APLICACIÓN DE FRACTALES A LA SOLUCIÓN DE FUNCIONES BOOLEANAS

FRACTALS APPLICATION TO THE SOLUTION OF SHOWS BOOLEANS

Sergio Adrián Martín
Investigador y Catedrático
Universidad Francisco Gavidia

ISSN 1992-6510

22

REALIDAD Y REFLEXIÓN
Reality and Reflection

Año 7, Nº 22 | San Salvador, El Salvador, Centroamérica | Revista Cuatrimestral | enero-abril 2008
Year 7, Nº 22 | San Salvador, El Salvador, Central America | Quarterly Journal | january-april 2008

Aplicación de fractales a la solución de funciones booleanas

Fractals application to the solution of shows Booleans

Sergio Adrián Martín
Investigador y Catedrático
Universidad Francisco Gavidia

El empleo de fractales en la investigación de un procedimiento perfeccionado con el fin de solucionar funciones booleanas comprueba ser viable. El próximo paso lógico consistiría en llevar a cabo un método que zanje problemas de lógica secuencial, no solamente combinacional, en el diseño de sistemas digitales, aprovechando lo que ya se logró aplicando fractales a los problemas de lógica combinacional. Los fractales están demostrando poseer aplicaciones impensadas y enormemente prometedoras, y que si bien no forzosamente ha de tornar obsoletas otras áreas de las matemáticas, consigue aportar métodos eficientes y prontos para resolver muchos problemas. FUNCIONES BOLEANAS, FRACTALES.

Fractals job in the investigation of a perfected procedure with the aim of solving Booleans functions shows that is viable. The next logical step would involve accomplishing a method that break problems through of sequential logic, not only combinational, in the design digital systems, making good use of what right now you got applying fractals to the problems of logic combinational. The fractals are proving to possess unexpected and enormously promising applications, and what even though there is of turning obsolete other areas of mathematics, you manage to contribute efficient and ready methods to solve many problems. SHOWS BOOLEANAS, FRACTALES.

Teorema 6.3: Los LLC son cerrados bajo homomorfismo inverso.

Demostración:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$$

$$M' = (Q', \Sigma, \Gamma, \delta', [q, \epsilon], F \times \{\epsilon\})$$

donde Q' contiene parejas $[q, x]$ y

1. $\delta'([q, x], \epsilon, Y)$ contiene todos los elementos $([p, x], \gamma)$ tales que $\delta(q, \epsilon, Y)$ contiene a (p, γ).
2. $\delta'([q, ax], \epsilon, Y)$ contiene todos los elementos $([p, x], \gamma)$ tales que $\delta(q, a, Y)$ contiene a (p, γ).
3. $\delta'([q, \epsilon], a, Y)$ contiene todos los elementos $([q, h(a)], Y)$ para toda $a \in \Sigma$ y $Y \in \Gamma$.

un polinomio de Reed Muller (PPRM), no me pasó por la mente que pudiese existir algún tipo de fractal que simplificase el método que lleva a una solución canónica de minterms.

LOS FRACTALES OCULTOS

El primer paso a considerar es: ya que se necesita contar con todas las tablas de verdad de todos los posibles minterms que se generan con n variables de entrada, ¿existirá algún tipo de fractal que pueda generarlas todas?

Después de todo, en la matriz que se usa para resolver los casos de PPRM, a cada columna corresponde la tabla de verdad de un minterm, con la diferencia de esta matriz cuenta con solo 2^n columnas, mientras que el total de minterms a evaluar en la forma canónica es de 3^n.

En un principio fui optimista pensando que al igual que en la matriz usada para PPRM, se podría llegar a un método tan simple que implicara solo la interacción de los elementos de una fila. Sin embargo, en cierto momento me percaté que el comportamiento de todas las tablas de verdad, formando columnas, seguía un comportamiento más parecido al de una expansión de Kronecker.

Las tablas de verdad debían agruparse guiándose por el equivalente ternario de cada función (por ejemplo b'a' equivaldría a 00, b'a a 01, y b' a 02, donde 2 equivale a la ausencia de una variable).

Si se sigue este sistema, el primer mintermin en un sistema de dos variables correspondería a 00, y el último a 22. La matriz que en la que se basará en fractal es la siguiente:

$$\begin{bmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \end{bmatrix} \quad (1)$$

Si se lleva a cabo una expansión fractal de la matriz anterior, en un primer paso se generara una matriz de nueve columnas y cuatro filas, así:

$$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \quad (2)$$

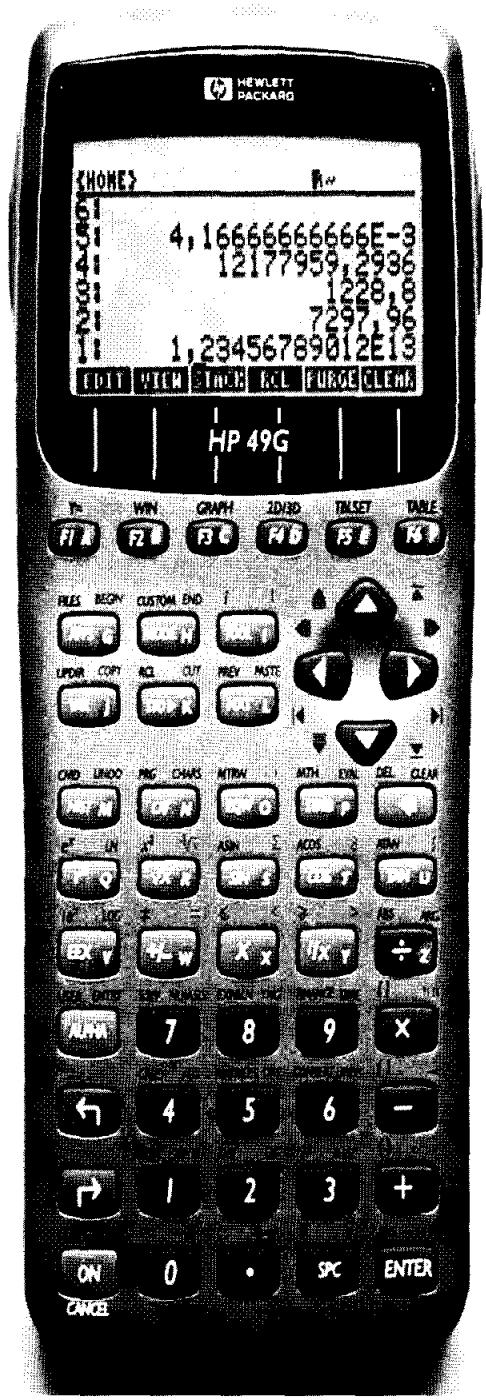
$$\begin{bmatrix} 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 \end{bmatrix} \quad (3)$$

Al analizar esta nueva matriz, la primera columna corresponde a la tabla de verdad del minterm b'a', la segunda a b'a, la tercera a b', la cuarta a ba', la quinta a ba, la sexta a b, la séptima a a', la octava a a, y la última a 1.

Se cubren los nueve minterms posibles que se pueden generar con dos variables, y el orden en que se generan corresponden a los equivalentes ternarios que van desde 00 hasta 22.

Si este proceso se repite para un sistema de tres variables, simplemente se expande el fractal anterior para originar una matriz de 27 columnas y ocho filas. El resultado contendrá las tablas de verdad de todos los minterms de un sistema de tres variables.

Se puede concluir de aquí que mediante la expansión fractal de la matriz (1) se pueden generar todas las tablas de verdad de cualquier sistema de n variables. Esto abrevia notablemente el proceso de generar las tablas de verdad de los minterms a usar.



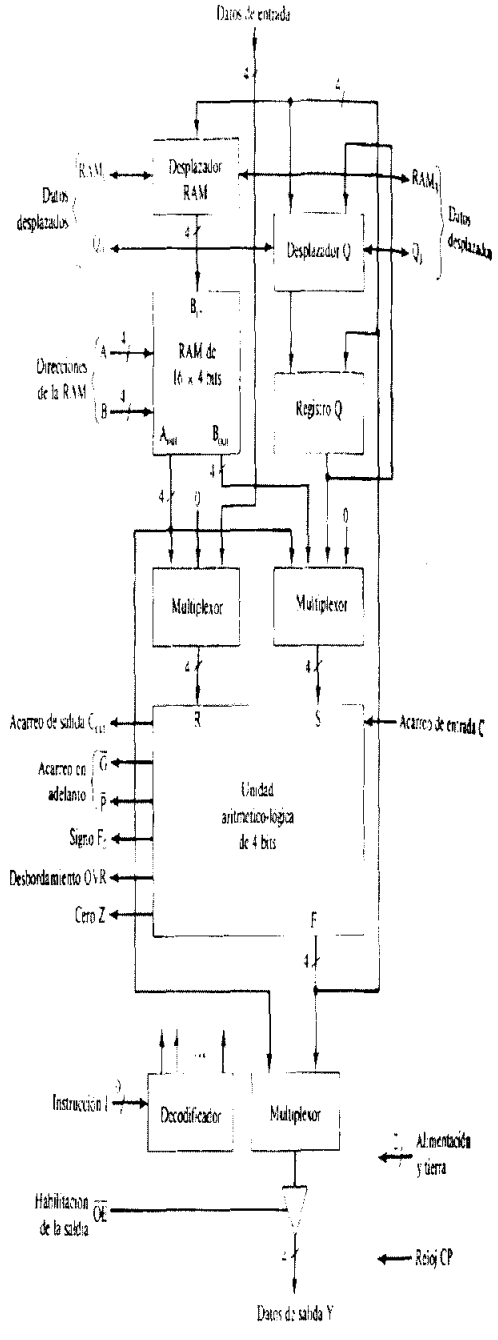


Figura 8.26 Organización interna del módulo microprocesador de 4 bits AMD 2901.

LA PARADOJA DE LA CANTIDAD DE INFORMACIÓN

En un primer momento, el simple desarrollo de cada una de las tablas de verdad podría ser visto como un logro en sí mismo, pero al observar la matriz resultante para un sistema con n variables, se puede observar una peculiaridad: la cantidad de unos en la matriz primigenia es de cuatro, de un total de seis casillas, y en la segunda matriz es de 16 de un total de 36 casillas. Significa que el porcentaje de unos en relación al total de bits de la matriz se rige por la siguiente fórmula:

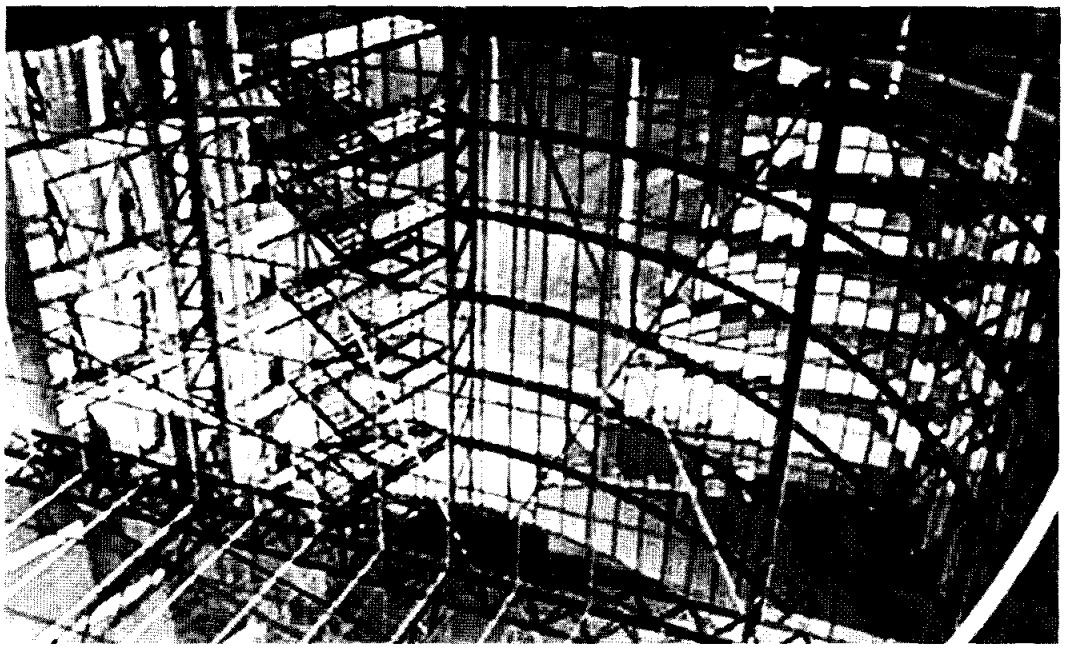
$$P = \left(\frac{4}{6}\right)^{n} x100\%$$

Donde n es el número de variables independientes del sistema. Ya que la relación 4/6 = 0.666 es menor a la unidad, P tiende a 0 para valores grandes de n. La paradoja en relación a esto, es que podría suponerse que si al incrementarse el valor de n se esta en posición de crear funciones lógicas más complejas, la cantidad de unos en la matriz que refleja el comportamiento de los minterms debería incrementarse, o en todo caso mantenerse constante, sin embargo la fórmula indica lo opuesto.

En realidad es como si diera a entender que con una cantidad de unos pequeña al tamaño de la matriz se pueden llegar a conseguir todas las funciones lógicas posibles, así que la cantidad de unos no es representativa de la cantidad de funciones lógicas que se pueden generar.

LA MATRIZ DE INCLUSIÓN

Para determinar qué minterms contiene a otros sin necesidad de iterar entre las tablas de verdad de los minterms, siempre maneje la idea de un archivo o una tabla



que determinara de antemano qué minterms estaban incluidos dentro de otros. Sin embargo esto nunca lo llevé a la práctica.

La principal razón de no hacerlo es porque de todas formas el método de la iteración de tablas no implicaba un consumo extremo de tiempo, a pesar de que de contar con datos previamente almacenados se ahorraría tiempo máquina.

Otro factor limitante es que si cada registro equivaldría al ancho de una tabla de verdad, habría que tener un especial cuidado al respecto debido al crecimiento exponencial del número de bits de cada tabla de verdad, con el incremento del número de variables.

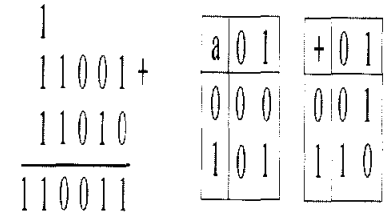
Esto podría representar una desventaja, especialmente si se pretendía manejar una aplicación que operara en el Web.

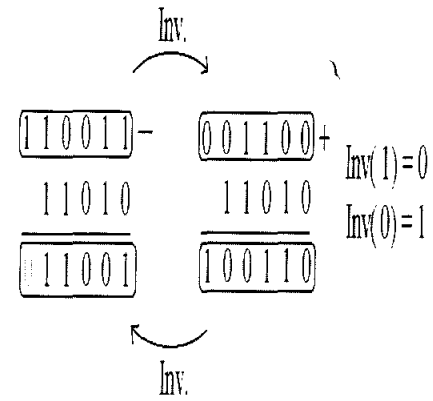
Sin embargo, una vez llegue a la conclusión de que un arreglo fractal era capaz de originar todas las tablas de verdad necesarias para un sistema de n variables, cabía la posibilidad de que otro fractal pudiese generar una matriz que indicara qué minterms estaban incluidos en otros.

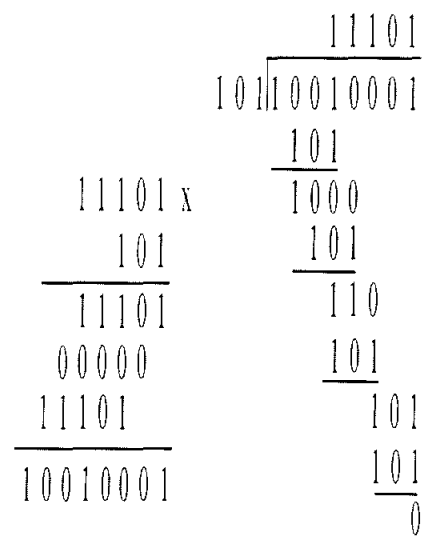
Así pues en un sistema de dos variables, existirán nueve minterms a partir de los cuales se pueden generar todas las funciones posibles. Si se construyese una matriz que indicara con un 1 si un minterm está incluido dentro de otro, los incluidos se ordenarían en las columnas, y los incluyentes en las filas.

La única forma en que un minterm esté incluido en otro y a la vez sea incluyente es cuando se compara consigo mismo. Esto determina que todos los elementos de la diagonal principal de la matriz siempre han de ser 1.

Pero si se ordenan los minterms siguiendo un orden ternario, como el que se explicó anteriormente, los minterms que incluyen menos variables serán los que ocuparan las columnas más a la derecha, y éstos no







podrán estar incluidos dentro de los minterms que usan más variables.

De allí que todos los términos a la derecha y arriba de la diagonal principal han de ser cero. Para el caso de un sistema de nueve minterms la matriz de inclusión se presenta de la siguiente manera:

	b'a'	b'a	b'	ba'	ba	b	a'	a	1
b'a'	1	0	0	0	0	0	0	0	0
b'a	0	1	0	0	0	0	0	0	0
b'	1	1	1	0	0	0	0	0	0
ba'	0	0	0	1	0	0	0	0	0
ba	0	0	0	0	1	0	0	0	0
b	0	0	0	1	1	1	0	0	0
a'	1	0	0	1	0	0	1	0	0
a	0	1	0	0	1	0	0	1	0
1	1	1	1	1	1	1	1	1	1

Tabla 1

Al observar detenidamente esta tabla se puede observar un fractal básico que se expande de acuerdo a su distribución de 1 y 0.

1	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0
0	0	0	1	0	0	0	0	0
0	0	0	0	1	0	0	0	0
0	0	0	1	1	1	0	0	0
1	0	0	1	0	0	1	0	0
0	1	0	0	1	0	0	1	0
1	1	1	1	1	1	1	1	1

Aunque no es idéntico al fractal que genera las tablas de verdad, mediante este se puede generar la matriz de inclusión, y usarla para descartar los minterms que ya

están incluidos dentro de otros, teniendo la precaución de nunca permitir que un minterm se descarte a sí mismo.

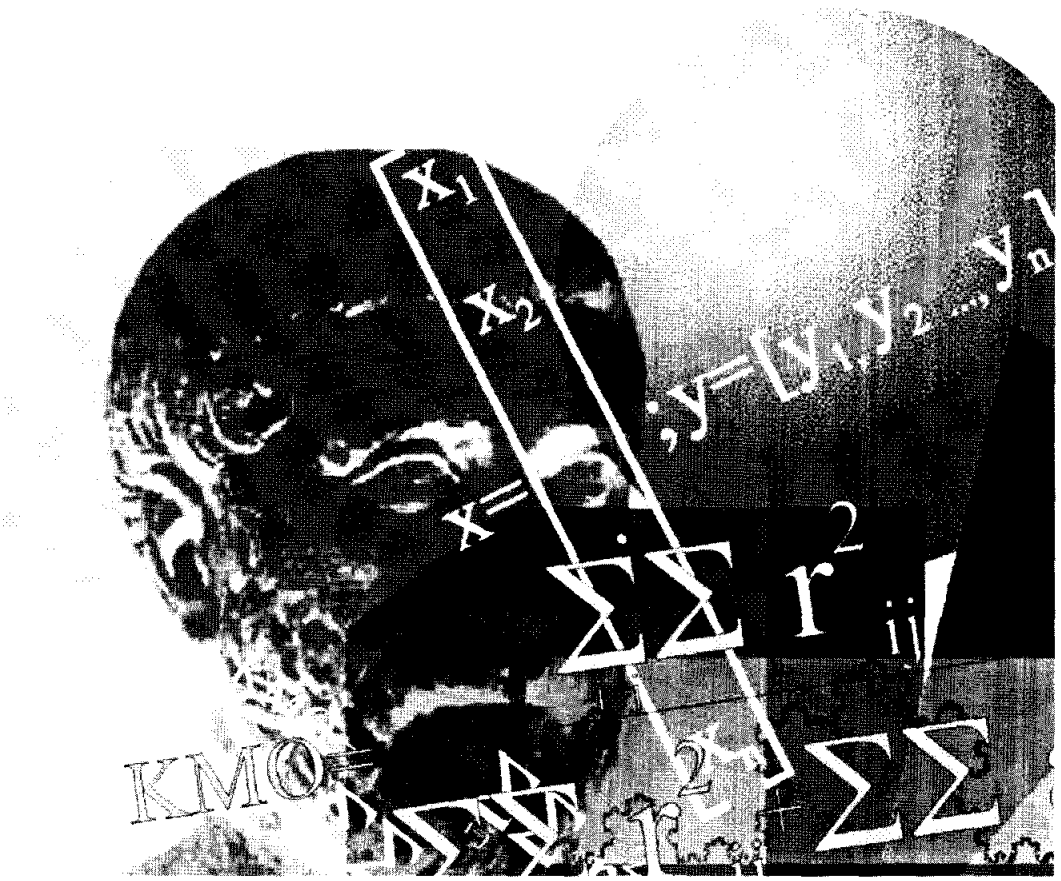
El único inconveniente real del uso de esta matriz está en el consumo de memoria que implica, pues para un sistema de ocho variables, la matriz ha de tener 6,561 filas y 6,561 columnas, lo cual consumirá al menos 43MB de RAM.

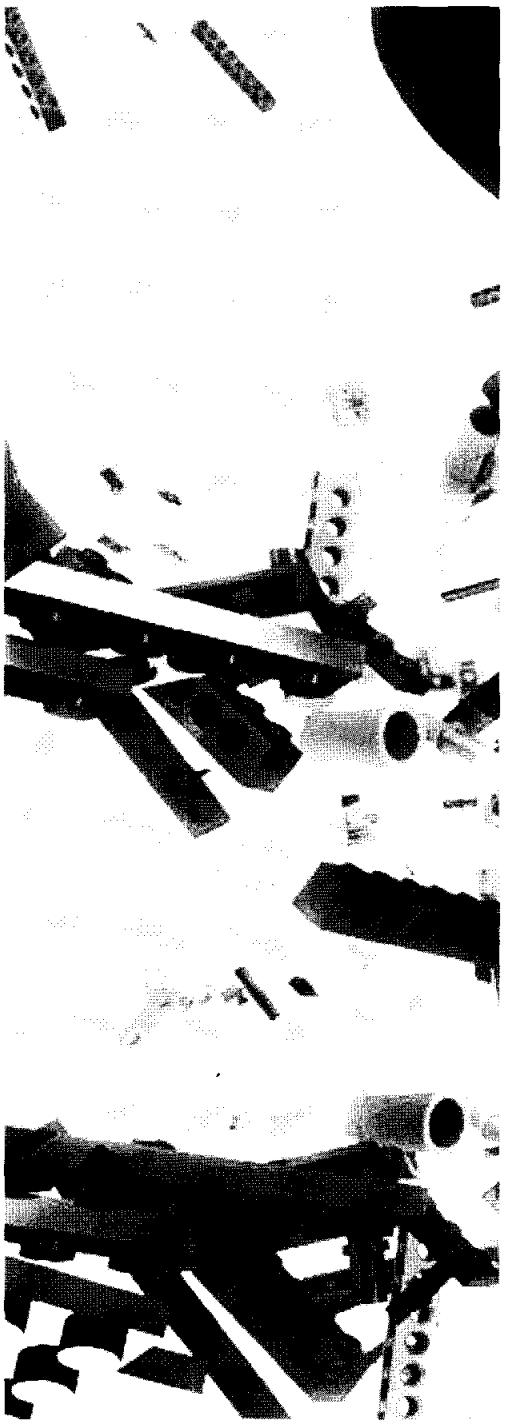
Dependiendo de limitantes impuestas por el lenguaje en que se programa la aplicación y de la memoria disponible en la computadora esto puede representar un límite al número de variables que se pueden ocupar como máximo siguiendo este método.

IMPLICACIONES PARA SIMPLIFICACIONES CON BASE EN MANTERMS

Ya que el uso de fractales para deducir qué minterms son partes de una solución es factible, vale la pena preguntarse si lo mismo no es aplicable a una solución basada en manteras.

En el pasado planteé y comprobé que se puede seguir un método que lleva a la solución en base a manteras, siempre y cuando se conozcan todas las tablas de verdad





de todos los manteras que puedan generarse con *n* variables.

De aquí que el problema no sea en sí el llegar a una solución contando con las tablas de verdad, sino más bien si estas pueden ser generadas mediante un fractal. Tomando en cuenta el teorema de D'Morgan:

$$(a'+b') = (ab)'$$

Todo minterm negado daría lugar a un manterm. De allí se puede deducir que la tabla de verdad negada de un manterm corresponderá a la tabla de verdad de un minterm.

Ya que un fractal es capaz de originar todas las tablas de verdad de los minterms, esa matriz contendrá las tablas de verdad negadas de todos los manterms, la única diferencia será la forma en que están ordenadas las columnas respecto a como aparecerían respecto a una matriz que contenga todas las tablas de verdad de los minterms.

Para un sistema de dos variables, las tablas de verdad de los manterms formarían el siguiente arreglo:

a'+b'	b'+a	b'	b+a'	b+a	b	a'	a	0
1	1	1	1	0	0	1	0	0
1	1	1	0	1	0	0	1	0
1	0	0	1	1	1	1	0	0
0	1	0	1	1	1	0	1	0

(Tabla 2)

Si se invierten los valores de este arreglo, se llega a la siguiente matriz, en la que es fácil reconocer un patrón fractal:

0	0	0	0	1	1	0	1	1
0	0	0	1	0	1	1	0	1
0	1	1	0	0	0	0	1	1
1	0	1	0	0	0	1	0	1

(Tabla 3)

Es interesante el hecho de que este patrón fractal es similar al de la matriz de minterms, excepto porque éste correspondería a la imagen del espejo del otro (si se coloca la primera fila en el lugar de la última, y viceversa, y la segunda se intercambia con la tercera se obtiene la matriz de minterms para dos variables).

En este punto resulta evidente que la idea de usar fractales para obtener las tablas de verdad de los manteras es factible, aunque sólo un poco más complicada que el procedimiento en que se usan minterms.

CONCLUSIONES:

El uso de fractales en la búsqueda de un mejor método para resolver funciones booleanas demuestra ser factible, al punto de poder desarrollar programas más compactos y rápidos con ese fin (la última versión que desarrollé basada en fractales es un programa en Visual Basic que ocupa tan solo 40K de espacio en disco y permite resolver sistemas de siete variables).

Esto deja la puerta abierta al que sería el siguiente paso lógico: desarrollar un sistema que permita resolver problemas de lógica secuencial, no solamente combinacional, en el diseño de sistemas digitales, aprovechando lo que ya se logró aplicando fractales a los problemas de lógica combinacional.

Tal vez la mejor conclusión a la que se puede llegar, es que los fractales están demostrando tener aplicaciones insospechadas y muy prometedoras, para ser una disciplina con menos de 40 años de existencia, y que si bien no necesariamente ha de volver obsoletas otras áreas de las matemáticas, puede aportar métodos eficientes y rápidos para resolver muchos problemas.