Lect 07

Boolean Functions: Canonical Form and Minimization

CS221: Digital Design

Dr. A. Sahu
Dept of Comp. Sc. & Engg.
Indian Institute of Technology Guwahati

1

Outline

- Boolean Functions
  - Truth Table Conversion and Vice Versa
- Canonical form of Function
  - SOP and POS
- Gray Code and Hamming Distances
- K-Maps

2

Converting among Representations

Q: Convert to equation

a	b	F	Term
0	0	**1**	**a'b'**
0	1	**1**	**a'b**
1	0	**0**	
1	1	**0**	

F = a'b' + a'b

Converting among Representations

Q: Convert to equation

a	b	c	F	Term
0	0	0	**0**	
0	0	1	**0**	
0	1	0	**0**	
0	1	1	**0**	
1	0	0	**0**	
1	0	1	**1**	**ab'c**
1	1	0	**1**	**abc'**
1	1	1	**1**	**abc**

F = ab'c + abc' + abc

Converting among Representations

Q: Convert to truth table: F = a'b' + a'b

Inputs				Output
a	b	a'b'	a'b	F
0	0	1	0	1
0	1	0	1	1
1	0	0	0	0
1	1	0	0	0

Standard Representation

- How to determine two functions are the same?
  - Use algebraic methods
  - But if we failed, does that prove *not* equal? No.
- Solution: Convert to truth tables
  - Only ONE **truth table** representation of given same functions: Standard representation

Standard Representation: Truth Table

Only ONE truth table representation of given same functions: Standard representation

F=ab+a'		
a	b	F
0	0	**1**
0	1	**1**
1	0	**0**
1	1	**1**

F=a'b'+a'b+ab		
a	b	F
0	0	**1**
0	1	**1**
1	0	**0**
1	1	**1**

Same

Represent TT in efficiently/elegantly?

- **Truth tables too big for numerous inputs ☹ ☹**
- Use standard form of equation instead
  - Known as **canonical form**
  - English meaning of "Canonical" : simplest or standard in mathematics
- **Regular algebra**: group terms of polynomial by power
  - $ax^2 + bx + c$
  - $(3x^2 + 4x + 2x^2 + 3 + 1 \rightarrow 5x^2 + 4x + 4)$

Boolean Algebra Canonical Form

- Truth tables too big for numerous inputs
- Use standard form of equation instead: **Canonical**
- Boolean algebra: **create sum of minterms**
  - ***Minterm***: product term with every function literal appearing exactly once, in true or complemented form
  - Just multiply-out equation until sum of product terms
  - Then expand each term until all terms are minterms

Canonical Form -- Sum of Minterms

Determine if F(a,b)=ab+a' is same function as F(a,b) = a'b'+a'b+ab by to canonical form.

F = ab+a' (already **sum of products**)

F = ab + a'(b+b') (expanding term)

F = ab + a'b + a'b' (it is **canonical form**)

Minterm: product term with every function literal appearing exactly once, in true or complemented form

Canonical form and Standard Form

- Canonical forms
  - Sum of minterms (SOM)
  - Product of maxterms (POM)
- Standard forms (may use less gates)
  - Sum of products (SOP)
  - Product of sums (POS)
- **SOP form may not be in Canonical Form**

F = ab+a' (already **sum of products:SOP**)
F = ab + a'(b+b') (expanding term)
F = ab + a'b + a'b' (it is **canonical form:SOM**)

Canonical Forms

- It is useful to specify Boolean functions in a form that:
  - Allows comparison for equality.
  - Has a correspondence to the truth tables
- Canonical Forms in common usage:
  - Sum of Minterms (SOM)
  - Product of Maxterms (POM)

Minterms in SOP

- **Product term** is a term where literals are ANDed
  - Example: x'y', xz, xyz, …
- **minterm** : A product term in which all variables appear exactly once, in normal or complemented form
  - Example: F(x,y,z) has 8 minterms

    x'y'z', x'y'z, x'yz', …

Minterms

- Function with **n** variables has $\mathbf{2^n}$ minterms
- A minterm equals 1 at exactly one input combination and is equal to 0 otherwise
  - Example: x'y'z' = 1 only when x=0, y=0, z=0
- Minterm is denoted as m_i where i corresponds the input combination at which this minterm is equal to 1

Example: 2 Variable Minterms

- Two variables (X and Y) produce 2x2=4 combinations

  XY (both normal)

  XY' (X normal, Y complemented)

  X'Y (X complemented, Y normal)

  X'Y' (both complemented)

Maxterms

- Maxterms are OR terms with every variable in true or complemented form.

  X+Y (both normal)

  X+Y' (x normal, y complemented)

  X'+Y (x complemented, y normal)

  X'+Y' (both complemented)

Maxterms and Minterms

- Two variable minterms and maxterms.

Index	Minterm	Maxterm
0	x' y'	x + y
1	x 'y	x + y'
2	x y'	x' + y
3	x y	x' + y'

- The index above is important for describing which variables in the terms are true and which are complemented.

Minterms : three variables

X	Y	Z	Term	Symbol	m_0	m_1	m_2	m_3	m_4	m_5	m_6	m_7
0	0	0	x'y'z'	m_0	1	0	0	0	0	0	0	0
0	0	1	x'y'z	m_1	0	1	0	0	0	0	0	0
0	1	0	x'yz'	m_2	0	0	1	0	0	0	0	0
0	1	1	x'yz	m_3	0	0	0	1	0	0	0	0
1	0	0	xy'z'	m_4	0	0	0	0	1	0	0	0
1	0	1	xy'z	m_5	0	0	0	0	0	1	0	0
1	1	0	xyz'	m_6	0	0	0	0	0	0	1	0
1	1	1	xyz	m_7	0	0	0	0	0	0	0	1

Variable complemented if 0
Variable uncomplemented if 1

m_i indicated the i^{th} minterm
i indicates the binary combination
m_i is equal to 1 for ONLY THAT combination

Maxterms in POS

- **Sum term** : A term where literals are ORed.
  - Example: x'+y', x+z, x+y+z, …
- **Maxterm** : a sum term in which all variables appear exactly once, in normal or complemented form
  - Example: F(x,y,z) has 8 maxterms
    (x+y+z), (x+y+z'), (x+y'+z), …

Maxterms

- Function with n variables has 2^n maxterms
- A maxterm equals 0 at exactly one input combination and is equal to 1 otherwize
  - Example: (x+y+z) = 0 only when x=0, y=0, z=0
- A maxterm is denoted as M_i where i corresponds the input combination at which this maxterm is equal to 0

Maxterms : three variable

X	Y	Z	Term	Sym	M_0	M_1	M_2	M_3	M_4	M_5	M_6	M_7
0	0	0	X+Y+Z	M_0	0	1	1	1	1	1	1	1
0	0	1	X+Y+Z'	M_1	1	0	1	1	1	1	1	1
0	1	0	X+Y'+Z	M_2	1	1	0	1	1	1	1	1
0	1	1	X+Y'+Z'	M_3	1	1	1	0	1	1	1	1
1	0	0	X'+Y+Z	M_4	1	1	1	1	0	1	1	1
1	0	1	X'+Y+Z'	M_5	1	1	1	1	1	0	1	1
1	1	0	X'+Y'+Z	M_6	1	1	1	1	1	1	0	1
1	1	1	X'+Y'+Z'	M_7	1	1	1	1	1	1	1	0

Variable complemented if 1
Variable not complemented if 0

M_i indicated the i^{th} maxterm i indicates the binary combination M_i is equal to 0 for ONLY THAT combination

Expressing Functions with Minterms

- Boolean function can be expressed algebraically from a give truth table
  - Forming sum of ALL the minterms that produce 1 in the function

Example : Consider the function defined by the truth table

F(X,Y,Z)= X'Y'Z' + X'YZ' + XY'Z + XYZ

$= m_0 + m_2 + m_5 + m_7$

$= \sum m(0, 2, 5, 7)$

X	Y	Z	m	F
0	0	0	m_0	1
0	0	1	m_1	0
0	1	0	m_2	1
0	1	1	m_3	0
1	0	0	m_4	0
1	0	1	m_5	1
1	1	0	m_6	0
1	1	1	m_7	1

Expressing Functions with Maxterms

- Boolean function : Expressed algebraically from a give truth table
- By forming logical product (AND) of ALL the maxterms that produce 0 in the function

Example:
Consider the function defined by the truth table

X	Y	Z	M	F	F'
0	0	0	M_0	1	0
0	0	1	M_1	0	1
0	1	0	M_2	1	0
0	1	1	M_3	0	1
1	0	0	M_4	0	1
1	0	1	M_5	1	0
1	1	0	M_6	0	1
1	1	1	M_7	1	0

$F(X,Y,Z) = \Pi\ M(1,3,4,6)$

Applying DeMorgan

$F' = m_1 + m_3 + m_4 + m_6 = \sum m(1,3,4,6)$

$F = F'' = [m_1 + m_3 + m_4 + m_6]'$

$= m_1'.m_3'.m_4'.m_6'$

$= M_1.M_3.M_4.M_6$

$= \Pi\ M(1,3,4,6)$

Note the indices in this list are those that are missing from the previous list in Σm(0,2,5,7)

Sum of Minterms vs Product of Maxterms

- A function can be expressed algebraically as:
  - The sum of minterms
  - The product of maxterms
- Given the truth table, writing F as
  - $\sum m_i$ – for all minterms that produce 1 in the table, or
  - ΠM_i – for all maxterms that produce 0 in the table
- Minterms and Maxterms are complement of each other.

Example: minterm & maxterm

- Write E = Y' + X'Z' in the form of $\sum m_i$ and $\prod M_i$?
- **Method1**

First construct the Truth Table as shown

E = $\sum$m(0,1,2,4,5), and

E = $\prod$M(3,6,7)

X	Y	Z	m	M	E
0	0	0	m_0	M_0	1
0	0	1	m_1	M_1	1
0	1	0	m_2	M_2	1
0	1	1	m_3	M_3	0
1	0	0	m_4	M_4	1
1	0	1	m_5	M_5	1
1	1	0	m_6	M_6	0
1	1	1	m_7	M_7	0

Example (Cont.)

Solution: **Method2_a**

E = Y' + X'Z'
= Y'(X+X')(Z+Z') + 'Z'(Y+Y')
= (XY'+X'Y')(Z+Z') + X'YZ'+ X'Z'Y'
= Y'Z+X'Y'Z+XY'Z'+X'Y'Z'+ X'YZ'+X'Z'Y'
= $m_5 + m_1 + m_4 + m_0 + m_2 + m_0$
= $m_0 + m_1 + m_2 + m_4 + m_5$
= $\sum$ m(0,1,2,4,5)

To find the form $\prod$Mi, consider the remaining indices

E = $\prod$M(3,6,7)

Solution: **Method2_b**

E = Y' + X'Z'
E' = Y(X+Z)
= YX + YZ
= YX(Z+Z') + YZ(X+X')
= XYZ+XYZ'+X'YZ

E = (X'+Y'+Z')(X'+Y'+Z)(X+Y'+Z')
= $M_7 . M_6 . M_3$
= $\prod$M(3,6,7)

To find the form Sm_i, consider the remaining indices

E = $\sum$ m(0,1,2,4,5)

Canonical Forms

- The sum of minterms and the product of maxterms forms are known as the **canonical forms** of a function.

Standard Forms

- Sum of Products (SOP) and Product of Sums (POS) are also standard forms
  - AB+CD = (A+C)(B+C)(A+D)(B+D)
- The sum of min-terms is a special case of the SOP form, where all product terms are min-terms
- The product of max-terms is a special case of the POS form, where all sum terms are max-terms

SOP and POS Conversion

SOP → POS

F = AB + CD
= (AB+C)(AB+D)
= (A+C)(B+C)(AB+D)
= (A+C)(B+C)(A+D)(B+D)

Hint 1: Use id15: X+YZ=(X+Y)(X+Z)
Hint 2: Factor

POS → SOP

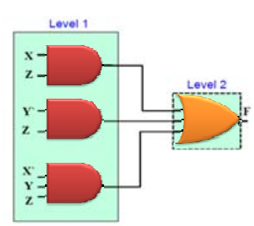
F = (A'+B)(A'+C)(C+D)
= (A'+BC)(C+D)
= A'C+A'D+BCC+BCD
= A'C+A'D+BC+BCD
= A'C+A'D+BC

Hint 1: Use id15 (X+Y)(X+Z)=X+YZ
Hint 2: Multiply

Implementation of SOP

F(X,Y,Z) = XZ+Y'Z+X'YZ

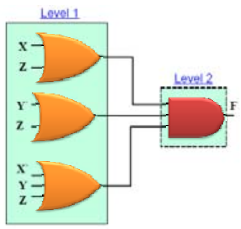
- Any SOP expression can be implemented using 2-levels of gates
- The 1st level consists of AND gates, and the 2nd level consists of a single OR gate
- Also called 2-level Circuit



Implementation of POS

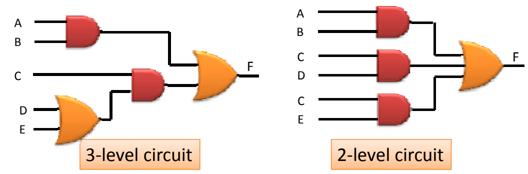
$F(X,Y,Z) = (X+Z)(Y'+Z)(X'+Y+Z)$

- Any POS expression can be implemented using 2-levels of gates
- The 1st level consists of OR gates, and the 2nd level consists of a single AND gate
- Also called 2-level Circuit



Implementation of SOP

- Consider F = AB + C(D+E)
  - This expression is NOT in the sum-of-products form
  - Use the identities/algebraic manipulation to convert to a standard form (sum of products), as in F = AB + CD + CE
- Logic Diagrams:



Canonical Forms

- It is useful to specify Boolean functions in a form that:
  - **Allows comparison for equality.**
  - **Has a correspondence to the truth tables**
- Canonical Forms in common usage:
  - Sum of Minterms (SOM)/Sum of Product (SOP)
  - Product of Maxterms/Sum (POM)/POS

Simplification: Theorem method

$E = \sum m(0,1,2,4,5)$

$= m_0 + m_1 + m_2 + m_4 + m_5$

$= m_5 + m_1 + m_4 + m_0 + m_2 + m_0$

$= \mathbf{XY'Z+X'Y'Z+XY'Z'+X'Y'Z'+X'YZ'+X'Z'Y'}$

$= (XY'+X'Y')(Z+Z') + X'YZ' + X'Z'Y'$

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

$= \mathbf{Y' + X'Z'}$

Simplified one: Require less Gates and faster 2 Level

Both are in SOP format : 2 level

Simplification of Boolean Functions

- An implementation of a Boolean Function requires the use of logic gates.
- A smaller number of gates, with each gate (other then Inverter) having less number of inputs, may reduce the cost of the implementation.
- There are 2 methods for simplification of Boolean functions.

35

Simplification of Boolean Functions: Two Methods

- **Algebraic method** by using Identities & Theorem
- **Graphical method** by using Karnaugh Map method
  - The K-map method is easy and straightforward.
  - A K-map for a function of n variables consists of 2^n cells, and,
  - in every row and column, two adjacent cells should differ in the value of only one of the logic variables.

Maurice Karnaugh, Bell Lab, 1954.

36

Karnaugh Map Method

- A graphical method of simplifying logic equations or truth tables.
- Also called a K map
- Theoretically can be used for any number of input variables, but **practically limited to 5 or 6 variables**.

Karnaugh Map Advantages

- Minimization can be done more systematically
- Much simpler to find minimum solutions
- Easier to see what is happening (graphical)
- Almost always used instead of boolean minimization.

Gray Codes

- Gray code is a binary value encoding in which adjacent values only differ by one bit

2-bit Gray Code
00
01
11
10

Truth Table Adjacencies

F = A'

A	B	F
0	0	1
0	1	1
1	0	0
1	1	0

These are *adjacent in a gray code sense*
- they differ by 1 bit

We can apply XY + XY' = X
A'B' + A'B = A'(B'+B) = A'(1) = A'

F = B

A	B	F
0	0	0
0	1	1
1	0	0
1	1	1

Same idea: A'B + AB = B

Key idea:

Gray code adjacency allows use of simplification theorems

Problem:
Physical adjacency in truth table does not indicate gray code adjacency

Karnaugh Map Method

- The truth table values are placed in the K map.
- **Adjacent K map square differ in only one variable both horizontally and vertically.**
- **The pattern from top to bottom and left to right must be in the form**
- A SOP expression can be obtained by ORing all squares that contain a 1.

A'B', A'B, AB, AB'
00, 01, 11, 01

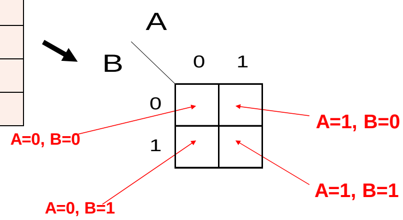
Filling of Karnaugh Map

Why not: A'B', A'B, AB', AB
00, 01, 10, 11
Only two adjacent can be grouped
Group Reduce a variable: AB'+AB=A(B'+B)=A

A'B', A'B, AB, AB'
00, 01, 11, 01
All 4 Adjacent can be grouped

2-Variable Karnaugh Map

A	B	F
0	0	
0	1	
1	0	
1	1	



A different way to draw a truth table: by folding it

Karnaugh Map: 2, 3, 4 Variable
Gray coding

00	01
10	11

000	001	011	010
100	101	111	110

0000	0001	0011	0010
0100	0101	0111	0110
1100	1101	1111	1110
1000	1001	1011	1010

Karnaugh Map: 2, 3, 4 Variable
Gray coding, Decimal

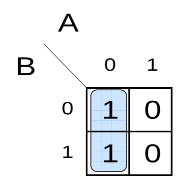
0	1
2	3

0	1	3	2
4	5	7	6

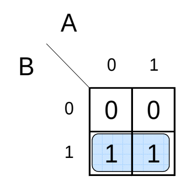
0	1	3	2
4	5	7	6
12	13	15	14
8	9	11	10

Karnaugh Map

- In a K-map, physical adjacency **does** imply gray code adjacency



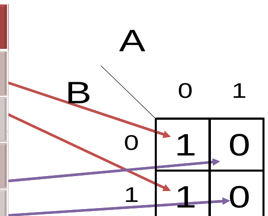
F =A'B' + A'B = A'



F = A'B + AB = B

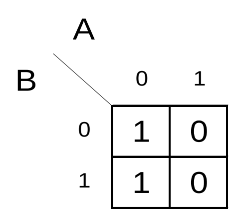
2-Variable Karnaugh Map

A	B	index	F
0	0	0	1
0	1	1	1
1	0	2	0
1	1	3	0



2-Variable Karnaugh Map

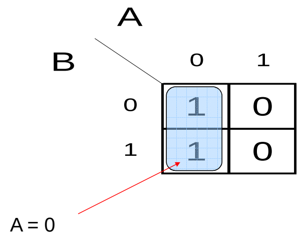
A	B	index	F
0	0	0	1
0	1	1	1
1	0	2	0
1	1	3	0



F = A'B' + A'B = A'

2-Variable K Map: Grouping

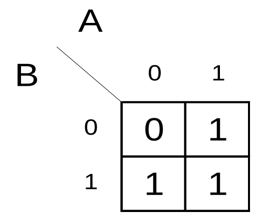
A	B	index	F
0	0	0	1
0	1	1	1
1	0	2	0
1	1	3	0



F = A’

Another Example

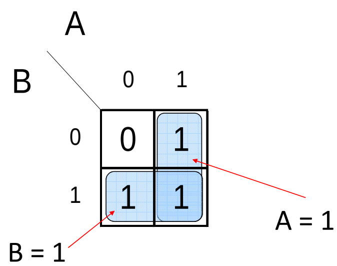
A	B	index	F
0	0	0	0
0	1	1	1
1	0	2	1
1	1	3	1



F = A’B + AB’ + AB
= (A’B + AB) + (AB’ + AB)
= A + B

Another Example

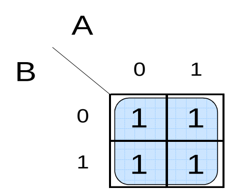
A	B	index	F
0	0	0	0
0	1	1	1
1	0	2	1
1	1	3	1



F = A + B

Yet Another Example

A	B	index	F
0	0	0	1
0	1	1	1
1	0	2	1
1	1	3	1



F = 1

Groups of more than two 1’s can be combined