Boolean Algebra

'An algebra of Logic'

PRAVEEN M JIGAJINNI
PGT (Computer Science)
MTech[IT],MPhil (Comp.Sci), MCA, MSc[IT], PGDCA, ADCA, Dc. Sc. & Engg.
email: praveenkumarjigajinni@yahoo.co.in

Introduction

- Developed by English Mathematician *George Boole* in between 1815 - 1864.
- It is described as *an algebra of logic or an algebra of two values i.e True or False*.
- The term *logic* means a statement having binary decisions i.e *True/Yes or False/No*.

Application of Boolean algebra

- It is used to perform the logical operations in digital computer.
- In digital computer True represent by '1' (high volt) and False represent by '0' (low volt)
- Logical operations are performed by logical operators. The fundamental logical operators are:
  1. AND (conjunction)
  2. OR (disjunction)
  3. NOT (negation/complement)

AND operator

- It performs logical multiplication and denoted by (.) dot.

X	Y	X.Y
0	0	0
0	1	0
1	0	0
1	1	1

OR operator

- It performs logical addition and denoted by (+) plus.

X	Y	X+Y
0	0	0
0	1	1
1	0	1
1	1	1

NOT operator

- It performs logical negation and denoted by (-) bar. It operates on single variable.

X	$\overline{X}$	(means complement of x)
0	1	
1	0	

Truth Table

- Truth table is a table that contains all possible values of logical variables/statements in a Boolean expression.

No. of possible combination =

2^n, where n=number of variables used in a Boolean expression.

Truth Table

- The truth table for XY + Z is as follows:

Dec	X	Y	Z	XY	XY+Z
0	0	0	0	0	0
1	0	0	1	0	1
2	0	1	0	0	0
3	0	1	1	0	1
4	1	0	0	0	0
5	1	0	1	0	1
6	1	1	0	1	1
7	1	1	1	1	1

Tautology & Fallacy

- If the output of Booean expression is always True or 1 is called Tautology.
- If the output of Boolean expression is always False or 0 is called Fallacy.

P	P'	output (PVP')	output (PΛP')
0	1	1	0
1	0	1	0

PVP' is Tautology and PΛP' is Fallacy

Exercise

1. Evaluate the following Boolean expression using Truth Table.

(a) $X'Y'+X'Y$

(b) $X'YZ'+XY'$

(c) $XY'(Z+YZ')+Z'$

2. Verify that $P+(PQ)'$ is a Tautology.
3. Verify that $(X+Y)'=X'Y'$

Implementation

- Boolean Algebra applied in computers electronic circuits. These circuits perform Boolean operations and these are called logic circuits or logic gates.

Logic Gate

- A gate is an digital circuit which operates on one or more signals and produce single output.
- Gates are digital circuits because the input and output signals are denoted by either 1(high voltage) or 0(low voltage).
- Three type of gates are as under:
  1. AND gate
  2. OR gate
  3. NOT gate

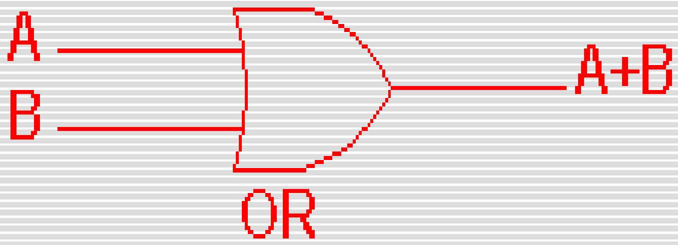
AND gate

- The AND gate is an electronic circuit that gives a high output (1) only if all its inputs are high.
- AND gate takes two or more input signals and produce only one output signal.

Input A	*Input B*	*Output AB*
0	0	0
0	1	0
1	0	0
1	1	1

OR gate

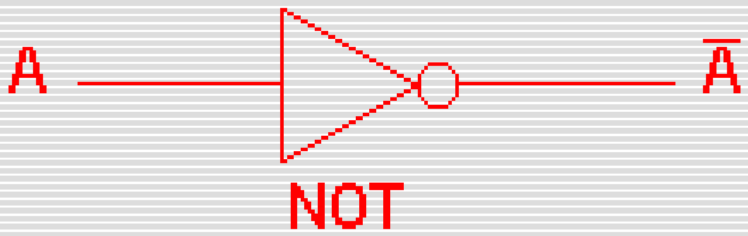
- The OR gate is an electronic circuit that gives a high output (1) if one or more of its inputs are high.
- OR gate also takes two or more input signals and produce only one output signal.



Input A	*Input B*	*Output A+B*
0	0	0
0	1	1
1	0	1
1	1	1

NOT gate

- The NOT gate is an electronic circuit that gives a high output (1) if its input is low .
- NOT gate takes only one input signal and produce only one output signal.
- The output of NOT gate is complement of its input.
- It is also called inverter.



Input A	*Output $\overline{A}$*
0	1
1	0

Principal of Duality

- In Boolean algebras the duality Principle can be is obtained by interchanging AND and OR operators and replacing 0's by 1's and 1's by 0's. Compare the identities on the left side with the identities on the right.

Example

X.Y+Z' = (X'+Y').Z

Basic Theorem of Boolean Algebra

T1 : Properties of 0

(a) $0 + A = A$

(b) $0\,A = 0$

T2 : Properties of 1

(a) $1 + A = 1$

(b) $1\,A = A$

Basic Theorem of Boolean Algebra

T3 : Commutative Law

(a) $A + B = B + A$

(b) $A\,B = B\,A$

T4 : Associate Law

(a) $(A + B) + C = A + (B + C)$

(b) $(A\,B)\,C = A\,(B\,C)$

T5 : Distributive Law

(a) $A\,(B + C) = A\,B + A\,C$

(b) $A + (B\,C) = (A + B)\,(A + C)$

(c) A+A’B = A+B

Basic Theorem of Boolean Algebra

T6 : Indempotence (Identity) Law

(a) $A + A = A$

(b) $A\,A = A$

T7 : Absorption (Redundance) Law

(a) $A + A\,B = A$

(b) $A\,(A + B) = A$

Basic Theorem of Boolean Algebra

T8 : Complementary Law

(a) $X+X'=1$

(b) $X.X'=0$

T9 : Involution

(a) $x'' = x$

T10 : De Morgan's Theorem

(a) $(X+Y)'=X'.Y'$

(b) $(X.Y)'=X'+Y'$

Exercise

Q 1. State & Verify De Morgan's Law by using truth table and algebraically.

Q 2. State and verify distributive law.

Q 3. Draw a logic diagram for the following expression:

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

(b) $(a+b).(a+b').c$

Representation of Boolean expression

Boolean expression can be represented by either

(i)Sum of Product(SOP) form or

(ii)Product of Sum (POS form)

e.g.

AB+AC → SOP
(A+B)(A+C) → POS

In above examples both are in SOP and POS respectively but they are not in Standard SOP and POS.

Canonical form of Boolean Expression (Standard form)

- In standard SOP and POS each term of Boolean expression must contain all the literals (with and without bar) that has been used in Boolean expression.
- If the above condition is satisfied by the Boolean expression, that expression is called Canonical form of Boolean expression.

Canonical form of Boolean Expression (Standard form) Contd..

- In Boolean expression **AB+AC** the literal C is mission in the 1st term **AB** and B is mission in 2nd term **AC**. That is why AB+AC is not a Canonical SOP.

Canonical form of Boolean Expression (Standard form) Contd..

Convert AB+AC in Canonical SOP (Standard SOP)

Sol. $AB + AC$

$AB(C+C') + AC(B+B')$

$ABC+ABC'+ABC+AB'C$ **Distributive law**

$ABC+ABC'+AB'C$

Canonical form of Boolean Expression (Standard form) Contd..

Convert (A+B)(A+C) in Canonical SOP (Standard SOP)

Sol. (A+B).(A+C)

(A+B)+(C.C’) . (A+C)+(B.B’)

(A+B+C).(A+B+C’).(A+B+C)(A+B’+C) **Distributive law**

(A+B+C).(A+B+C’)(A+B’+C) **Remove duplicates**

Canonical form of Boolean Expression (Standard form) Contd..

Minterm and Maxterm

Individual term of Canonical Sum of Products (SOP) is called Minterm. In otherwords minterm is a product of all the literals (with or without bar) within the Boolean expression.

Individual term of Canonical Products of Sum (POS) is called Maxterm. In otherwords maxterm is a sum of all the literals (with or without bar) within the Boolean expression.

Minterms & Maxterms for 2 variables (Derivation of Boolean function from Truth Table)

x	y	Index	Minterm	Maxterm
0	0	0	$m_0 = x' y'$	$M_0 = x + y$
0	1	1	$m_1 = x' y$	$M_1 = x + y'$
1	0	2	$m_2 = x y'$	$M_2 = x' + y$
1	1	3	$m_3 = x y$	$M_3 = x' + y'$

The minterm m_i should evaluate to 1 for each combination of x and y.
The maxterm is the complement of the minterm

Minterms & Maxterms for 3 variables

x	y	z	Index	Minterm	Maxterm
0	0	0	0	$m_0 = \bar{x}\,\bar{y}\,\bar{z}$	$M_0 = x + y + z$
0	0	1	1	$m_1 = \bar{x}\,\bar{y}\,z$	$M_1 = x + y + \bar{z}$
0	1	0	2	$m_2 = \bar{x}\,y\,\bar{z}$	$M_2 = x + \bar{y} + z$
0	1	1	3	$m_3 = \bar{x}\,y\,z$	$M_3 = x + \bar{y} + \bar{z}$
1	0	0	4	$m_4 = x\,\bar{y}\,\bar{z}$	$M_4 = \bar{x} + y + z$
1	0	1	5	$m_5 = x\,\bar{y}\,z$	$M_5 = \bar{x} + y + \bar{z}$
1	1	0	6	$m_6 = x\,y\,\bar{z}$	$M_6 = \bar{x} + \bar{y} + z$
1	1	1	7	$m_7 = x\,y\,z$	$M_7 = \bar{x} + \bar{y} + \bar{z}$

Maxterm M_i is the complement of minterm m_i

$$M_i = \overline{m_i} \text{ and } m_i = \overline{M_i}$$

Solved Problem

Prob. Find the minterm designation of $XY'Z'$

Sol. Subsitute 1's for non barred and 0's for barred letters

Binary equivalent = 100

Decimal equivalent = 4

Thus $XY'Z'=m_4$

Purpose of the Index

- Minterms and Maxterms are designated with an index
- The index number corresponds to a binary pattern
- The index for the minterm or maxterm, expressed as a binary number, is used to determine whether the variable is shown in the true or complemented form
- For Minterms:
  - ‘1’ means the variable is “Not Complemented” and
  - ‘0’ means the variable is “Complemented”.
- For Maxterms:
  - ‘0’ means the variable is “Not Complemented” and
  - ‘1’ means the variable is “Complemented”.

Solved Problem

Write SOP form of a Boolean Function F, Which is represented by the following truth table.

Sum of minterms of entries that evaluate to '**1**'

x	y	z	F	Minterm
0	0	0	0	
0	0	1	**1**	$m_1 = x'y'z$
0	1	0	0	
0	1	1	0	
1	0	0	0	
1	0	1	0	
1	1	0	**1**	$m_6 = xyz'$
1	1	1	**1**	$m_7 = xyz$

Focus on the '1' entries

$$F = m_1 + m_6 + m_7 = \sum(1, 6, 7) = \bar{x}\,\bar{y}\,z + x\,y\,\bar{z} + x\,y\,z$$

Exercise

1. Write POS form of a Boolean Function F, Which is represented by the following truth table

x	y	z	F
0	0	0	1
0	0	1	**1**
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	**0**
1	1	1	**1**

2. Write equivalent canonical Sum of Product expression for the following Product of Sum Expression: $F(X,Y,Z)=\Pi(1,3,6,7)$

Minimization of Boolean Expression

- Canonical SOP (Sum of Minterms) and POS (Product of Maxterm) is the derivation/expansion of Boolean Expression.
- Canonical forms are not usually minimal.
- Minimization of Boolean expression is needed to simplify the Boolean expression and thus reduce the circuitry complexity as it uses less number of gates to produce same output that can by taken by long canonical expression.

Minimization of Boolean Expression (Contd…)

- **Two method can by applied to reduce the Boolean expression –**

i) Algebraic
ii) Using Karnaugh Map (K-Map).

Minimization of Boolean Expression (Contd...)

- **Algebraic Method**

  - **The different Boolean rules and theorems are used to simplify the Boolean expression in this method.**

Minimization of Boolean Expression (Contd…)

Solved Problem

Minimize the following Boolean Expression:

1. $a'bc + ab'c' + ab'c + abc' + abc$

   $= a'bc + ab' + ab$

   $= a'bc + a$

2. $AB'CD' + AB'CD + ABCD' + ABCD$

   $= AB'C + ABC$

   $= AC$

Minimization of Boolean Expression (Contd…)

Exercise

A. Minimize the following Boolean Expression:

1. $X'Y'Z' + X'YZ' + XY'Z' + XYZ'$
2. $a(b + b'c + b'c')$

B. Prove algebraically that

1. $(x+y+z)(x'+y+z)=y+z$
2. $A+A'B'=A+B'$

Minimization of Boolean Expression (Contd…)

Karnaugh Map

The Karnaugh map (K-map for short), Maurice Karnaugh's 1953 refinement of Edward Veitch's 1952 Veitch diagram, is a method to simplify Boolean algebra expressions. K-map is

- K-Maps are a convenient way to simplify Boolean Expressions.
- They can be used for up to 4 or 5 variables.
- They are a visual representation of a truth table.

Truth table to K-Map (2 variable minterm)

A	B	P
0	0	1
0	1	1
1	0	0
1	1	1

B / A		B′ 0	B 1
A′	0	1	1
A	1		1

The expression is:

$\overline{A}.\overline{B} + \overline{A}.B + A.B$

minterms are represented by a 1 in the corresponding location in the K map.

K-Maps (2 Variables k-map contd…)

- Adjacent 1's can be "paired off"
- Any variable which is both a 1 and a zero in this pairing can be eliminated
- Pairs may be adjacent horizontally or vertically

The expression is:
A'.B' + A'.B + A.B

A \ B	0	1
0	1	1
1		1

a pair

another pair

B is eliminated, leaving $\overline{A}$ as the term

A is eliminated, leaving B as the term

After reduction the expression becomes $\overline{A}$ + B

Three Variable K-Map

A	B	C	P
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

$$\bar{A}.B.\bar{C} + A.\bar{B}.\bar{C} + A.\bar{B}.C$$

BC →

A \ BC	00	01	11	10
0				1
1	1			1

One square filled in for each minterm.

Notice the code sequence:
00 01 11 10 – a Gray code.

Three Variable K-Map (Contd…)

Grouping the Pairs

equates to $B.\bar{C}$ as A is eliminated.

A \ BC	00	01	11	10
0				1
1	1			1

Our truth table simplifies to $A.\bar{C} + B.\bar{C}$ as before.

Here, we can "wrap around" and this pair equates to $A.\bar{C}$ as B is eliminated.

Three Variable K-Map (Contd…)

Expression is ABC+A′BC′+A′BC+ABC′

Groups of 4 in a block can be used to eliminate two variables:

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

A \ BC	00	01	11	10
0			1	1
1			1	1

Groups of 4

QUAD = A’BC+A’BC’+ABC+ABC’

= A’B+AB

=B

Karnaugh Maps - Four Variable K-Map

AB \ CD	00	01	11	10
00	$\overline{A}.\overline{B}.\overline{C}.\overline{D}$	$\overline{A}.\overline{B}.\overline{C}.D$	$\overline{A}.\overline{B}.C.D$	$\overline{A}.\overline{B}.C.\overline{D}$
01	$\overline{A}.B.\overline{C}.\overline{D}$	$\overline{A}.B.\overline{C}.D$	$\overline{A}.B.C.D$	$\overline{A}.B.C.\overline{D}$
11	$A.B.\overline{C}.\overline{D}$	$A.B.\overline{C}.D$	$A.B.C.D$	$A.B.C.\overline{D}$
10	$A.\overline{B}.\overline{C}.\overline{D}$	$A.\overline{B}.\overline{C}.D$	$A.\overline{B}.C.D$	$A.\overline{B}.C.\overline{D}$

K-Map
Reduction Rule

To reduce the Boolean expression, first we have to mark pairs, quads and octets.

Pair – remove one variable
Quad – remove two variables
Octet – remove three variables

Imp – To get the optimum reduction, priority is given to octet first, then quad and then pair.

Karnaugh Maps - Four Variable K-Map

Octet Reduction

CD AB	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1	1	1	1
A'B[01]	1	1	1	1
AB[11]				
AB'[10]				

A'B'[00]	1	1		
A'B[01]	1	1		
AB[11]	1	1		
AB'[10]	1	1		

Karnaugh Maps - Four Variable K-Map

Octet Reduction

AB \ CD	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1	1	1	1
A'B[01]				
AB[11]				
AB'[10]	1	1	1	1

A'B'[00]	1			1
A'B[01]	1			1
AB[11]	1			1
AB'[10]	1			1

Karnaugh Maps - Four Variable K-Map

Quad Reduction

AB \ CD	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1	1		1
A'B[01]	1	1		1
AB[11]				
AB'[10]				

A'B'[00]	1			
A'B[01]	1			
AB[11]	1			
AB'[10]	1			

Karnaugh Maps - Four Variable K-Map

Quad Reduction

AB \ CD	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1	1		1
A'B[01]	1	1		1
AB[11]				
AB'[10]				

A'B'[00]	1			
A'B[01]	1			
AB[11]	1			
AB'[10]	1			

Karnaugh Maps - Four Variable K-Map

Quad Reduction

CD AB	C'D'[00]	C'D[01]	CD[11]	CD'[10]
A'B'[00]	1			1
A'B[01]	1			1
AB[11]				
AB'[10]				

A'B'[00]	1			1
A'B[01]				
AB[11]				
AB'[10]	1			1