THE COMPLETE CLASSIFICATION OF BOOLEAN EXPRESSIONS

> The great advances in mathematics since antiquity, for instance in algebra, have been dependent to a large extent upon success in finding a usable and efficient symbolism. The purpose of symbolic language in mathematical logic is to achieve in logic what has been achieved in mathematics, namely, an exact scientific treatment of the subject matter.
> - D. Hilbert, 1928, Principles of Mathematical Logic

1 Introduction

I'd like to explain a cute trick I discovered to literally write truth tables as non-negative integers in base 2 in their natural order, and how to completely classify all possible boolean expressions. Lets begin with the following definition:

> **Definition:** A binary boolean operator is a function which takes as input two boolean variables and produces as output a single boolean variable.

You may already be familiar with the binary boolean operators AND and OR. For example, X AND Y return 1 if and only if both of the input boolean variables X, Y both have the value 1, and returns 0 otherwise; whereas X OR Y returns 1 if either or both of the two input variables is 1, and 0 otherwise.

Our goal in this paper is to obtain the well known result that there are exactly 16 binary boolean operators; and our approach can easily be generalized to the case of any number n of input boolean variables.

2 Boolean Expressions

If you are familiar with computer programming, then you already know what a "boolean expression" is. By definition, a single boolean variable can take the values of either 0 or 1 (in the language of logic, 0 corresponds to *False*, and 1 corresponds to *True*).

If A is a boolean variable, then an example of a boolean expression is

NOT A

This has the following truth table:

$$\textbf{NOT} = \left\{ \begin{array}{l} 0 \to 1 \\ 1 \to 0 \end{array} \right\}$$

Another example involving the boolean operators AND and NOT which acts on a single input boolean variable would be:

A AND NOT A

this boolean expression will always produce an output of 0 or *False* (can you see why?). This means the boolean expression "A AND NOT A" will always output 0. I call this the "zero function", because no matter what the input is, it will always output 0.

The reason for using truth tables is this approach completely sidesteps needing to deal with boolean expressions at all. This is very important for obtaining a complete classification, because there are infinitely many boolean expressions that all correspond to the exact same truth table (if this is unclear, then prove this as an exercise). In this case, we have seen for example that the boolean expression "0" and "A AND NOT A" are different boolean expressions that share the same truth table.

3 One Input Variable

Let's begin by considering the case of $n = 1$, for a single boolean input variable. In other words, lets consider the collection of all functions of the form

$$\{0, 1\} \to \{0, 1\}$$

In this case, there are two possible input combinations, namely: 0 and 1. Therefore we can obtain a complete classification of all possible boolean valued functions which act on these possible input combinations, by forming the so-called "truth tables".

There are only four possible such functions, they are the functions: **ZERO** (Z), **IDENTITY** (I), **NOT** (!), and **CONSTANT** (C)

$$Z = \left\{ \begin{array}{l} 0 \to 0 \\ 1 \to 0 \end{array} \right\} \quad I = \left\{ \begin{array}{l} 0 \to 0 \\ 1 \to 1 \end{array} \right\} \quad ! = \left\{ \begin{array}{l} 0 \to 1 \\ 1 \to 0 \end{array} \right\} \quad C = \left\{ \begin{array}{l} 0 \to 1 \\ 1 \to 1 \end{array} \right\}$$

We can also define these four functions as a sequence of symbols xy, since every boolean function f is defined by a collection of rules which corresponds to an ordered pair of symbols xy where $x, y \in \{0, 1\}$

$$xy = \left\{ \begin{array}{l} 0 \to x \\ 1 \to y \end{array} \right\} \qquad \Big\{ f : \{0,1\} \to \{0,1\} \Big\} = \left\{ \begin{array}{ll} 00 & 01 \\ 10 & 11 \end{array} \right\}$$

$$Z = 00 \qquad I = 01$$

$$! = 10 \qquad C = 11$$

Figure 1 shows how these four functions act on the underlying set $\{0, 1\}$

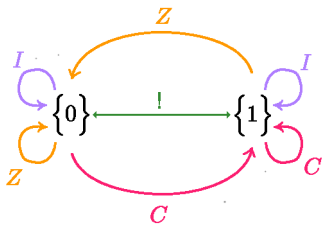

Figure 1

4 Two Input Variables

Let's now turn our attention to the case of 2 input boolean variables. The *Binary Boolean Operators* are functions of the form:

$$\{0, 1\} \times \{0, 1\} \to \{0, 1\}$$

and our goal in this section is to obtain the set of all possible such functions and show that there are exactly 16 of them.

In this case, there will be $2^2 = 4$ possible input combinations, these are 00, 01, 10, 11 (these happen to be the truth tables from the case of $n = 1$) and there will be $2^4 = 16$ total possible truth tables. Since each truth table can be written as a finite binary string of length 4, it's easy to see that there are exactly 16 such possible finite binary strings. This completes the proof that there are exactly 16 binary boolean operators.

If $\Diamond$ is a binary boolean operator that assigns an ordered pair of boolean variables $(x, y) \to z$, we write this as

$$z = x \Diamond y$$

We have already considered the two binary boolean operators **AND** (&) and **OR** (||)

$$\textbf{AND} = \left\{ \begin{array}{l} (1,1) \to 1 \\ (1,0) \to 0 \\ (0,1) \to 0 \\ (0,0) \to 0 \end{array} \right\} \qquad \textbf{OR} = \left\{ \begin{array}{l} (1,1) \to 1 \\ (1,0) \to 1 \\ (0,1) \to 1 \\ (0,0) \to 0 \end{array} \right\}$$

The next operator we shall consider is called **IMPLIES** ($\longrightarrow$). If the boolean expresion $x \longrightarrow y$ is *True*, this means "if x is 1, then y is 1". We can write this symbolically as

$$\textbf{IMPLIES} \;\; = \left\{ \begin{array}{l} (1,1) \to 1 \\ (1,0) \to 0 \\ (0,1) \to 1 \\ (0,0) \to 1 \end{array} \right\}$$

Observe that $x \longrightarrow y$ is 1 whenever x is 0. If we interpret 1 as $True$ and 0 as $False$, this captures the idea that it's possible to prove anything if you start from a false assumption.

Just as we did for the case of one input variable, we can form a sequence of four symbols $wxyz$ to represent the truth tables for these operators.

$$wxyz \;\; = \left\{ \begin{array}{l} (1,1) \to w \\ (1,0) \to x \\ (0,1) \to y \\ (0,0) \to z \end{array} \right\}$$

Let's once again show that there are exactly sixteen distinct binary boolean operators. Every such operator corresponds to one of the truth tables in the collection

$$\left\{ \begin{array}{llll} 0000 & 0001 & 0010 & 0011 \\ 0100 & 0101 & 0110 & 0111 \\ 1000 & 1001 & 1010 & 1011 \\ 1100 & 1101 & 1110 & 1111 \end{array} \right\}$$

By listing these truth tables in their natural order and counting their number, we find there are sixteen.

5 Multiple Perspectives

It's easy to count the total number of truth tables when they are literally written as non-negative integers in base 2.

Theorem (Truth Tables): For n input boolean variables the total number of distinct truth tables is exactly 2^{2^n}

> given n input boolean variables; there are 2^n possible input combinations; therefore the truth tables are finite binary strings of length 2^n; and hence the total number of possible truth tables will be 2^{2^n}

The truth table theorem provides a complete classification of all possible boolean valued functions for any number of input variables, or alternatively, a complete classification of all possible boolean expressions.

The two most interesting things about this approach are that it enables us to write the truth tables in their natural order, and the exponential explosion in the number of truth tables as the number of input variables n is incremented. This approach also provides some interesting insights into the non-negative integers from a unique perspective. This kind of "multiple perspective" thinking plays an important role in mathematics in general. It's not that one or another perspective is better than the others, but rather, each different way of looking at things provides valuable information about the underlying mathematical structure, and these different perspectives compliment and reinforce each other and help us get a better idea of how the mathematical machinery works.

A useful perspective for thinking about a collection of functions is to abstract away the underlying symbols these functions act on, and represent everything with a diagram where you connect functions together with arrows to show how they are related by function composition. Let's show this function diagram for the 4 truth tables from the case $n = 1$

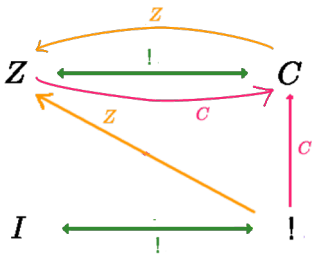

Figure 2

One way of interpreting this picture is that without anything else the

symbols 0 and 1 are completely meaningless: it's only the composition rules that give them meaning. But since the choice of the symbols 0 and 1 was arbitrary, the only thing that really matters is that they are different and there are only two of them. By removing the underlying symbols we have gained a deeper insight into the mathematical structure itself.

Problems

1. Show that $!(x \ \& \ y) = !x \ || \ !y$ (this is the **NAND** operator)
2. Show that x **NAND** $x = !x$
3. Show that $!(!x\&!y) = x||y$
4. Show that $x \longrightarrow y = (x \ \& \ y) \ || \ !x$
5. Show that $\Big(x \longrightarrow y\Big) \longrightarrow \Big(!y \longrightarrow !x\Big)$
6. Complete the table shown below

$$\left\{\begin{array}{cccc} 0000 & \underbrace{0001}_{!(x||y)} & \underbrace{0010}_{!(y\longrightarrow x)} & 0011 \\ 0100 & 0101 & 0110 & 0111 \\ 1000 & 1001 & 1010 & 1011 \\ 1100 & 1101 & 1110 & 1111 \end{array}\right\}$$

7. Complete the previous table using only the **NAND** operator