

Midterm Review for CSE 140

Chester Holtz

Based on SP19 & WI16 slides

April 23, 2021

Logistics

- ▶ Released on course website:
`http://cseweb.ucsd.edu/classes/sp21/cse140-a/`
- ▶ Full 48 hours, submission on gradescope
- ▶ Released Sunday 4/25 10:00 am PST, due Thursday 4/27 10:00 am PST
- ▶ 2 sections:
 - ▶ Section A: 90% question-solving style, roughly 80 minutes
 - ▶ Section B: 10% exploratory style, no penalties

Overview

- ▶ Boolean algebra
- ▶ Consensus theorem
- ▶ Shannon expansion
- ▶ Canonical SOP & POS, minterms, maxterms
- ▶ Minimizing two level SOP/POS using Kmaps
- ▶ Universal logic

Boolean algebra

Boolean Algebra is a multiple valued logic $\mathcal{A} = \{a_0, a_1, a_2, \dots a_n\}$, defined over two operations $\mathcal{O} = \{\mathbf{o_1}, \mathbf{o_2}\}$, that satisfies the following properties:

- ▶ It is closed with respect to each of the operations
- ▶ Each operation has an identity element:

Example (switching logic):

- ▶ $1 \text{ AND } X = X \rightarrow \text{"1" is the identity element of the AND operation}$
- ▶ $0 \text{ OR } X = X \rightarrow \text{"0" is the identity element of the OR operation.}$

Complements

Each element a has a unique complement element a' :

- ▶ $a\mathbf{o_1}a' = \text{Identity element of } \mathbf{o_1}$
- ▶ $a\mathbf{o_2}a' = \text{Identity element of } \mathbf{o_2}$
 - ▶ for each M from $\{a_0, a_1, a_2, \dots a_n\}$ we can find a unique N from $\{a_0, a_1, a_2, \dots a_n\}$ that satisfies the above equalities and we call N the complement of m (and visa-versa)
 - ▶ We cannot have Boolean algebra that has odd number of values!

Example:

- ▶ $0 \text{ OR } 1 = 1 \rightarrow \text{Identity element of } \mathbf{AND} \text{ operation}$
- ▶ $0 \text{ AND } 1 = 0 \rightarrow \text{Identity element of } \mathbf{OR} \text{ operation}$

Axioms

- ▶ **Commutativity:** For any x, y in $\mathcal{A}$ and $\mathbf{o}$ in $\mathcal{O}$, $x\mathbf{o}y = y\mathbf{o}x$
 - ▶ x **AND** $y = y$ **AND** x
 - ▶ x **OR** $y = y$ **OR** x
- ▶ **Distributivity:** For any x, y, z in $\mathcal{A}$,
 $x\mathbf{o}_1(y\mathbf{o}_2z) = (x\mathbf{o}_1y)\mathbf{o}_2(x\mathbf{o}_1z)$
 - ▶ x **AND** $(y$ **OR** $z) = (x$ **AND** $y)$ **OR** $(x$ **AND** $z)$
 - ▶ x **OR** $(y$ **AND** $z) = (x$ **OR** $y)$ **AND** $(x$ **OR** $z)$
- ▶ **De Morgan's laws** (not an axiom):
 - ▶ $(x$ **OR** $y)' = x'$ **AND** y'
 - ▶ $(x$ **AND** $y)' = x'$ **OR** y'

Duality principle

Theorem

For every postulate (e.g. distributive law) in a Boolean algebra system, if we transform the equation by swapping the identity elements 0 and 1 and exchanging the **OR** and **AND** operators, the equality remains to be true

Duality principle

For every postulate (e.g. distributive law) in a Boolean algebra system, if we transform the equation by swapping the identity elements 0 and 1 and exchanging the **OR** and **AND** operators, the equality remains to be true

Question: Does this apply to arbitrary logical expressions?

Consensus theorem

$$\mathbf{A}\underline{B} + \mathbf{A}'\underline{C} + \underline{BC} = AB + A'C \text{ [} BC \text{ is called the consensus term of } AB \text{ and } A'C]$$

Consensus theorem

$$\underline{A}\underline{B} + \underline{A}'\underline{C} + \underline{B}\underline{C} = AB + A'C \text{ [} \underline{BC} \text{ is called the consensus term of } AB \text{ and } A'C \text{]}$$

In general form: $AN + A'M + MN = AN + A'M$ where M and N are boolean expressions.

Consensus theorem

$\underline{A}B + A'\underline{C} + \underline{BC} = AB + A'C$ [BC is called the consensus term of AB and $A'C$]

In general form: $AN + A'M + MN = AN + A'M$ where M and N are boolean expressions.

Examples

- ▶ $abcd + a'ce + bcde$
- ▶ $abc + a'(c + d) + bc + bcd$

Consensus theorem

$\underline{AB} + \underline{A'C} + \underline{BC} = AB + A'C$ [$\underline{BC}$ is called the consensus term of $\underline{AB}$ and $\underline{A'C}$]

In general form: $AN + A'M + MN = AN + A'M$ where M and N are boolean expressions.

Examples

- ▶ $\underline{abcd} + \underline{a'ce} + bcde$
- ▶ $\underline{abc} + \underline{a'(c+d)} + bc + bcd$

Consensus theorem

Dual form

$$(\mathbf{A} + \underline{B})(\mathbf{A}' + \underline{C})(B + C) = (A + B)(A' + C)$$

In general form: $(A + N)(A' + M)(M + N) = (A + N)(A' + M)$
where M and N are boolean expressions.

Consensus theorem

Dual form

$$(\mathbf{A} + \underline{B})(\mathbf{A}' + \underline{C})(B + C) = (A + B)(A' + C)$$

In general form: $(A + N)(A' + M)(M + N) = (A + N)(A' + M)$
where M and N are boolean expressions.

Examples

$$\blacktriangleright (a + b + c + d) (a' + c + e) (b + c + d + e)$$

Consensus theorem

Dual form

$$(\mathbf{A} + \underline{B})(\mathbf{A}' + \underline{C})(B + C) = (A + B)(A' + C)$$

In general form: $(A + N)(A' + M)(M + N) = (A + N)(A' + M)$
where M and N are boolean expressions.

Examples

► $(a + \underline{b + c + d})(a' + \underline{c + e})(b + c + d + e)$

Shannon expansion

Definition

$$\begin{aligned}f(x, y, z) &= x.f(1, y, z) + x'.f(0, y, z) \\&= y.f(x, 1, z) + y'.f(x, 0, z) \\&= z.f(x, y, 1) + z'.f(x, y, 0)\end{aligned}$$

Example

$$f(x, y, z) = xyz + x'y + z'y$$

Which variable should we expand on?

Shannon expansion

Definition

$$f(x, y, z) = x.f(1, y, z) + x'.f(0, y, z)$$

Example—try x

$$\begin{aligned}f(x, y, z) &= xyz + x'y + z'y \\&= x.f(1, y, z) + x'.f(0, y, z) \\&= x.(yz + 0 + z'y) + x'.(0 + y + z'y) \\&= x.(y(z + z')) + x'.(y(1 + z')) \\&= x.y + x'.y \\&= y.(x + x') = y\end{aligned}$$

Shannon expansion

Definition

$$f(x, y, z) = x.f(1, y, z) + x'.f(0, y, z)$$

Example—try x

$$\begin{aligned}f(x, y, z) &= xyz + x'y + z'y \\&= x.f(1, y, z) + x'.f(0, y, z) \\&= x.(yz + 0 + z'y) + x'.(0 + y + z'y) \\&= x.(y(z + z')) + x'.(y(1 + z')) \\&= x.y + x'.y \\&= y.(x + x') = y\end{aligned}$$

Shannon expansion

Definition

$$f(x, y, z) = x.f(1, y, z) + x'.f(0, y, z)$$

Example—try x

$$\begin{aligned}f(x, y, z) &= xyz + x'y + z'y \\&= x.f(1, y, z) + x'.f(0, y, z) \\&= x.(yz + 0 + z'y) + x'.(0 + y + z'y) \\&= x.(y(z + z')) + x'.(y(1 + z')) \\&= x.y + x'.y \\&= y.(x + x') = y\end{aligned}$$

Shannon expansion

Definition

$$f(x, y, z) = x.f(1, y, z) + x'.f(0, y, z)$$

Example—try x

$$\begin{aligned} f(x, y, z) &= xyz + x'y + z'y \\ &= x.f(1, y, z) + x'.f(0, y, z) \\ &= x.(yz + 0 + z'y) + x'.(0 + y + z'y) \\ &= x.(y(z + z')) + x'.(y(1 + z')) \\ &= x.y + x'.y \\ &= y.(x + x') = y \end{aligned}$$

Shannon expansion

Definition (dual of prev.)

$$\begin{aligned}f(x, y, z) &= (x + f(0, y, z)).(x'.f(1, y, z)) \\&= (y + f(x, 0, z)).(y'.f(x, 1, z)) \\&= (z + f(x, y, 0)).(z'.f(x, y, 1))\end{aligned}$$

Example

$$f(x, y, z) = (x + y + z').(x' + y' + z).(z + y)$$

Which variable should we expand on?

Shannon expansion

Definition

$$f(x, y, z) = (x + f(0, y, z)).(x'.f(1, y, z))$$

Example—try x

$$\begin{aligned}f(x, y, z) &= (x + y + z').(x' + y' + z).(z + y) \\&= (x + f(0, y, z)).(x'.f(1, y, z)) \\&= (x + (y + z')(1)(z + y)).(x' + (1)(y' + z)(z + y)) \\&= (x + y + z'z).(x' + z + y'y) \\&= (x + y + 0).(x' + z + 0) \\&= (x + y)(x' + z)\end{aligned}$$

Shannon expansion

Definition

$$f(x, y, z) = (x + f(0, y, z)) \cdot (x' \cdot f(1, y, z))$$

Example

$$f(x, y) = (x' + y) \oplus (x + y') \oplus xy \oplus x'y' \oplus x'$$

Shannon expansion

Definition

$$f(x, y, z) = (x + f(0, y, z)) \cdot (x' \cdot f(1, y, z))$$

Example—try x

$$\begin{aligned} f(x, y) &= (x' + y) \oplus (x + y') \oplus xy \oplus x'y' \oplus x' \\ &= xf(1, y) + x'f(0, y) \\ &= x((0 + y) \oplus (1 + y') \oplus 1y \oplus 0y' \oplus 0) + \\ &\quad x'((1 + y) \oplus (0 + y') \oplus 0y \oplus 1y' \oplus 1) \\ &= x(y \oplus 1 \oplus y \oplus 0 \oplus 0) + x'(1 \oplus y' \oplus 0 \oplus y' \oplus 1) \\ &= x \end{aligned}$$

Shannon expansion

Definition

$$f(x, y, z) = (x + f(0, y, z)) \cdot (x' \cdot f(1, y, z))$$

Example—try x

$$\begin{aligned} f(x, y) &= (x' + y) \oplus (x + y') \oplus xy \oplus x'y' \oplus x' \\ &= xf(1, y) + x'f(0, y) \\ &= x((0 + y) \oplus (1 + y') \oplus 1y \oplus 0y' \oplus 0) + \\ &\quad x'((1 + y) \oplus (0 + y') \oplus 0y \oplus 1y' \oplus 1) \\ &= x(y \oplus 1 \oplus y \oplus 0 \oplus 0) + x'(1 \oplus y' \oplus 0 \oplus y' \oplus 1) \\ &= x \end{aligned}$$

Definitions

- ▶ **Literal:** a variable or its complement: x, x'
- ▶ **Product term:** **AND** of some literals: $(xy'z), (xy)$
- ▶ **Sum term:** **OR** of some literals: $(x + y' + z), (y + z)$

Minterms and maxterms

Definitions

- ▶ **Minterm:** A product term with all the variables specified in either direct form or the complement form.
- ▶ **Maxterm:** A sum term that has all the variables in either the direct form or the complement form.

Note: Each minterm is equal to value “one” only for a unique input combination and is “zero” otherwise. Each maxterm is equal to value “zero” only for a unique input combination and is “one” otherwise.

Question: For a system with n variables, how many possible min/max-terms?

Canonical form

We can represent any Boolean function two ways:

canonical sum of products: Sum of its minterms (i.e ORing all places where the function is one)

canonical product of sums: Product of its maxterms (i.e ANDing all places where the function is zero)

Canonical form (example)

a	b	c	F
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Canonical SOP

Canonical POS

Canonical form (example)

<i>a</i>	<i>b</i>	<i>c</i>	<i>F</i>
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

Canonical SOP

$$F(a, b, c) =$$

$$a'b'c' + a'bc + ab'c + abc'$$

Canonical POS

$$F(a, b, c) = (a + b + c')(a + b' + c)(a' + b + c)(a' + b' + c')$$

Logic minimization

Canonical forms are inefficient. Want to find ways to merge two or more minterms (maxterms) together in order to get a smaller product (sum) term.

Adjacency: two minterms (maxterms) are adjacent, if they differ only in one variable: $abcd$ and $ab'cd$ are adjacent. $(a + b + c)$ and $(a + b + c')$ are adjacent.

Adjacent pairs can be merged into single terms. This is the basis for many logic minimization algorithms.

Logic minimization

Adjacency: two minterms (maxterms) are adjacent, if they differ only in one variable: $abcd$ and $ab'cd$ are adjacent. $(a + b + c)$ and $(a + b + c')$ are adjacent.

Adjacent pairs can be merged into single terms.

Example

- ▶ $abcd + ab'cd$
- ▶ $(a + b + c)(a + b + c')$

Logic minimization

Adjacency: two minterms (maxterms) are adjacent, if they differ only in one variable: $abcd$ and $ab'cd$ are adjacent. $(a + b + c)$ and $(a + b + c')$ are adjacent.

Adjacent pairs can be merged into single terms.

Example

► $abcd + ab'cd = acd(b + b') = acd$

► $(a + b + c)(a + b + c') = (a + b) + (cc') = a + b$

K-maps

All neighbouring cells in a Kmaps are adjacent.

Definitions

- ▶ **on-set**: set of all the “ones” in the function
- ▶ **off-set**: set of all the “zeros” in the function
- ▶ **Implicant**: A product term that has non-empty intersection with on-set and does not intersect with the off-set. It can also contain “dont cares”
 - ▶ How many minterms?

Definitions

- ▶ **on-set:** set of all the “ones” in the function
- ▶ **off-set:** set of all the “zeros” in the function
- ▶ **Implicant:** A product term that has non-empty intersection with on-set and does not intersect with the off-set. It can also contain “dont cares”
 - ▶ How many minterms?
 - ▶ Being a product term means that it should include 1 minterm, or 2 minterms, or 4, ...(powers of two)... . It CANNOT have 3 minterms!

Definitions

- ▶ **Prime implicant:** Is an Implicant which is not a proper subset of any other Implicant.
 - ▶ Cannot find any other implicant that completely covers a prime implicant. For logic minimization we use Prime Implicants.
- ▶ **Essential prime implicant:** A Prime Implicant that has at least one element from the on-set that cannot be covered by any other Prime Implicant.

Minimal SOP

1. Draw the kmap representing the given function
2. Find all the Prime Implicants using the on-set.
3. Find all the Essential Prime Implicants.
4. Write all the Essential Prime Implicants in the final cover.
5. Search through the remaining non-essential prime implicants that will cover the remaining ones in the map.

Minimal SOP

1. Draw the kmap representing the given function
2. Find all the Prime Implicants using the on-set.
3. Find all the Essential Prime Implicants.
4. Write all the Essential Prime Implicants in the final cover.
5. Search through the remaining non-essential prime implicants that will cover the remaining ones in the map.

Example

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	1	0	0	1
	01	1	1	1	1
	11	0	1	x	0
	10	0	1	0	0

Prime implicants

Essential Prime implicants

Minimal SOP

Minimal SOP

1. Draw the kmap representing the given function
2. Find all the Prime Implicants using the on-set.
3. Find all the Essential Prime Implicants.
4. Write all the Essential Prime Implicants in the final cover.
5. Search through the remaining non-essential prime implicants that will cover the remaining ones in the map.

Example

		<i>CD</i>			
		00	01	11	10
<i>AB</i>	00	1	0	0	1
	01	1	1	1	1
	11	0	1	x	0
	10	0	1	0	0

Prime implicants

► $\sum m(0, 4)$?

► $\sum m(5, 9, 13)$?

Essential Prime implicants

Minimal SOP

Minimal SOP

1. Draw the kmap representing the given function
2. Find all the Prime Implicants using the on-set.
3. Find all the Essential Prime Implicants.
4. Write all the Essential Prime Implicants in the final cover.
5. Search through the remaining non-essential prime implicants that will cover the remaining ones in the map.

Example

		CD			
		00	01	11	10
AB	00	1	0	0	1
	01	1	1	1	1
	11	0	1	x	0
	10	0	1	0	0

Prime implicants

- ▶ $\sum m(0, 2, 4, 6)$
- ▶ $\sum m(4, 5, 6, 7)$
- ▶ $\sum m(5, 7, 13, 15)$
- ▶ $\sum m(9, 13)$

Essential Prime implicants

- ▶ $\sum m(0, 2, 4, 6)$
- ▶ $\sum m(9, 13)$

Minimal SOP

- ▶ $F(a, b, c, d) =$
 $a'd' + ac'd + a'b$

Minimal POS

1. Draw the kmap representing the given function
2. Find all the Prime Implicants using the on-set.
3. Find all the Essential Prime Implicants.
4. Write all the Essential Prime Implicants in the final cover.
5. Search through the remaining non-essential prime implicants that will cover the remaining zeros in the map.

(Briefly) Universal logic

1. { **AND** , **NOT** } is a universal set of gates.
2. { $\oplus$, **NOT** } is a universal set of gates.
3. { $f(a, b, c) = a'b' + b'c' + a'c'$ } is a universal set of gates.

(Briefly) Universal logic

1. { **AND** , **NOT** } is a universal set of gates: True
(DeMorgan's)
2. { $\oplus$, **NOT** } is a universal set of gates: False (**AND**)
3. { $f(a, b, c) = a'b' + b'c' + a'c'$ } is a universal set of gates:
True

Some more tips

- ▶ Part A (problem-solving)
 - ▶ Similar to previous exams
 - ▶ 5 sub-sections
 - ▶ 1 section True/false
 - ▶ 4 sections on course topics
- ▶ Part B (exploratory)
 - ▶ Circuit design (e.g. draw a logic diagram, create a truth table, etc)
 - ▶ Boolean algebra
 - ▶ Universal logic

Good Luck!