito combinacional tiene memoria, mientras que un circuito secuencial opera solo con entradas actuales.

44. ¿Qué es una tabla de verdad en el contexto de circuitos lógicos?

Respuesta:

- a) Una tabla de verdad es una representación tabular que muestra todas las posibles combinaciones de entradas y las salidas correspondientes de una función booleana.
- b) Una tabla de verdad es una herramienta utilizada para programar microcontroladores.
- c) Una tabla de verdad es una representación gráfica de un circuito lógico.

45. ¿Por qué es importante la simplificación de expresiones booleanas en la electrónica digital?

Respuesta:

- a) La simplificación de expresiones booleanas reduce la complejidad de los circuitos y ahorra recursos, como tiempo y espacio.
- b) La simplificación de expresiones booleanas aumenta la complejidad de los circuitos y mejora su rendimiento.
- c) La simplificación de expresiones booleanas no tiene impacto en la electrónica digital.

46. ¿Qué son los mapas de Karnaugh y cómo se utilizan en la simplificación de funciones booleanas?

Respuesta:

- a) Los mapas de Karnaugh son herramientas gráficas que ayudan a simplificar funciones booleanas al identificar patrones de 1s en la tabla de verdad.
- b) Los mapas de Karnaugh son un tipo de dispositivo de almacenamiento de datos.
- c) Los mapas de Karnaugh son utilizados únicamente para aumentar la complejidad de las funciones booleanas.

47. ¿Cuál es la importancia de la minimización de funciones booleanas en el diseño de circuitos lógicos?

Respuesta:

- a) La minimización de funciones booleanas es importante para reducir la complejidad y el costo de los circuitos lógicos, lo que mejora su rendimiento y eficiencia.
- b) La minimización de funciones booleanas aumenta la complejidad de los circuitos lógicos y los hace menos eficientes.

- c) La minimización de funciones booleanas solo es relevante en la programación de software.

48. **¿Cómo se utilizan los operadores lógicos AND, OR y NOT en el Algoritmo de Quine-McCluskey para simplificar funciones booleanas?**

Respuesta:

- a) Los operadores lógicos AND se utilizan para identificar términos comunes en las mintérminos, los operadores OR se utilizan para combinar términos complementarios, y los operadores NOT se utilizan para invertir variables.
- b) Los operadores lógicos AND se utilizan para invertir variables en el Algoritmo de Quine-McCluskey.
- c) Los operadores lógicos AND se utilizan para combinar términos complementarios en el Algoritmo de Quine-McCluskey.

49. **¿Cuál es el papel de los operadores lógicos en la representación de funciones booleanas en el contexto del Algoritmo de Quine-McCluskey? Respuesta:**

- a) Los operadores lógicos son fundamentales para definir la relación entre las variables en una función booleana y permiten expresar las operaciones lógicas necesarias para la simplificación.
- b) Los operadores lógicos no tienen ningún papel en la representación de funciones booleanas en el Algoritmo de Quine-McCluskey.
- c) Los operadores lógicos solo se utilizan en la implementación de circuitos, no en la representación de funciones booleanas.

50. **¿En qué medida los operadores lógicos afectan la eficiencia y la complejidad de la simplificación de funciones booleanas utilizando el Algoritmo de Quine-McCluskey?**

Respuesta:

- a) Los operadores lógicos adecuados permiten simplificar funciones booleanas de manera eficiente y reducir la complejidad, mientras que un uso inadecuado puede aumentar la complejidad y dificultar la simplificación.
- b) Los operadores lógicos no tienen ningún impacto en la eficiencia y la complejidad de la simplificación de funciones booleanas.
- c) El uso de operadores lógicos siempre reduce la complejidad de la simplificación de funciones booleanas en el Algoritmo de Quine-McCluskey.

Bibliografía

Fabián Espino, Alejandro Alberto. (2003, octubre). *Método alternativo para la reducción de funciones por polinomios Reed-Muller*. Rescatado de: <https://rd.udb.edu.sv/items/16e406d4-ef73-4dc6-9639-17be47da0c97>

Dr. Ing. D. Assandri, Armando. (2008). *Microprogramación*. Facultad de Ingeniería, Universidad Nacional de San Juan. Rescatado de: <http://dea.unsj.edu.ar/sisdig2/Microprogramaci%F3n%20V3%200.pdf>

García, Javier; Angulo Martínez, Ignacio; et Angulo Usategui, José. (2007). *Sistemas digitales y tecnología de computadores*. Capítulo 3. Álgebra de Boole. Rescatado de:

<https://books.google.es/books?hl=es&lr=&id=i8eX0aMzmzcC&oi=fnd&pg=PP1&dq=libro+métodos+de+minimización+de+funciones+booleanas&ots=4OLVgmdfpc&sig=aBcXMuHnCiQubZvcr0aZRGes9-M#v=onepage&q=false>

Oliver, Joan; et Ferrer, Carles. (1998). *Diseño de sistemas digitales: introducción práctica*. 1.9 Minimización por Quine-McCluskey. En: <https://books.google.es/books?hl=es&lr=&id=jACme7V3Q2IC&oi=fnd&pg=PA3&dq=libro+métodos+de+minimización+de+funciones+booleanas&ots=GoxG8DdkvP&sig=bHMIfiUvsE5aPkOyOC7il3I6opw#v=onepage&q&f=true>

Morris Mano, M. (2003). *Diseño Digital*. Tercera edición. 2 Algebra Booleana y compuertas lógicas. Rescatado de: http://aniei.org.mx/paginas/uam/Descargas/Recursos/Diseno_Digital_de_Morris_Mano.pdf

ANGULO, JOSE. (1991). *ELECTRÓNICA DIGITAL MODERNA. Teoría y Práctica*. Editorial Paraninfo sa. Rescatado de: https://virtual.unju.edu.ar/pluginfile.php/147533/mod_resource/content/8/%5BAngulo-91%5D%20Electronica%20Digital%20Moderna_CAP%203.pdf

Falconi, Francisco. (2013). *TEMAS SELECTOS DE MATEMÁTICAS DISCRETAS*. 1ra Edición. Tabasco, México. Rescatado de: <https://ri.ujat.mx/bitstream/200.500.12107/4015/1/Temas%2Bselectos%2Bde%2Bmatema%CC%81ticas%2Bdiscretas.pdf>

Mandado, Enrique. (1998). *SISTEMAS ELECTRÓNICOS DIGITALES*. 8va Edición. Rescatado de: <https://es.scribd.com/document/642718969/Sistemas-Electronicos-Digitales-Enrique-Mandado-pdf>

Estructuras Discretas
Equipo 7

Holguín, Germán; et Holguín, Mauricio. (2021). *Principios y métodos combinatoriales en sistemas automáticos digitales*. Rescatado de: <https://repositorio.utp.edu.co/items/c994bddc-c06c-4ec0-b7ee-89a8e7a8b700>

Artículos:

Sánchez, Narcisa; Villagómez, Daniela. (2019, 29 de noviembre). “UTILIZACIÓN DE LOS MAPAS DE KARNAUGH COMO ESTRATEGIA DIDÁCTICA PARA EL APRENDIZAJE DE LA SIMPLIFICACIÓN DE FUNCIONES BOLEANAS CON LOS ESTUDIANTES DE OCTAVO SEMESTRE DE LA CARRERA DE CIENCIAS EXACTAS PERÍODO ABRIL-AGOSTO 2019”. Rescatado de: <http://dspace.unach.edu.ec/handle/51000/6228>

Canto, María. (1983). *Estudio sistemático de circuitos lógicos aritméticos: multiplicadores e integradores digitales*. Rescatado de: <https://www.proquest.com/openview/abf74f2a62e2c9fcf89fc50d11245c9d/1?pq-origsite=gscholar&cbl=2026366&diss=y>

Escartín, Rubén. (2017). *Una aproximación a la optimización de algoritmos mediante el uso de minimización de funciones booleanas*. Rescatado de: <https://zaguan.unizar.es/record/61415>

Reynoso, Gilberto. (2005). *Una propuesta en la implementación del algoritmo Quine-McCluskey para la minimización de funciones booleanas*. Rescatado de: <https://www.angelfire.com/extreme/greynosom/Correcciones/AMCA05069.pdf>

Martín, Sergio. (2005). Algoritmos alternativos para la simplificación de funciones booleanas. UFG Editores. Rescatado de: <http://ri.ufg.edu.sv/jspui/handle/11592/8407>