UNIDAD 5:
Diagramas de Decisión Binaria (BDD)

(86:44) Técnica Digital Avanzada- Unidad 5.

Profesor: Ing. Miguel Antonio Martínez.

Diagramas de decisión binaria (BDD).

En esta parte desarrollaremos el tema de Binary Decision Diagram. Hasta ahora hemos discutido mucho la representación de funciones binarias, canónicas y no canónicas. Muchos algoritmos de síntesis y verificación manejan grandes funciones de conmutación. Por lo tanto, es importante contar con formas eficientes de representación y gestión para manipular estas funciones booleanas.

En los últimos tiempos los diagramas de decisión binarios (BDD) han surgido como la representación de elección para muchas aplicaciones.

Aunque los BDD son relativamente viejos fue con el trabajo de Randal Bryant que sacó a relucir sus ventajas como representación canónica, que comenzaron a atraer la atención de muchos investigadores.

El impacto de los BDD ha sido enorme. En la síntesis de circuitos, los BDD han penetrado en prácticamente todos los subcampos en las áreas de diseño y verificación.

Como vemos, los BDD tienen dos propiedades notables. Primero, son canónicos. Así si se construyen correctamente los BDD para dos circuitos, estos son equivalentes si y solo si los BDD son idénticos. La segunda ventaja es que son increíblemente efectivos en la representación de conjuntos combinatorios grandes. Esto ha llevado a muchos avances en la minimización de lógica de dos niveles.

Ya hemos discutido varias formas de representar una función booleana, por ejemplo, con minitérminos o maxitérminos. Estas dos formas son canónicas. Una función es canónica si la representación de la función en cualquiera de las dos formas es única. Recordar que hablamos de funciones canónicas cuando en todos los términos aparecen todas las variables independientes de la función (negadas o sin negar), independientemente que los términos canónicos sean tipo suma o tipo producto.

Una función canónica es deseable porque facilita las pruebas de equivalencia. Desde esta definición diremos que si dos funciones f_1 y f_2 tiene la misma representación canónica si y solo si $f_1 = f_2$. O sea son equivalentes porque representan la misma tabla de verdad.

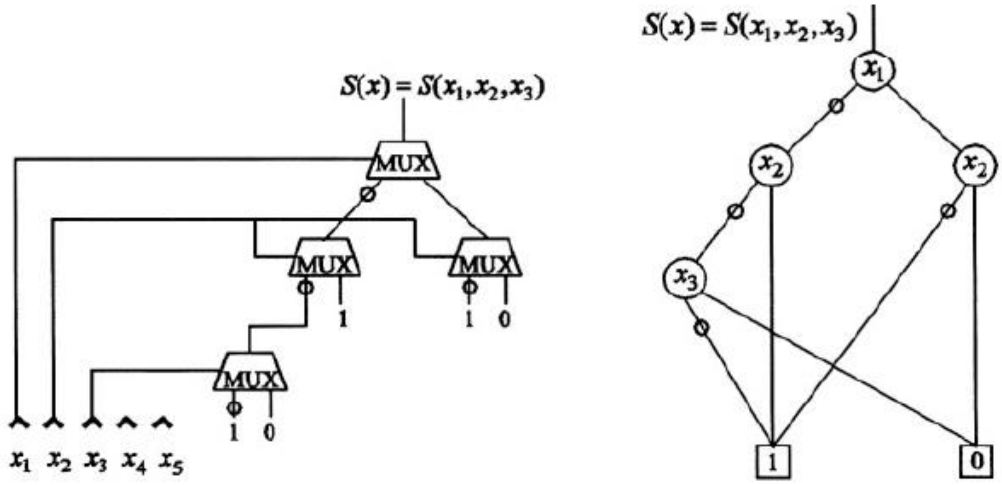
Sin embargo, la representación de funciones en minitérminos y maxitérminos tiene un inconveniente. Las representaciones suelen ser muy grandes ya que su tamaño crece en forma exponencial al aumentar el número de variables independientes.

Entre las formas no canónicas, el tipo de funciones conocido como suma de productos (SOP) o producto de sumas (PDS) han sido ampliamente utilizadas. Son representaciones de dos niveles pero que también tiene algunos inconvenientes.

- Las representaciones de dos niveles de algunas funciones son demasiado grandes para ser prácticas (por ejemplo, or exclusiva).
- Pasar de SDP a SDP o viceversa es difícil, como consecuencia por ejemplo realizar el complemento es difícil.
- Dado que tanto SDP como PDS no son canónicas, estudiar la equivalencia se complica.

Los diagramas BDD tiene la ventaja de ser compacto para muchas funciones y definitivamente superior a otras formas canónicas.

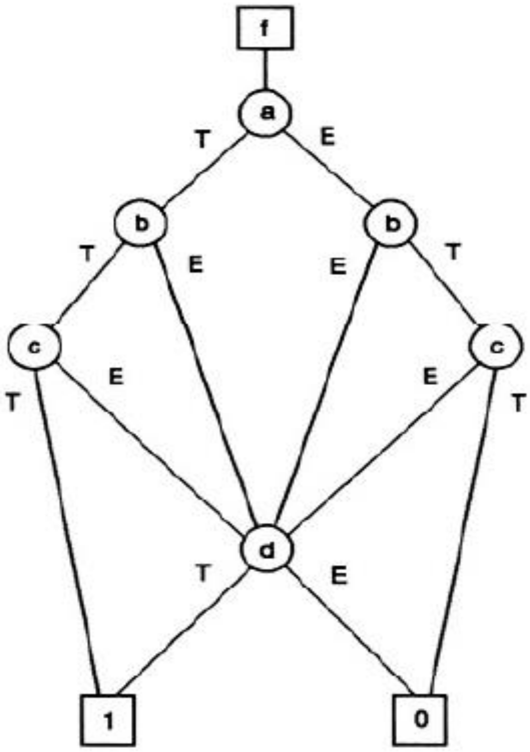
Primero veremos un ejemplo, antes de eso diremos que un BDD es un **Grafo Dirigido Acíclico** (tema que desarrollaremos después en el tema Grafos). Comparemos un gráfico BDD con un arreglo de Multiplexores. Nótese que cada nodo del gráfico está en correspondencia con cada entrada de control de los MUX.



Como segundo ejemplo consideramos la siguiente función que está expresada como Suma de Productos:

$$f = a.b.c + \bar{b}.d + \bar{c}.d$$

El correspondiente diagrama se ve en la siguiente figura:



Si queremos conocer el valor de una asignación particular para las variables a, b, c y d debemos seguir la correspondiente ruta desde el casillero cuadrada denominado “f”, que es nodo raíz de este BDD. Supongamos que nosotros queremos saber la asignación:

f (1, 0, 1, 0)

La primera variable encontrada desde la raíz es “a”, cuyo valor es 1. Entonces debemos seguir el borde etiquetado T (que significa entonces). Después nos encontramos con un nodo etiquetado como “b”. Dado que su valor es 0 seguimos el borde etiquetado como E (que significa de otro modo). El siguiente nodo está etiquetado como “d”, que implica que para a = 1 y b= 0, la función no depende de “c”. Siguiendo el borde E finalmente llegamos a la etiqueta 0. Esto nos dice que la función es 0, como puede verificarse fácilmente en la formula SOP.

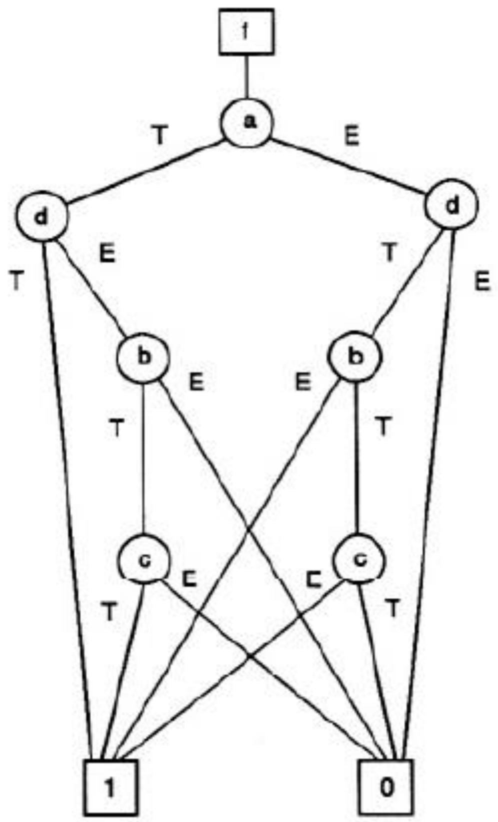
El diagrama BDD de este ejemplo es un **diagrama de decisión binario ordenado**, porque las variables aparecen en el mismo orden a lo largo de todas las rutas desde la raíz hasta el final. El ordenamiento de las variables es el siguiente:

$$a \leq b \leq c \leq d$$

La apariencia y el tamaño de los diagramas BDD dependen del orden de las variables. Si ordenamos las variables de la siguiente manera:

$$a \leq d \leq b \leq c$$

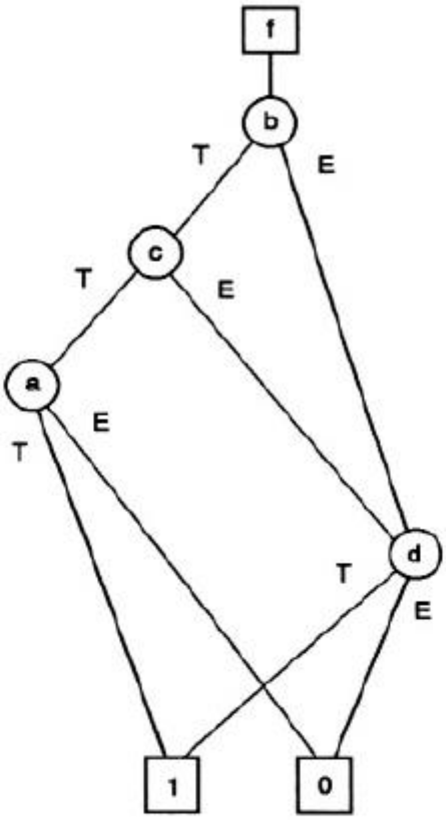
Tenemos en siguiente diagrama:



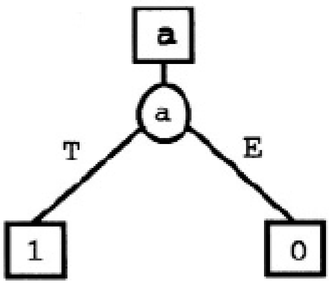
Finalmente, si las ordenamos de la forma

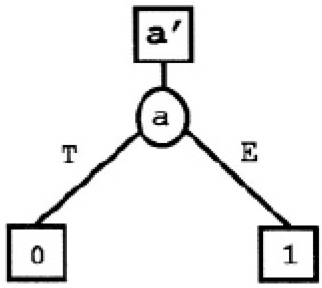
$$b \leq c \leq a \leq d$$

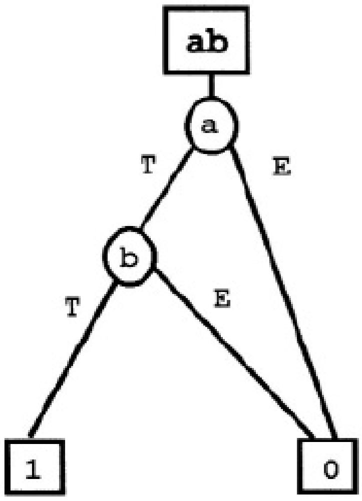
Obtenemos lo siguiente:

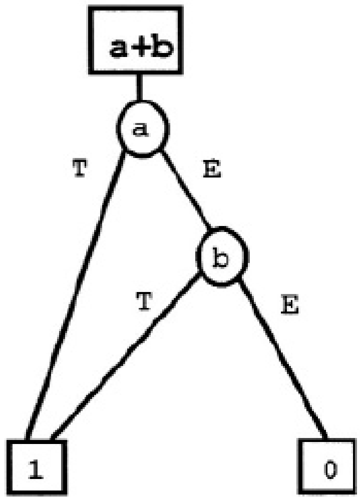


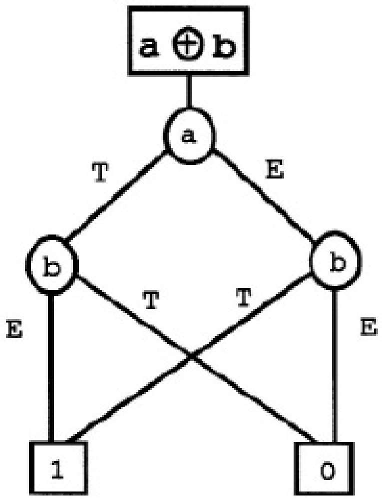
Este es un orden óptimo, ya que tenemos exactamente un nodo para cada variable. Cuando no se especifique lo contrario, asumiremos que nuestros BDD son ordenados. En las siguientes figuras mostramos algunos BDD para funciones elementales. Nótese la similitud de los BDD para f = a y para f = $\bar{a}$. Una se obtiene de la otra intercambiando los dos nodos terminales.

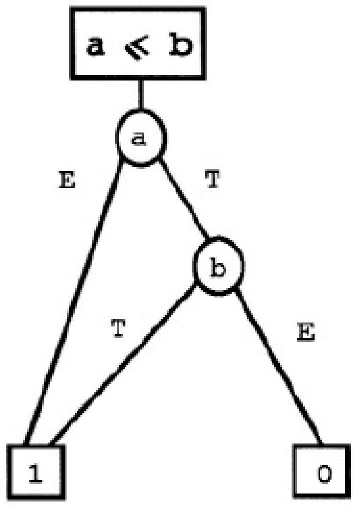












<u>Definición formal:</u>

Un BDD es un gráfico dirigido acíclico ($V \cup \Phi \cup \{1\}$, E) representando una función de conmutación de salida múltiple F. Los nodos se dividen en tres subconjuntos. V es el conjunto de nodos internos. El grado de salida de $v \in V$ es 2. Cada nodo tiene una etiqueta $l(v) \in S_F$. Aquí $S_F = \{x_1, \ldots., x_n\}$ denota el conjunto de variables que depende realmente F. Así, $l(v)$ es una de las variables $\{x_i\}$. 1 es el nodo terminal. Su grado de salida es 0. Φ es el conjunto de funciones de nodos. El grado de salida de $\phi \in \Phi$ es 1 y su grado de entrada es 0. Las funciones de los nodos están uno a uno en correspondencia con los componentes de F. Los bordes salientes de las funciones de los nodos pueden tener atributos de complemento. Los bordes salientes de los nodos son etiquetados con T y E respectivamente. El borde E puede tener el atributo de complemento. Nosotros usamos ($l_{(v)}$, T, E) para indicar un nodo interno y sus dos bordes de salida. Las variables en S_F están ordenadas y si v_j es descendiente de v_i ($v_i, v_j \in V$), entonces $l(v_i) < l(v_j)$.

La función F representada por un BDD se define como sigue:

- La función del nodo terminal es la función constante 1.

- La función de un borde es la función del nodo principal, a menos que el borde tenga el atributo de complemento, en cuyo caso la función del borde es el complemento de la función del nodo.
- La función de un nodo v ∈ V está dada por $l_{(v)}.f_T + \bar{l}_{(v)}.f_E$, donde f_T (f_E) es la función del borde $T_{(E)}$.
- La función de ϕ ∈ Φ es la función de su borde saliente.

Los gráficos BDD son canónicos (la representación de F es única para un orden de variables dado) si:

- Todos los nodos son descendientes de algún nodo en Φ.
- No existen subgrafos isomorfos.
- Para cualquier nodo $f_T \neq f_E$.

Nosotros consideramos BDD que cumplan con estos requisitos. La restricción que el borde T no puede ser complementado se impone para garantizar canonicidad.

Los BDD pueden construirse a partir del teorema de expansión de Boole. Veremos que este teorema juega un papel central en la definición y manipulación de los BDD. Como ejemplo veremos cómo se construye un BDD para la función:

$$f = a.b.c + \bar{b}.d + \bar{c}.d$$

donde el orden de las variables es:

$$b \leq c \leq d \leq a$$

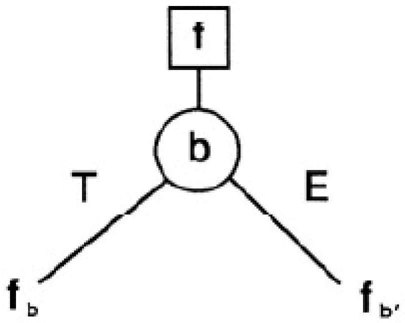
Comenzamos calculando el cofactor de f respecto de b, que es la primera variable que aparece en el ordenamiento. Obtenemos:

$$fb = a.c + \bar{c}.d$$

y:

$$f\bar{b} = d + \bar{c}.d$$

Podemos resumir este resultado inicial mediante un diagrama parcial como el de la figura:



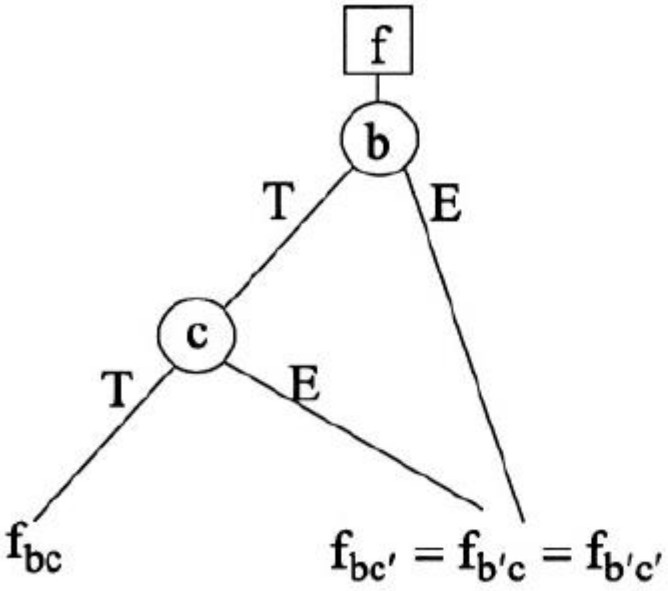
En general es cierto que los dos hijos de un nodo representan los dos cofactores de la función representada por el nodo con respecto a la variable etiquetada en dicho nodo.

Ahora calculamos los cofactores de f_b y $f_{b'}$ respecto de c. Obtenemos:

$$(f_b)_c = f_{bc} = a \qquad f_{b'c} = d$$

$$F_{bc'} = d \qquad f_{b'c'} = d$$

Obsérvese que tres de estos cofactores son idénticos. Por eso creamos un nodo simple para ellos en el nuevo BDD parcial, como se muestra en la siguiente figura:



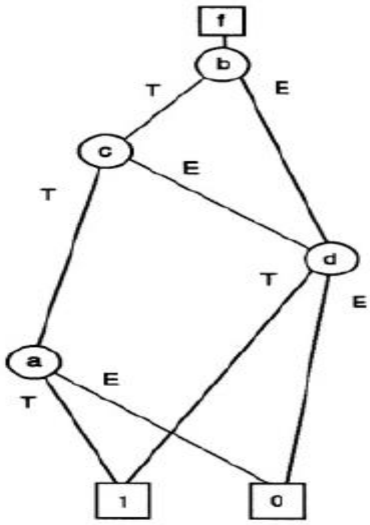
Al tener cofactores idénticos garantiza que el BDD será reducido (no contiene nodos duplicados ni superfluos). Esta es una propiedad importante que ya veremos. Finalmente observando que:

$$(x_i)_{x_i} = 1 \text{ y } (x_i)_{x'_i} = 0$$

Y que:

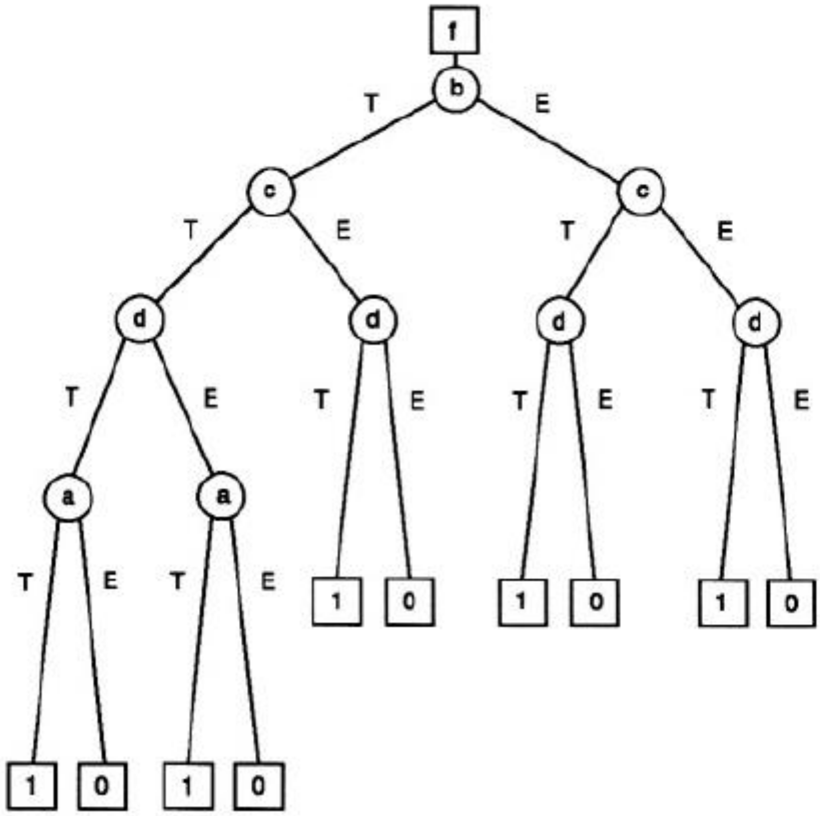
$$(x_i)_{x_j} = (x_i)_{x'_j} = x_i \quad \text{para } i \neq j$$

Obtenemos el BDD final:

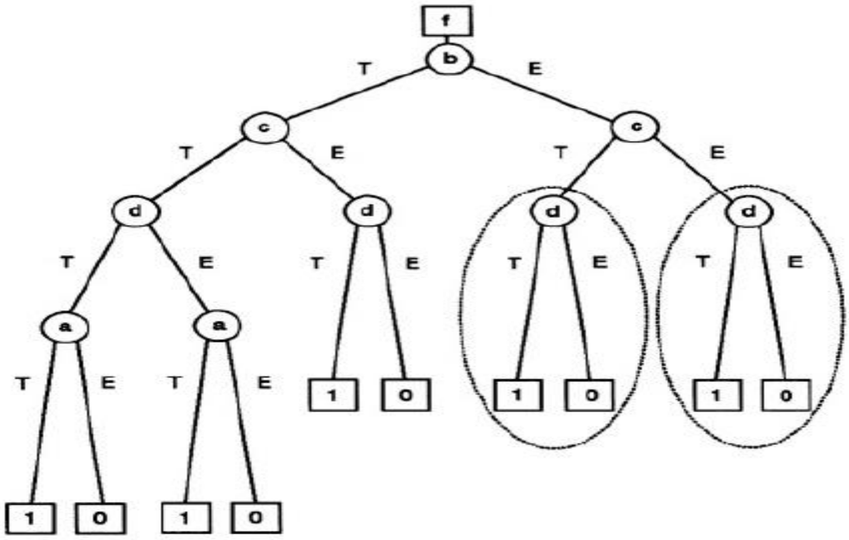


<u>Reducción de BDD:</u>

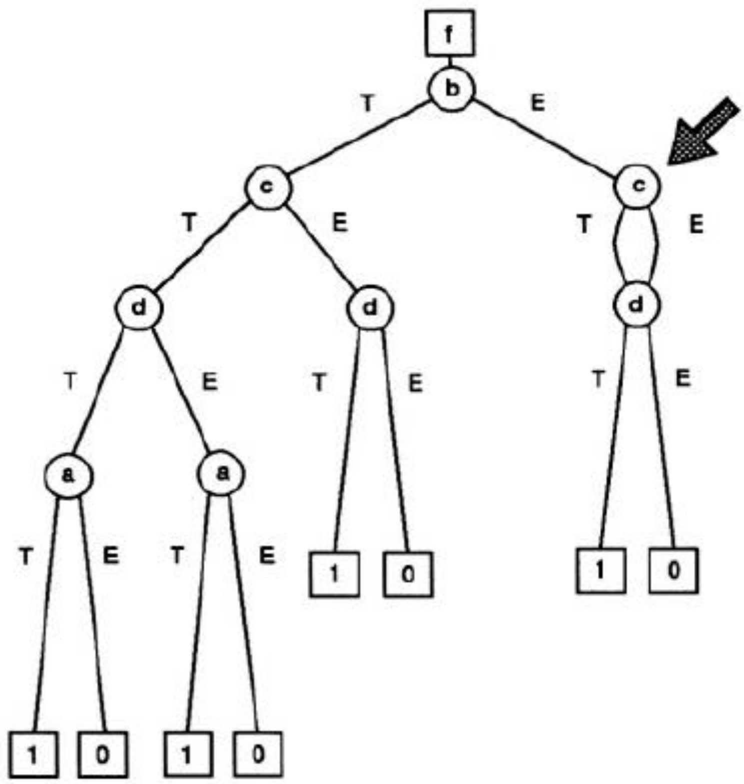
Tengamos en cuenta que, si hubiéramos identificado los cofactores idénticos, habríamos obtenido el árbol de la siguiente figura:



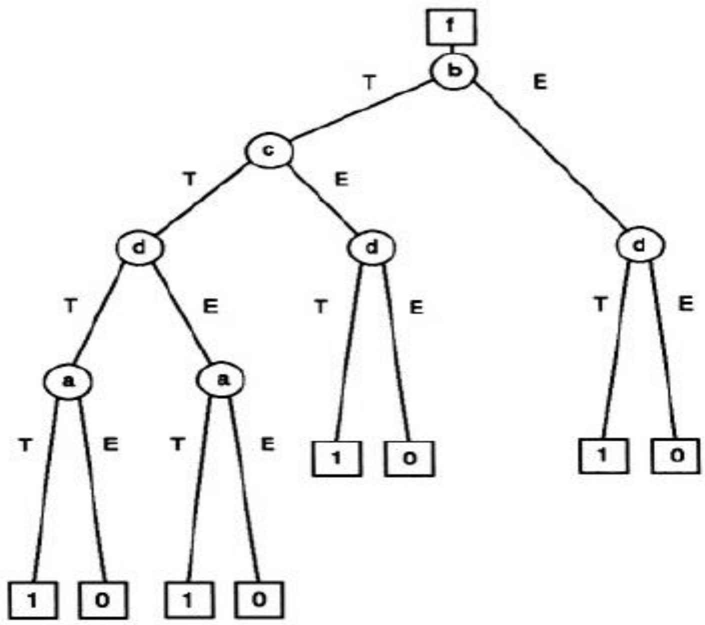
Este BDD, a diferencia de los anteriores, es no reducido. Un BDD no reducido se puede transformar sistemáticamente en uno reducido. Consideremos los dos subgrafos señalados en la siguiente figura:



Ellos representan la misma función y, por lo tanto, se pueden combinar como se ve en la siguiente figura:



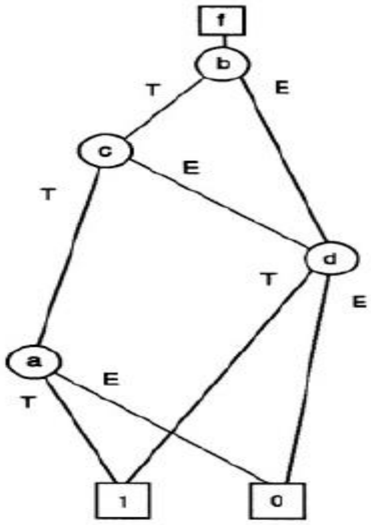
Ahora veamos el nodo apuntado por la flecha. Este nodo es redundante, ya que no corresponde a ninguna decisión, por lo que puede ser eliminado.



Para reducir un grafo BDD debemos aplicar dos reglas en forma iterativa:

- Identificación de subgrafos isomorfos.
- Eliminación de nodos redundantes.

Aplicando estas dos reglas llegamos al grafico inicial reducido.



Dado un orden, el grafico reducido para una función es único. Dos funciones son equivalentes si y solo si tienen el mismo BDD.

Otras propiedades interesantes de los BDD son:

- El tamaño del BDD (número de nodos) es exponencial con respecto al número de variables independientes. Sin embargo, son de muy buen comportamiento para muchas funciones que no son susceptibles de representación en dos niveles (por ejemplo, OR exclusivas).
- El AND lógico y el OR lógico tienen la misma complejidad (polinomio en el tamaño de los operandos) pero la complementación es barata.
- Los problemas de satisfactibilidad y las tautologías pueden ser resueltas en el mismo tiempo. De hecho, una función es una tautología si y solo si su BDD consiste en un nodo terminal "1".
- Los problemas de cobertura pueden resolverse en tiempo lineal de acuerdo al largo del BDD representando las restricciones.

Por otro lado:

- El tamaño del BDD depende del orden de las variables. A veces no es sencillo encontrar un ordenamiento óptimo.
- Existen funciones para las cuales las representaciones SOP o POS son más compactas que los BDD. Sin embargo, algunas restricciones de cobertura se manejan mejor en un BDD.

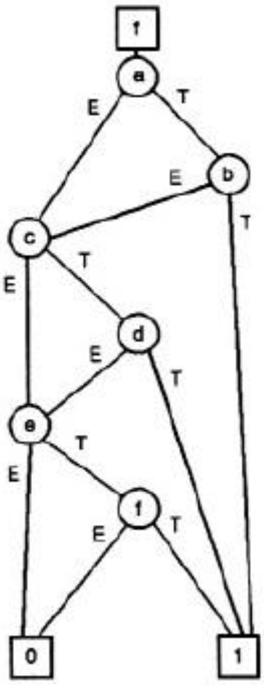
Veremos por qué el orden es importante, consideremos la siguiente función:

$$f = a.b + c.d + e.f$$

cuyo orden son:

$$a \leq b \leq c \leq d \leq e \leq f$$

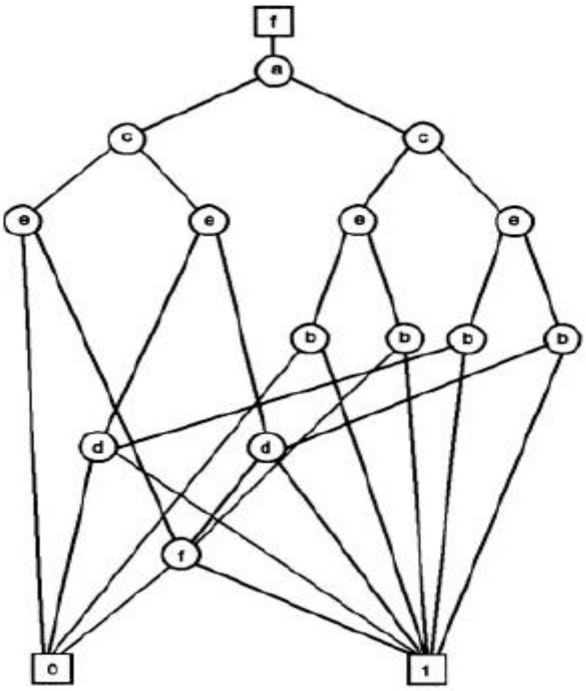
Su grafico BDD es el siguiente:



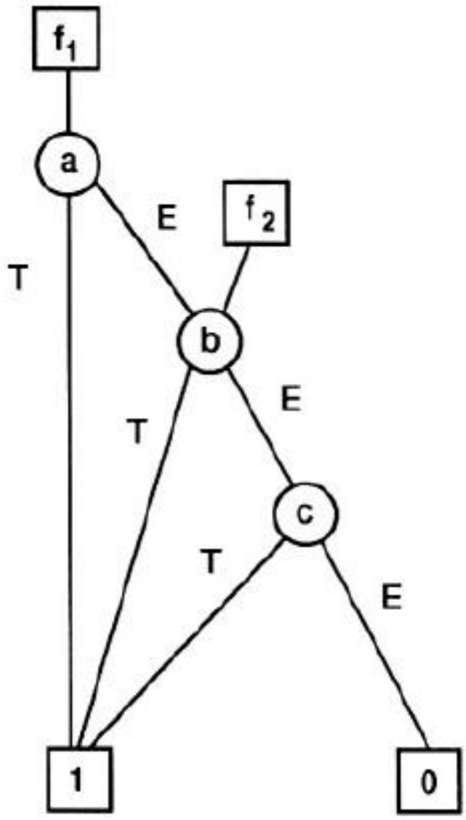
Ahora si el orden fuera:

$$a \leq c \leq e \leq b \leq d \leq f$$

nos queda:



Se ve que el ordenamiento es importante. Otra propiedad importante es compartir BDD para varias funciones. Si tenemos varias funciones, es probable que tengan subexpresiones en común. Por ejemplo, si consideramos las funciones $f_1 = b + c$ y $f_2 = a + b + c$. El grafo resultante podría ser:



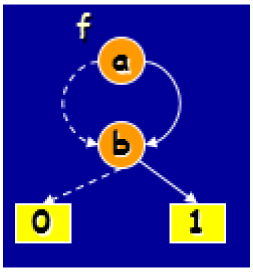
Ahora veremos un ejemplo sencillo. Queremos simplificar la función:

$$f = a.b + \bar{a}.b.c + \bar{a}.b.\bar{c}$$

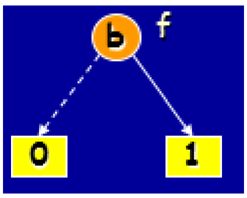
Calculando los cofactores, nos quedan:

$$fa = b \quad y \quad f\bar{a} = b$$

Llegamos al primer BDD:



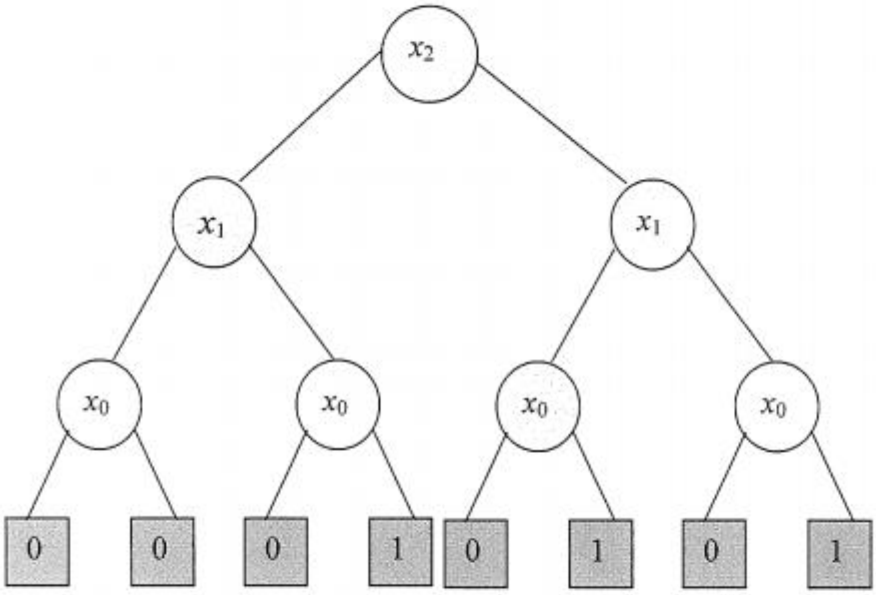
Vemos que el nodo "a" es redundante. Por lo tanto, el gráfico final es el siguiente:



<u>Ejemplo:</u> Simplificar usando diagramas BDD la siguiente función booleana:

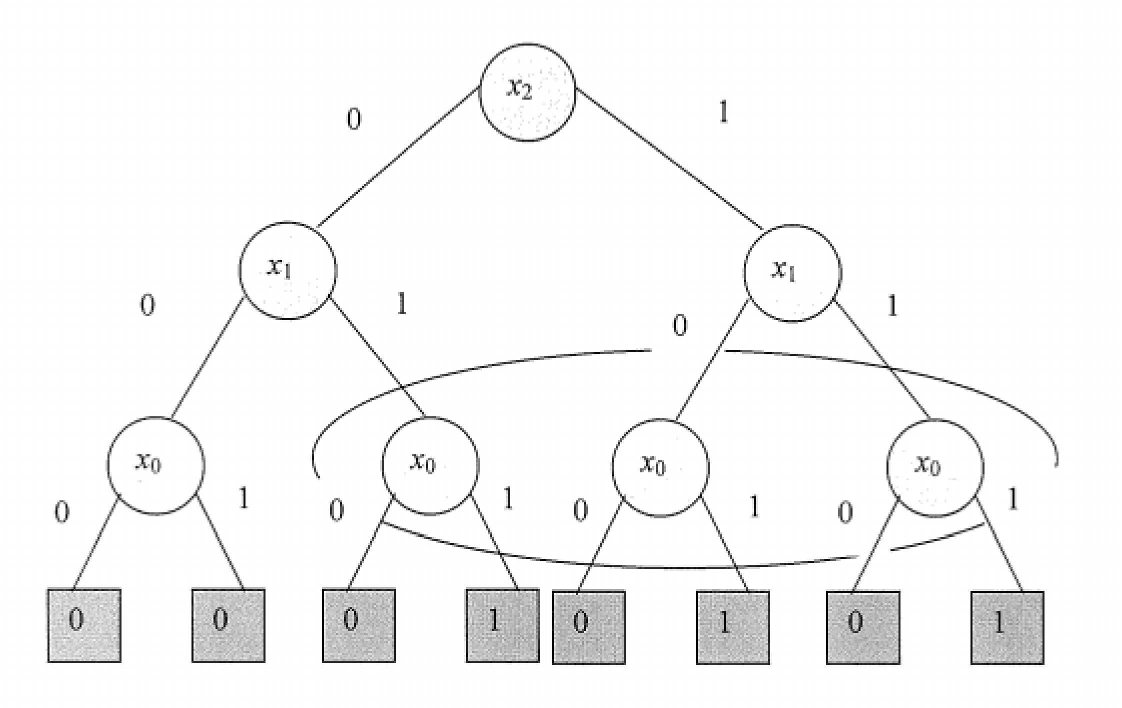
$$F(x_2, x_1, x_0) = \sum m(3, 5, 7)$$

El diagrama BDD correspondiente a esta función se ve en la siguiente figura. Como la función está expresada como suma de minitérminos, el diagrama es canónico.

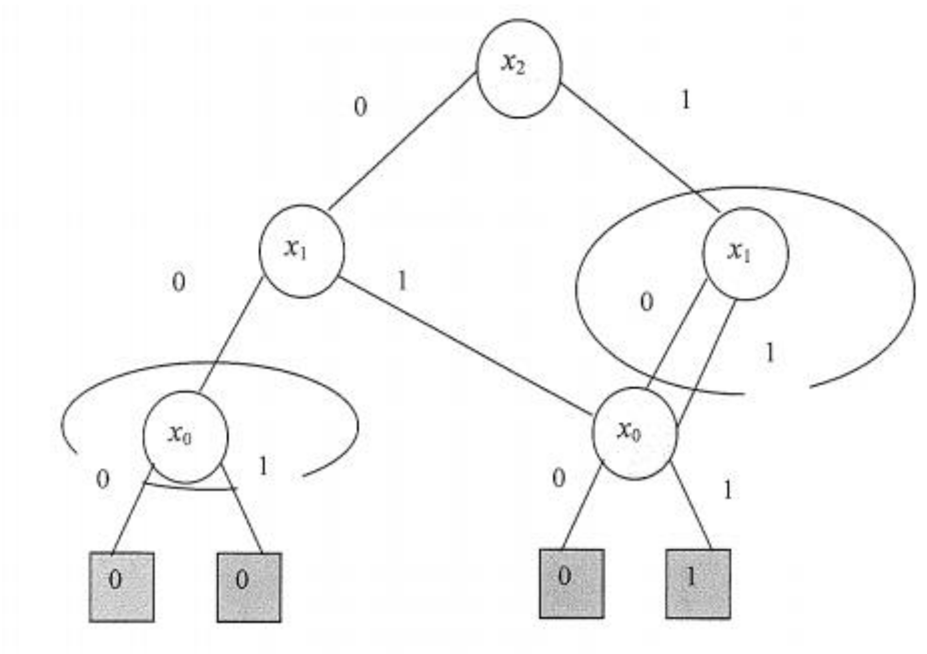


$$f(x_2, x_1, x_0) = (x_2 x_1 x_0 + x_2 \bar{x}_1 x_0 + \bar{x}_2 x_1 x_0)$$

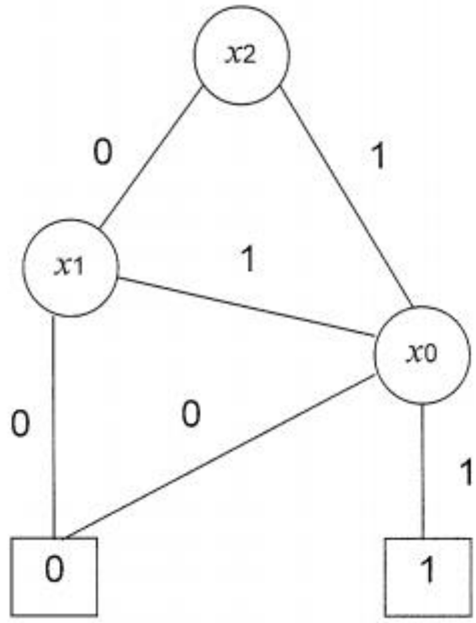
Como se ve en el diagrama los tres nodos internos representados por **x_0**, están duplicados. No sucede lo mismo con el de la izquierda pues los nodos a los que llega no son iguales a los otros tres. En las dos figuras siguientes vemos como se reducen los nodos internos de un diagrama BDD que representan la misma función:



Al eliminar estos nodos repetidos nos queda:



Nos queda ahora ver los nodos redundantes, cosa que pasa con **x_0** de la izquierda y con **x_1**. Los mismos se ven en la figura anterior marcados con círculos. Por lo tanto, la función simplificada queda representada por el siguiente diagrama BDD.



$f(x_2, x_1, x_0) = (x_1x_0 + x_2x_0)$

Otro ejemplo: Queremos simplificar la siguiente función.

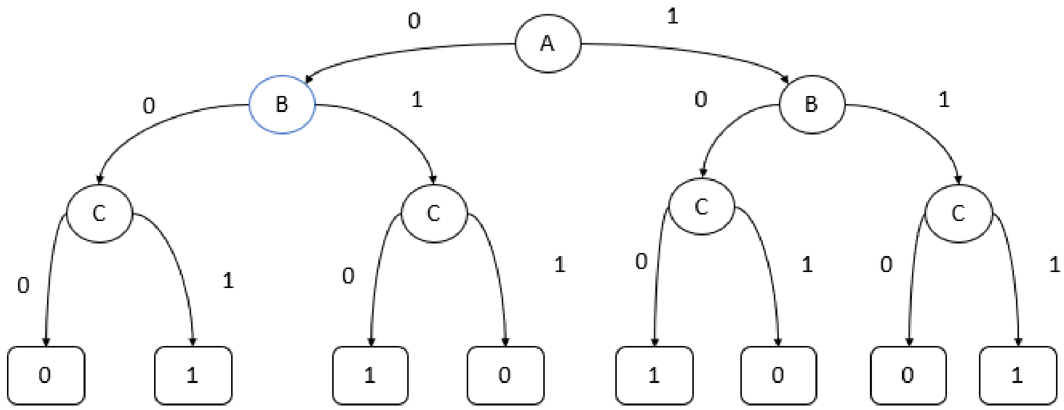
F (A, B, C) = ∑ m (1, 2, 4, 7)

Esta función se puede ver representada en el siguiente mapa de Karnaugh:

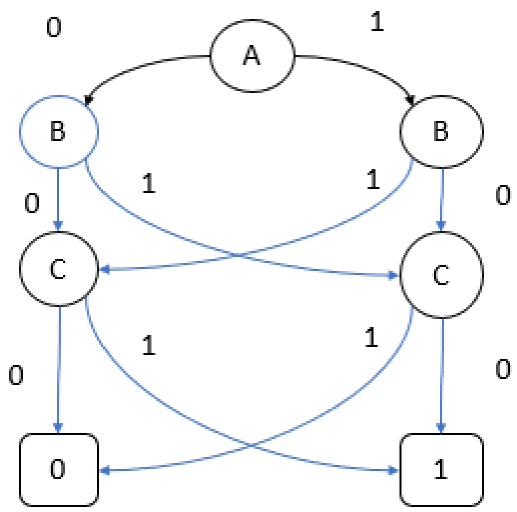
C \ AB	**00**	**01**	**10**	**11**
0		1		1
1	1		1	

Como se ve claramente esta función da una EXOR entre las tres variables, pero lo trataremos como si no conociéramos el resultado y lo Resolvemos aplicando diagramas BDD.

El diagrama correspondiente canónico a la función expresada como sumatoria de miniterminos es el siguiente:



Buscando nodos repetidos vemos que no todos los nodos etiquetados con la variable C son repetidos. Los de los extremos son iguales entre sí y difieren de los dos del medio. Resolviendo este problema nos queda el siguiente diagrama BDD:



Como se ve este diagrama es la función EXOR entre las tres variables:

$$F = A \oplus B \oplus C$$