MATEMATIKA DISKRIT II
(2 SKS)

Rabu, 18.50 – 20.20
Ruang Hard Disk
PERTEMUAN I

Dosen
Lie Jasa

Tujuan

1. Memahami bagaimana konsep aljabar Boolean dan penerapannya dalam sirkuit kombinatorial / Rangkaian.
2. Memahami konsep relasi dan bagaimana membangunnnya.
3. Memahami penggunaan algoritma dan mampu menulis algoritma untuk permasalahan-permasalahan tertentu.
4. Memahami konsep dasar grafik dan trees

Materi

Pemahaman lebih lanjut tentang Matematika Diskrit :

- **Aljabar Boolean.**
- **Sirkuit Kombinatorial**
- **Relasi Rekurensi**
- **Algoritma**
- **Graph**
- **Tree**

Referensi

Judul **Discrete Mathematics ISBN 0131176862**

Penerbit **Prentice Hall, 6th Edition**

Pengarang **Johnsonbaugh, R**

Terbitan 2005.

Judul **Discrete Mathematics and its Application**

Penerbit **Mac Graw-Hill, 6th Edition**

Pengarang **Rosen, K.H.,**

Singapore, 2007

JADWAL KULIAH **Tatap Muka**

MAR			APR					MEI				JUN		
11	18	25	1	8	15	22	29	6	13	20	27	3	10	17
1			2	3	4	5	6	7	8	9	10	11	12	13

Responsi / Bimbingan

Melalui : e-mail, SMS, Ketemu langsung, Telpon

Sistem Penilaian

ABSEN	**:**	**10% (kehadiran)**
QUIST	**:**	**10% (tidak terjadwal)**
TUGAS	**:**	**15% (ditentukan)**
UTS	**:**	**30% (terjadwal)**
UAS	**:**	**35% (terjadwal)**
TOTAL	**:**	**100%**

(NILAI TERTINGGI A TERENDAH D)

TARGET PEMBELAJARAN

1. **Memahami secara baik aljabar boolean dan Rangkaian Kombinatorial**
2. **Memahami secara baik konsep Relasi**
3. **Mampu merancang menulis Agoritma.**
4. **Mamahami konsep Graph dan Tree**

Tugas-mk-lie@ee.unud.ac.id
hp. 08123931535
http://nic.unud.ac.id/~lie_jasa

Kirim email:
Nama, NIM, Alamat Email, Kelas, Mata Kuliah, HP

Boolean Algebra

- VERY nice machinery used to manipulate (simplify) Boolean functions
- George Boole (1815-1864): “An investigation of the laws of thought”
- Terminology:
  - *Literal:* A variable or its complement
  - *Product term :* literals connected by •
  - *Sum term:* literals connected by +

Boolean Algebra Properties

Let X: boolean variable, 0,1: constants

1. X + 0 = X -- Zero Axiom
2. X • 1 = X -- Unit Axiom
3. X + 1 = 1 -- Unit Property
4. X • 0 = 0 -- Zero Property

Boolean Algebra Properties (cont.)

Let X: boolean variable, 0,1: constants

5. $X + X = X$ -- Idempotence
6. $X \bullet X = X$ -- Idempotence
7. $X + X' = 1$ -- Complement
8. $X \bullet X' = 0$ -- Complement
9. $(X')' = X$ -- Involution

Unchanged in value following multiplication by itself

The Duality Principle

- The dual of an expression is obtained by exchanging (• and +), and (1 and 0) in it, provided that the precedence of operations is not changed.
- Cannot exchange x with x'
- Example:
  - Find H(x,y,z), the dual of $F(x,y,z) = x'yz' + x'y'z$
  - $H = (x'+y+z')(x'+y'+z)$
- Dual does not always equal the original expression
- If a Boolean equation/equality is valid, its dual is also valid

The Duality Principle (cont.)

With respect to duality, Identities 1 – 8 have the following relationship:

1. $X + 0 = X$	2. $X \bullet 1 = X$	(dual of 1)
3. $X + 1 = 1$	4. $X \bullet 0 = 0$	(dual of 3)
5. $X + X = X$	6. $X \bullet X = X$	(dual of 5)
7. $X + X' = 1$	8. $X \bullet X' = 0$	(dual of 8)

More Boolean Algebra Properties

Let X,Y, and Z: boolean variables

10. $X + Y = Y + X$	11. $X \bullet Y = Y \bullet X$	-- Commutative
12. $X + (Y+Z) = (X+Y) + Z$	13. $X\bullet(Y\bullet Z) = (X\bullet Y)\bullet Z$	-- Associative
14. $X\bullet(Y+Z) = X\bullet Y + X\bullet Z$	15. $X+(Y\bullet Z) = (X+Y) \bullet (X+Z)$	-- Distributive
16. $(X + Y)' = X' \bullet Y'$	17. $(X \bullet Y)' = X' + Y'$	-- DeMorgan's

In general,

$(X_1 + X_2 + \ldots + X_n)' = X_1'\bullet X_2' \bullet \ldots \bullet X_n'$, and

$(X_1\bullet X_2\bullet \ldots \bullet X_n)' = X_1' + X_2' + \ldots + X_n'$

Absorption Property (Covering)

1. $x + x \bullet y = x$
2. $x \bullet (x+y) = x$ (dual)

- **Proof:**

$x + x \bullet y = x \bullet 1 + x \bullet y$

$= x \bullet (1+y)$

$= x \bullet 1$

$= x$

QED (2 true by duality)

Consensus Theorem

1. $xy + x'z + yz = xy + x'z$
2. $(x+y) \bullet (x'+z) \bullet (y+z) = (x+y) \bullet (x'+z)$ -- (dual)

- **Proof:**

$xy + x'z + yz = xy + x'z + (x+x')yz$

$= xy + x'z + xyz + x'yz$

$= (xy + xyz) + (x'z + x'zy)$

$= xy + x'z$

QED (2 true by duality).

Truth Tables (revisited)

- Enumerates all possible combinations of variable values and the corresponding function value
- Truth tables for some arbitrary functions $F_1(x,y,z)$, $F_2(x,y,z)$, and $F_3(x,y,z)$ are shown to the right.

x	y	z		F_1	F_2	F_3
0	0	0		0	1	1
0	0	1		0	0	1
0	1	0		0	0	1
0	1	1		0	1	1
1	0	0		0	1	0
1	0	1		0	1	0
1	1	0		0	0	0
1	1	1		1	0	1

Truth Tables (cont.)

- Truth table: a <u>unique</u> representation of a Boolean function
- If two functions have identical truth tables, the functions are equivalent (and vice-versa).
- Truth tables can be used to prove equality theorems.
- However, the size of a truth table grows <u>exponentially</u> with the number of variables involved, hence unwieldy. This motivates the use of Boolean Algebra.

Boolean expressions-NOT unique

- Unlike truth tables, expressions representing a Boolean function are NOT unique.
- Example:
  - F(x,y,z) = x'•y'•z' + x'•y•z' + x•y•z'
  - G(x,y,z) = x'•y'•z' + y•z'
- The corresponding truth tables for F() and G() are to the right. They are identical!
- Thus, F() = G()

x	y	z		F	G
0	0	0		1	1
0	0	1		0	0
0	1	0		1	1
0	1	1		0	0
1	0	0		0	0
1	0	1		0	0
1	1	0		1	1
1	1	1		0	0

Algebraic Manipulation

- Boolean algebra is a useful tool for simplifying digital circuits.
- Why do it? Simpler can mean cheaper, smaller, faster.
- Example: Simplify F = x'yz + x'yz' + xz.
  F = x'yz + x'yz' + xz
  = x'y(z+z') + xz
  = x'y•1 + xz
  = x'y + xz

Algebraic Manipulation (cont.)

- Example: Prove
  x'y'z' + x'yz' + xyz' = x'z' + yz'
- **Proof:**
  x'y'z'+ x'yz'+ xyz'
  = x'y'z' + x'yz' + x'yz' + xyz'
  = x'z'(y'+y) + yz'(x'+x)
  = x'z'•1 + yz'•1
  = x'z' + yz'
  QED.

Complement of a Function

- The complement of a function is derived by interchanging (• and +), and (1 and 0), and complementing each variable.
- Otherwise, interchange 1s to 0s in the truth table column showing F.
- The *complement* of a function IS NOT THE SAME as the *dual* of a function.

Complementation: Example

- Find G(x,y,z), the complement of F(x,y,z) = xy'z' + x'yz
- G = F' = (xy'z' + x'yz)'
  = (xy'z')' • (x'yz)' *DeMorgan*
  = (x'+y+z) • (x+y'+z') *DeMorgan* again
- Note: The complement of a function can also be derived by finding the function's *dual,* and then complementing all of the literals

Canonical and Standard Forms

- We need to consider formal techniques for the simplification of Boolean functions.
  - Minterms and Maxterms
  - Sum-of-Minterms and Product-of-Maxterms
  - Product and Sum terms
  - Sum-of-Products (SOP) and Product-of-Sums (POS)

Definitions

- *Literal:* A variable or its complement
- *Product term:* literals connected by •
- *Sum term:* literals connected by +
- *Minterm:* a product term in which all the variables appear exactly once, either complemented or uncomplemented
- *Maxterm:* a sum term in which all the variables appear exactly once, either complemented or uncomplemented

Minterm

- Represents exactly one combination in the truth table.
- Denoted by m_j, where j is the decimal equivalent of the minterm's corresponding binary combination *(bj).*
- A variable in *mj* is complemented if its value in *bj* is 0, otherwise is uncomplemented.
- Example: Assume 3 variables (A,B,C), and j=3. Then, b_j = 011 and its corresponding minterm is denoted by m_j = A'BC

Maxterm

- Represents exactly one combination in the truth table.
- Denoted by *Mj*, where *j* is the decimal equivalent of the maxterm's corresponding binary combination *(bj).*
- A variable in *Mj* is complemented if its value in b_j is 1, otherwise is uncomplemented.
- Example: Assume 3 variables (A,B,C), and *j*=3. Then, b_j = 011 and its corresponding maxterm is denoted by M_j = A+B'+C'

Truth Table notation for Minterms and Maxterms

- Minterms and Maxterms are easy to denote using a truth table.
- Example: Assume 3 variables x,y,z (order is fixed)

x	y	z		Minterm	Maxterm
0	0	0		x'y'z' = m_0	x+y+z = M_0
0	0	1		x'y'z = m_1	x+y+z' = M_1
0	1	0		x'yz' = m_2	x+y'+z = M_2
0	1	1		x'yz = m_3	x+y'+z'= M_3
1	0	0		xy'z' = m_4	x'+y+z = M_4
1	0	1		xy'z = m_5	x'+y+z' = M_5
1	1	0		xyz' = m_6	x'+y'+z = M_6
1	1	1		xyz = m_7	x'+y'+z' = M_7

Canonical Forms (Unique)

- Any Boolean function F( ) can be expressed as a *unique* **sum** of **min**terms and a unique **product** of **max**terms (under a fixed variable ordering).
- In other words, every function F() has two canonical forms:
  - Canonical Sum-Of-Products (sum of minterms)
  - Canonical Product-Of-Sums (product of maxterms)

Canonical Forms (cont.)

- Canonical Sum-Of-Products:
  The minterms included are those m_j such that F( ) = 1 in row *j* of the truth table for F( ).
- Canonical Product-Of-Sums:
  The maxterms included are those M_j such that F( ) = 0 in row *j* of the truth table for F( ).

Example

- Truth table for $f_1(a,b,c)$ at right
- The canonical sum-of-products form for f_1 is
  $f_1(a,b,c) = m_1 + m_2 + m_4 + m_6$
  $= a'b'c + a'bc' + ab'c' + abc'$
- The canonical product-of-sums form for f_1 is
  $f_1(a,b,c) = M_0 \bullet M_3 \bullet M_5 \bullet M_7$
  $= (a+b+c)\bullet(a+b'+c')\bullet(a'+b+c')\bullet(a'+b'+c')$.
- Observe that: $m_j = M_j'$

a	b	c		f_1
0	0	0		0
0	0	1		1
0	1	0		1
0	1	1		0
1	0	0		1
1	0	1		0
1	1	0		1
1	1	1		0

Shorthand: ? and ?

- $f_1(a,b,c) =$? m(1,2,4,6), where ? indicates that this is a sum-of-products form, and m(1,2,4,6) indicates that the minterms to be included are m_1, m_2, m_4, and m_6.
- $f_1(a,b,c) =$? M(0,3,5,7), where ? indicates that this is a product-of-sums form, and M(0,3,5,7) indicates that the maxterms to be included are M_0, M_3, M_5, and M_7.
- Since $m_j = M_j'$ for any *j*,
  ? m(1,2,4,6) = ? M(0,3,5,7) = $f_1(a,b,c)$

Conversion Between Canonical Forms

- Replace ? with ? (or *vice versa*) and replace those *j*'s that appeared in the original form with those that do not.
- Example:
  f_1(a,b,c) = a'b'c + a'bc' + ab'c' + abc'
  = m_1 + m_2 + m_4 + m_6
  = ? (1,2,4,6)
  = ? (0,3,5,7)
  = (a+b+c)•(a+b'+c')•(a'+b+c')•(a'+b'+c')

Standard Forms (NOT Unique)

- Standard forms are "*like*" canonical forms, except that not all variables need appear in the individual product (SOP) or sum (POS) terms.
- Example:
  f_1(a,b,c) = a'b'c + bc' + ac'
  is a *standard* sum-of-products form
- f_1(a,b,c) = (a+b+c)•(b'+c')•(a'+c')
  is a *standard* product-of-sums form.

Conversion of SOP from standard to canonical form

- Expand *non-canonical* terms by inserting equivalent of 1 in each missing variable x: $(x + x') = 1$
- Remove duplicate minterms
- $f_1(a,b,c) = a'b'c + bc' + ac'$
  $= a'b'c + (a+a')bc' + a(b+b')c'$
  $= a'b'c + abc' + a'bc' + abc' + ab'c'$
  $= a'b'c + abc' + a'bc + ab'c'$

Conversion of POS from standard to canonical form

- Expand noncanonical terms by adding 0 in terms of missing variables (*e.g.*, $xx' = 0$) and using the distributive law
- Remove duplicate maxterms
- $f_1(a,b,c) = (a+b+c)\bullet(b'+c')\bullet(a'+c')$
  $= (a+b+c)\bullet(aa'+b'+c')\bullet(a'+bb'+c')$
  $= (a+b+c)\bullet(a+b'+c')\bullet(a'+b'+c')\bullet$
  $(a'+b+c')\bullet(a'+b'+c')$
  $= (a+b+c)\bullet(a+b'+c')\bullet(a'+b'+c')\bullet(a'+b+c')$

TUGAS – 1 (11 Maret 2009)

Sederhanakan fungsi boole berikut

a. ((A + B + C) D)’
b. (ABC + DEF)’
c. (AB’ + C’D + EF)’
d. ((A + B)’ + C’)’
e. ((A’ + B) + CD)’
f. ((A + B)C’D’ + E + F’)’
g. AB + A(B + C) + B(B + C)
h. [AB’(C + BD) + A’B’]C
i. A’BC + AB’C’ + A’B’C’ + AB’C + ABC
j. (AB + AC)’ + A’B’C