Boolean Algebra & Minimization-1 Study Notes for GATE Computer Science Engineering

Boolean algebra, named after mathematician George Boole, is a fundamental mathematical framework that deals with variables that can take only two values: true or false, represented as 1 and 0, respectively. It plays a crucial role in the design and analysis of digital circuits, computer algorithms, and programming languages. Boolean algebra provides a systematic approach to manipulate logical expressions and simplify complex digital logic circuits.

Boolean Algebra

Boolean algebra is an algebraic structure defined on a set of elements together with two binary operators (+) and (.)

- A **variable** is a symbol, for example, *A,* used to represent a logical quantity, whose value can be **0** or **1**.
- The **complement** of a variable is the inverse of a variable and is represented by an overbar, for example ' .'
- A **literal** is a variable or the complement of a variable.

Closure: For any x and y in the alphabet A, x + y and x.y are also in A.

Boolean Value

The value of the Boolean variable can be either 1 or 0. The base for this number system is 2.

Boolean Operators

There are four Boolean operators

1. AND (·) operator = The output will be like Y= (A.B)

A	**B**	**Y**
0	0	0
0	1	0

1	0	0
1	1	1

1. OR (+) operator= The output will be like Y= (A + B)

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

1. NOT (A') operator= The output will be like Y= Complement of A

A	Y
0	1
1	0

1. **XOR (⊕) operator** = The output will be like

A	B	Y
0	0	0

0	1	1
1	0	1
1	1	0

It is also known as inequality checking gate.

2.

A	B	Y
0	0	1
1	0	0
0	1	0
1	1	1

It is also known as the equality checking gate.

Operator Precedence

The operator for evaluating Boolean expression is

1. Parenthesis
2. NOT
3. AND
4. OR.

Duality

If an expression contains only the operations AND, OR and NOT. Then, the dual of that expression is obtained by replacing each AND by OR, each OR by AND, all occurrences of 1 by 0, and all occurrences of 0 by 1. Principle of duality is useful in determining the complement of a function.

Example: Logic expression: $(x \cdot y' \cdot z) + (x \cdot y \cdot z') + (y \cdot z) + 0$,

Duality of above logic expression is: $(x + y' + z) \cdot (x + y + z') \cdot (y + z) \cdot 1$

Literals remain the same while calculating the above expression.

Boolean Function

- Any Boolean functions can be formed from binary variables and the Boolean operators •, +, and ′ (for AND, OR, and NOT, respectively).
- For a given value of variable, the function can take only one value either 0 or 1.
- A Boolean function can be shown by a truth table. To show a function in a truth table we need a list of the 2^n combinations of 1's and 0's of the n binary variables and a column showing the combinations for which the function is equal to 1 or 0. So, the table will have 2^n rows and columns for each input variable and tile final output.
- A function can be specified or represented in any of the following ways:
    - A truth table
    - A circuit
    - A Boolean expression
    - SOP (Sum Of Products)
    - POS (Product of Sums)
    - Canonical SOP
    - Canonical POS
- **Important Boolean operations over Boolean values:**

$0 \bullet 0 = 0$
$1 \bullet 1 = 1$
$0 \bullet 1 = 1 \bullet 0 = 0$
$0' = 1$
$1 + 1 = 1$
$0 + 0 = 0$
$1 + 0 = 0 + 1 = 1$
$1' = 0$

Table of Some Basic Theorems

Law/Theorem	Law of Addition	Law of Multiplication
Identity Law	$x + 0 = x$	$x \cdot 1 = x$
Complement Law	$x + x' = 1$	$x \cdot x' = 0$
Idempotent Law	$x + x = x$	$x \cdot x = x$
Dominant Law	$x + 1 = 1$	$x \cdot 0 = 0$
Involution Law	$(x')' = x$	
Commutative Law	$x + y = y + x$	$x \cdot y = y \cdot x$
Associative Law	$x+(y+z) = (x+y)+z$	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$
Distributive Law	$x \cdot (y+z) = x \cdot y+x \cdot z$	$x+y\cdot z = (x+y) \cdot (x+z)$
Demorgan's Law	$(x+y)' = x' \cdot y'$	$(x \cdot y)' = x' + y'$
Absorption Law	$x + (x \cdot y) = x$	$x \cdot (x + y) = x$

Important Theorems used in Simplification

- NOT-Operation theorem: $\bar{\bar{A}} = A$
- AND-Operation theorem: $\begin{bmatrix} A \cdot A = A \\ A \cdot 1 = A \\ A \cdot 0 = 0 \\ A \cdot \bar{A} = 0 \end{bmatrix}$
- OR-Operation theorem: $\begin{bmatrix} A + A = A \\ A + 0 = A \\ A + 1 = 1 \\ A + \bar{A} = 1 \end{bmatrix}$
- Distribution theorem: A + BC = A (A + B)(A + C)

$$A + \bar{A}B = A + B$$
$$A + \bar{A}\bar{B} = A + \bar{B}$$
$$\bar{A} + AB = \bar{A} + B$$
Note: $\bar{A} + A\bar{B} = \bar{A} + \bar{B}$

- Demorgan's Theorem: $\overline{ABC} = \bar{A} + \bar{B} + \bar{C}$ $\overline{A + B + C} = \bar{A} \cdot \bar{B} \cdot \bar{C}$
- Transposition Theorem: (A + B) (A + C) = A + BC
- Consensus Theorem: This theorem is used to eliminate redundant terms. It is applicable only when a boolean function contains three variables. Each variable was used two times. Only one variable is complimented or complemented. Then the related terms so that a complemented or uncomplemented variable is the answer.

$$AB + \bar{B}C + AC = \bar{B}C + AB$$
$$\bar{A}B + \bar{B}C + \bar{A}C = \bar{B}C + \bar{A}B$$
$$AB + \bar{A}C + BC = AB + \bar{A}C$$
$$A\bar{B} + AC + BC = A\bar{B} + BC$$

Universal Logic Gate: A gate is considered as universal logic gate if it is capable to obtain all the operations just by using that single gate. The two universal logic agtes are NAND, NOR.

NAND: AND followed by NOT

it is represented by (↑).

The output will be like Y=

The truth table will be

A	**B**	**Y**
0	0	1
1	0	1
0	1	1

1	1	0

NOR:OR followed BY NOT

It is represented by (↓).

The truth table will be

A	B	Y
0	0	1
0	1	0
1	0	0
1	1	0

No of Gate Required:

Operations	NAND Gates Needed	NOR Gates Needed
NOT	1	1
OR	3	2
AND	2	3
XOR	4	5
ExNOR	5	4

Thanks!

BYJU’S Exam Prep - The Most Comprehensive App!