Evolution of the Boolean Function Manipulation

Sim Poh Ching, Chin Kui Fern, and Mohamad Kadim Suaidi

Faculty of Engineering
Universiti Malaysia Sarawak
94300 Kota Samarahan, Sarawak, Malaysia

Abstract - Solving large systems of Boolean equations is a hard combinatorial problem. It can be greatly facilitated by preliminary reducing the number of roots in separate equations, which in turn, leads to decreasing the number of variables in a considered system, the number of equations and time complexity. This can have a significant effect upon the physical space and connectivity of electronic logic circuits when these are implemented using actual electronic devices or components. In this paper, we review various representation techniques, Boolean function manipulation issues and problems that are used in practical design environment for digital systems design.

1. INTRODUCTION

Boolean functions are used as models for changes in discrete, signal edges, errors, binary coded systems such as cryptosystems for information authentication as well as data encryption. An important Boolean problem is that of designing Boolean functions satisfying simultaneously more than one crucial criteria just as dependence and independence, linearity, monotonicity, high nonlinearity, high degree of propagation, 0-1 balancedness and high algebraic degree.

Boolean function manipulation is the art of exploiting simplification opportunities that exist inherently in logical structures by using the identities that exist within that algebra or reducing the number of terms used. We are seeking to express any logical function in as simplified a manner as possible because this leads inevitably to a reduced usage of resources when that same function is implemented physically.

2. CLASSICAL REPRESENTATION TECHNIQUES

2.1 Truth Tables

Row	n inputs						1 output
	x_n	x_{n-1}	x_{n-2}	...	x_{n2}	x_{n1}	$f(x_n, x_{n-1}, \ldots, x_1)$
0	0	0	0	...	0	0	d_0
1	0	0	0	...	0	1	d_1
2	0	0	0	...	1	0	d_2
⋮	⋮	⋮	⋮		⋮	⋮	⋮
i							d_i
⋮	⋮	⋮	⋮		⋮	⋮	⋮
2^n-1	1	1	1	...	1	1	d_{2^n-1}

Fig. 1. General structure of a truth table.

A truth table (table of combinations) specifies the values of a Boolean expression for every possible combination of values of the variables in the expression. Fig. 1 shows the general truth table for a function with n variables. The truth table of a function of n input variables has $N = 2^n$ rows. The exponential increase in storage requirement limits the truth table method of representing Boolean function if being used for large design.

2.2 Algebraic Expressions

The algebraic equivalent of the truth table is the canonical form of expression. By summing the minterms corresponding to the row of the truth table for which the particular function is true, a unique sum-of-product (SOP) algebraic expression known as the disjunctive canonical form (DCF) is obtained. In general,

$$f(x_n, x_{n-1}, \ldots, x_1) = \sum_{i=0}^{2^n-1} d_i\, m_i$$

where the summation is the logical OR operation.

Algebraic method of representing switching circuits specifies a design completely without the storage problem. Simplification algorithms of this method are easily implemented onto a computer. Hence, it has been the most popular way of representing switching circuits. However, when an algebraic representation is subjected to manipulation, the intermediate storage and computation time requirements may present problems.

2.3 Karnaugh Maps

Karnaugh map provides a graphical display of the output variables involving in any SOP canonical or standard form expression. This is a cellular structure, which contains 2^n cells for n variables, which can simultaneously display the operational behaviour and the functional relationship of a switching circuit. Each cell corresponds to one row of the truth table and one minterm of the DCF. Labeling the edges of the map identifies the cells and this is done in such a way as to give a unique combination of variables and their inverses for each cell. Each variable divides the map into two equal halves each containing 2^{n-1} cells.

The size of Karnaugh map grows exponentially as the number of input variables feeding in increases. The map structure and the manipulation algorithms have the difficulty to be implemented onto a computer. Therefore, this technique is not suitable for computing. Nevertheless, when the technique is compared with the previous two approaches, it is still the most attractive technique for representing and manipulating Boolean function.

0-7803-7565-3/02/$17.00 ©2002 IEEE.