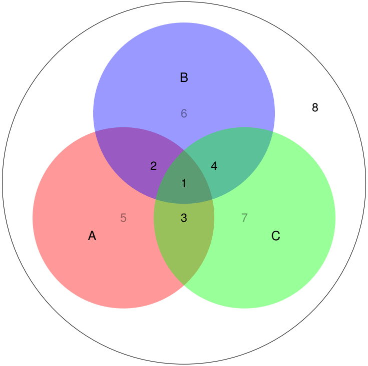
Contents

Section outline

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$

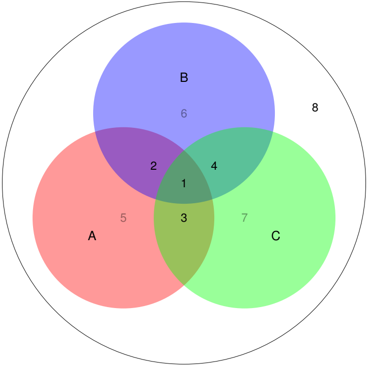


Selections

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$



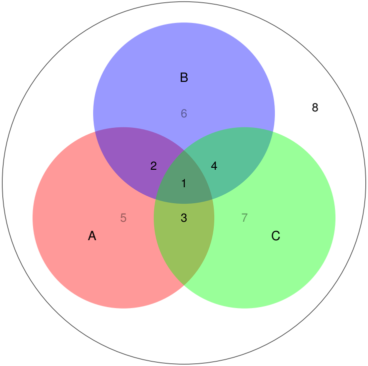
Selections

1, 2: $A \cap B$

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$



Selections

1, 2: $A \cap B$

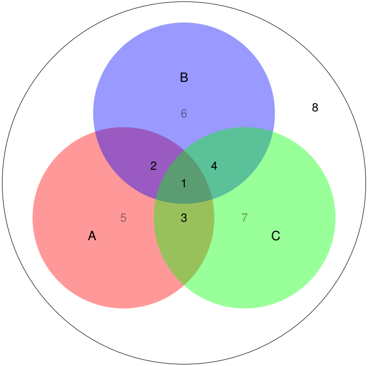
$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C})$
$abc + ab\overline{c} = ab$

1, 2, 3, 5: A

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$



Selections

1, 2: $A \cap B$

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C})$
$abc + ab\overline{c} = ab$

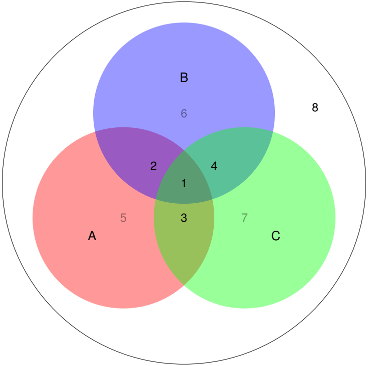
1, 2, 3, 5: A

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C}) \cup$
$(A \cap \overline{B} \cap C) \cup$
$(A \cap \overline{B} \cap \overline{C})$
$abc + ab\overline{c} + a\overline{b}c +$
$a\overline{b}\overline{c}$

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$



Selections

1, 2: $A \cap B$

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C})$
$abc + ab\overline{c} = ab$

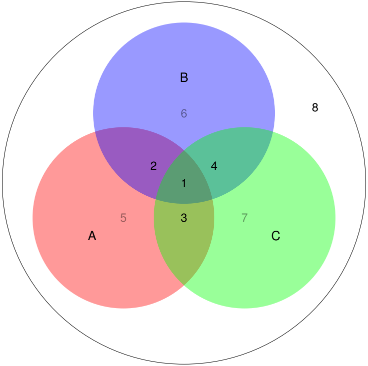
1, 2, 3, 5: A

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C}) \cup$
$(A \cap \overline{B} \cap C) \cup$
$(A \cap \overline{B} \cap \overline{C})$
$abc + ab\overline{c} + a\overline{b}c +$
$a\overline{b}\overline{c} = ab + a\overline{b} = a$

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$



a I have an item from A

$\bar{a}$ I don't have an item from A

Selections

1, 2: $A \cap B$

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C})$
$abc + ab\bar{c} = ab$

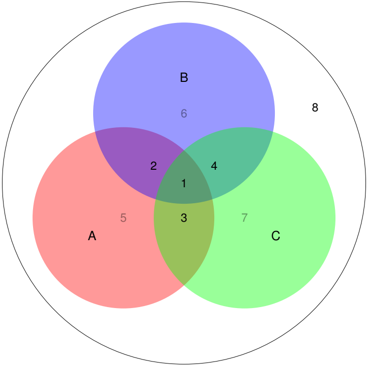
1, 2, 3, 5: A

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C}) \cup$
$(A \cap \overline{B} \cap C) \cup$
$(A \cap \overline{B} \cap \overline{C})$
$abc + ab\bar{c} + a\bar{b}c +$
$a\bar{b}\bar{c} = ab + a\bar{b} = a$

Sum of products from sets

Regions

1. $A \cap B \cap C$
2. $A \cap B \cap \overline{C}$
3. $A \cap \overline{B} \cap C$
4. $\overline{A} \cap B \cap C$
5. $A \cap \overline{B} \cap \overline{C}$
6. $\overline{A} \cap B \cap \overline{C}$
7. $\overline{A} \cap \overline{B} \cap C$
8. $\overline{A} \cap \overline{B} \cap \overline{C}$



Selections

1, 2: $A \cap B$

$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C})$
$abc + ab\overline{c} = ab$

1, 2, 3, 5: A

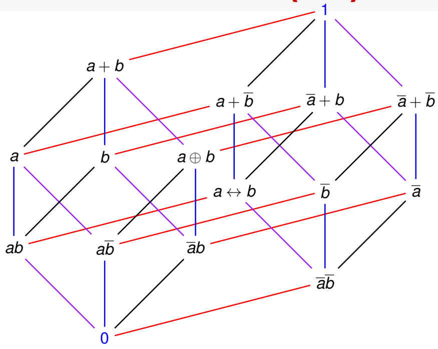
$(A \cap B \cap C) \cup$
$(A \cap B \cap \overline{C}) \cup$
$(A \cap \overline{B} \cap C) \cup$
$(A \cap \overline{B} \cap \overline{C})$
$abc + ab\overline{c} + a\overline{b}c +$
$a\overline{b}\overline{c} = ab + a\overline{b} = a$

a I have an item from A

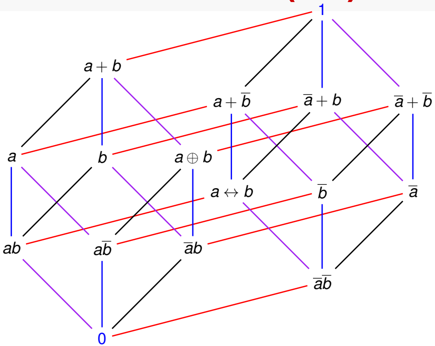
$\overline{a}$ I don't have an item from A

$a\overline{b} + c$ I have an item from A but not from B or an item from C

Boolean lattice (BL) for 2 variables

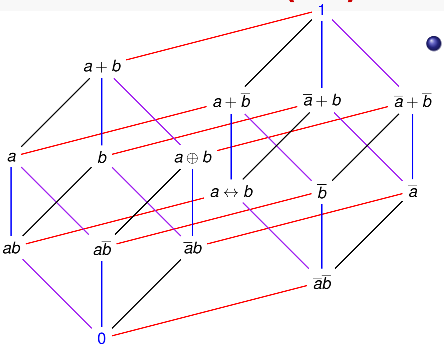


Boolean lattice (BL) for 2 variables



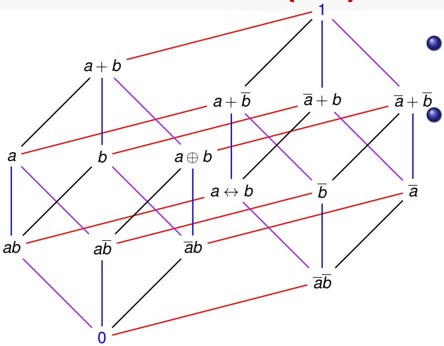
- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted

Boolean lattice (BL) for 2 variables



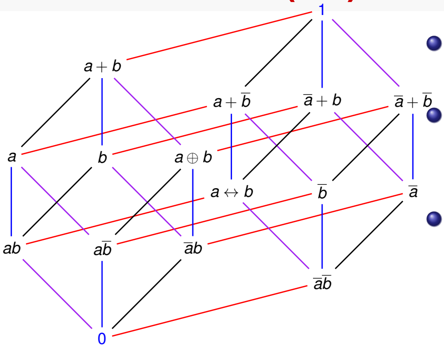
- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted
- Such an expression is *well formed* or syntactically correct

Boolean lattice (BL) for 2 variables



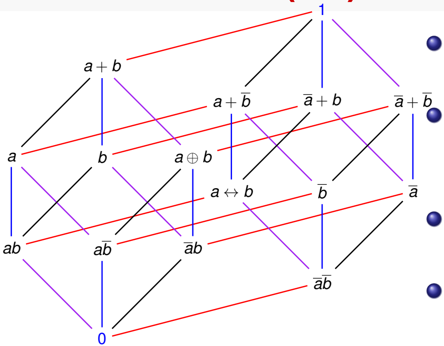
- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted
- Such an expression is *well formed* or syntactically correct
- A fundamental product (FP) is a literal or a product of two or more literals arising from distinct variables

Boolean lattice (BL) for 2 variables



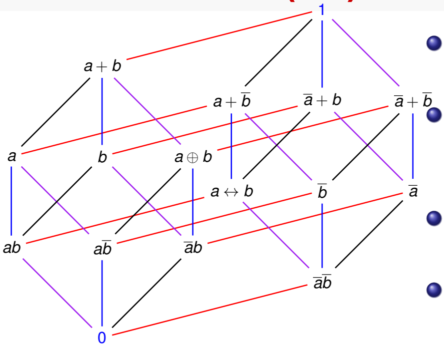
- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted
- Such an expression is *well formed* or syntactically correct
- A fundamental product (FP) is a literal or a product of two or more literals arising from distinct variables
- A FP involving all the variables is a *minterm* – atoms in the BL

Boolean lattice (BL) for 2 variables



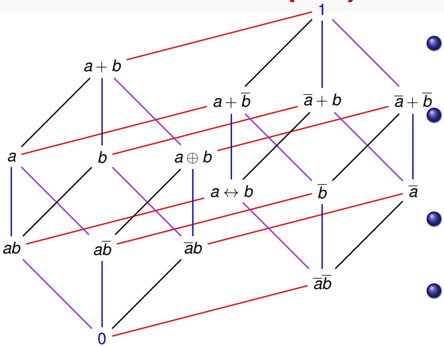
- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted
- Such an expression is *well formed* or syntactically correct
- A fundamental product (FP) is a literal or a product of two or more literals arising from distinct variables
- A FP involving all the variables is a *minterm* – atoms in the BL
- A FP P_1 is contained or included in P_2 if P_2 has all the literals of P_1; then $P_2 \Rightarrow P_1$ (P_2 implies P_1)

Boolean lattice (BL) for 2 variables



- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted
- Such an expression is *well formed* or syntactically correct
- A fundamental product (FP) is a literal or a product of two or more literals arising from distinct variables
- A FP involving all the variables is a *minterm* – atoms in the BL
- A FP P_1 is contained or included in P_2 if P_2 has all the literals of P_1; then $P_2 \Rightarrow P_1$ (P_2 implies P_1)
- A *sum of products* (SOP) expression is FP or a sum of two or more FPs $P_1, \ldots, P_n$ and $\forall i, j, P_i \not\Rightarrow P_j$

Boolean lattice (BL) for 2 variables



- A *literal* is a variable (a) or its complement ($\overline{a}$)
- A Boolean expression is a string built from literals and the Boolean operators without violating their arity
- Grouping with parentheses is permitted
- Such an expression is *well formed* or syntactically correct
- A fundamental product (FP) is a literal or a product of two or more literals arising from distinct variables
- A FP involving all the variables is a *minterm* – atoms in the BL
- A FP P_1 is contained or included in P_2 if P_2 has all the literals of P_1; then $P_2 \Rightarrow P_1$ (P_2 implies P_1)
- A *sum of products* (SOP) expression is FP or a sum of two or more FPs $P_1, \ldots, P_n$ and $\forall i, j, P_i \not\Rightarrow P_j$
- DeMorgan's laws, distributivity, commutativity, idempotence, involution may be used to transform a Boolean expression to SOP

Functional completeness

- May be derived from the Boolean lattice

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

NAND $\overline{x \cdot y}$

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

NAND $\overline{x \cdot y}$

NOR $\overline{x + y}$

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

NAND $\overline{x \cdot y}$

NOR $\overline{x + y}$

XOR,AND $x \oplus y, x \cdot y$

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

NAND $\overline{x \cdot y}$

NOR $\overline{x + y}$

XOR,AND $x \oplus y, x \cdot y$

MUX $s \cdot x + \overline{s} \cdot y$

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

NAND $\overline{x \cdot y}$

NOR $\overline{x + y}$

XOR,AND $x \oplus y, x \cdot y$

MUX $s \cdot x + \overline{s} \cdot y$

RAM Random access memory

Functional completeness

- May be derived from the Boolean lattice
- OR is required to compute the joins on the elements
- NOT and AND are required to compute the atoms from the proposition variables

x	y	$\overline{x}$	$x \cdot y$	$x + y$
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

NAND $\overline{x \cdot y}$

NOR $\overline{x + y}$

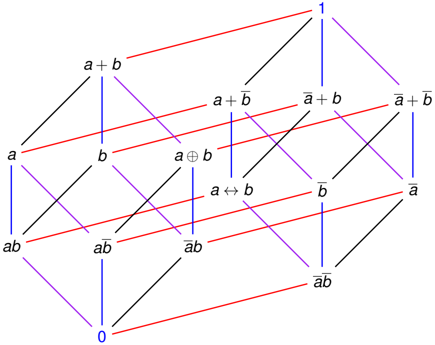
XOR,AND $x \oplus y, x \cdot y$

MUX $s \cdot x + \overline{s} \cdot y$

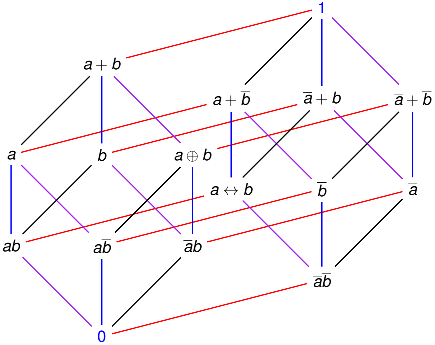
RAM Random access memory

Minority Minority value among given inputs

Boolean expressions

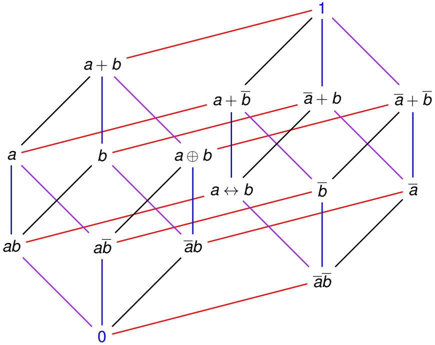


Boolean expressions



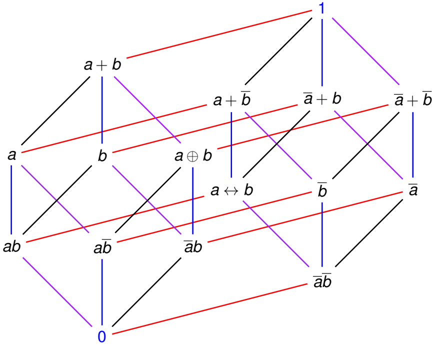
- $E = x\overline{z} + \overline{y}z + xy\overline{z}$

Boolean expressions



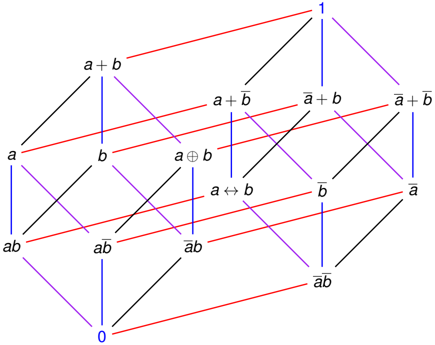
- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

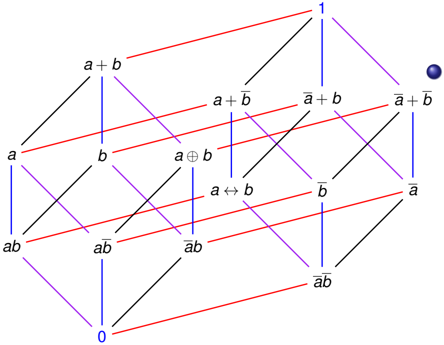
Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)

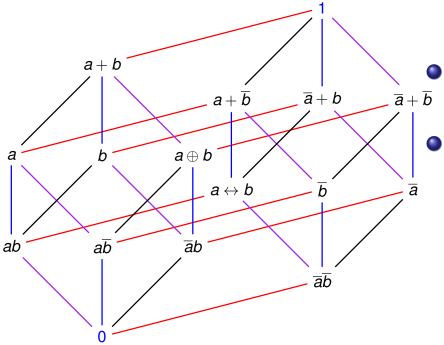
Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)
- The DNF of any SOP is unique (why?) – cannonical SOP

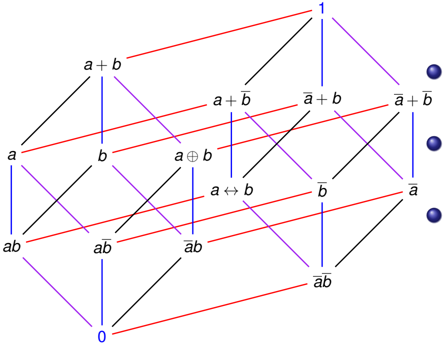
Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)
- The DNF of any SOP is unique (why?) – cannonical SOP
- An element x in a BL is *maxterm* if it has 1 as its only successor

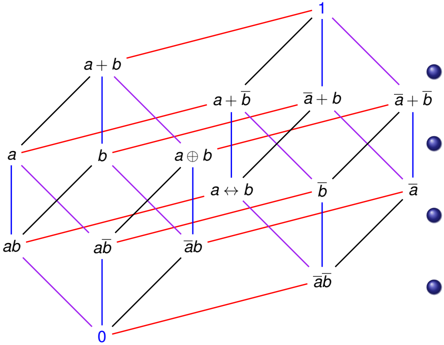
Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)
- The DNF of any SOP is unique (why?) – cannonical SOP
- An element x in a BL is *maxterm* if it has 1 as its only successor
- A maxterm is a sum of literals involving all the variables

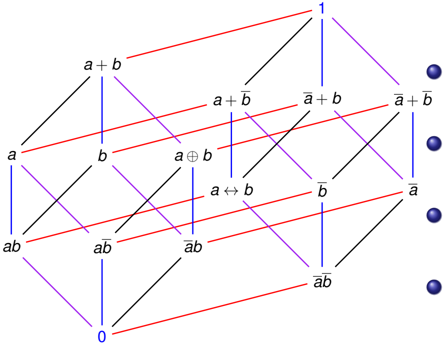
Boolean expressions



- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)
- The DNF of any SOP is unique (why?) – cannonical SOP
- An element x in a BL is *maxterm* if it has 1 as its only successor
- A maxterm is a sum of literals involving all the variables
- Similar to SOP, *product of sums* (POS) may be defined

- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

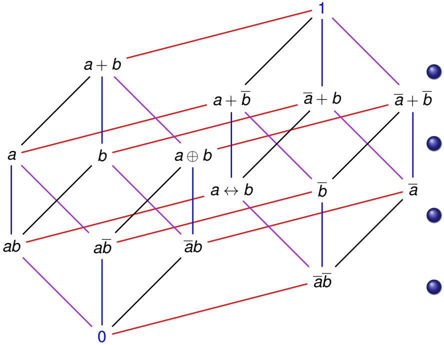
Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)
- The DNF of any SOP is unique (why?) – cannonical SOP
- An element x in a BL is *maxterm* if it has 1 as its only successor
- A maxterm is a sum of literals involving all the variables
- Similar to SOP, *product of sums* (POS) may be defined
- A Boolean expression which is a product of maxterms is said to be in *conjunctive normal form* (CNF)

Boolean expressions



- $E = x\overline{z} + \overline{y}z + xy\overline{z}$
- $E = ?$

- A SOP expression where each FP is a minterm is said to be in *disjunctive normal form* (DNF)
- The DNF of any SOP is unique (why?) – cannonical SOP
- An element x in a BL is *maxterm* if it has 1 as its only successor
- A maxterm is a sum of literals involving all the variables
- Similar to SOP, *product of sums* (POS) may be defined
- A Boolean expression which is a product of maxterms is said to be in *conjunctive normal form* (CNF)
- The CNF of any POS is unique (why?) – cannonical POS

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Alternate argument for minterm expansion

Acceptance for complements: $\bar{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

- An acceptance for minterm expansion on truth assignment of variables happens due to acceptance of exactly one minterm

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

- An acceptance for minterm expansion on truth assignment of variables happens due to acceptance of exactly one minterm
- If m_i and m_j are two distinct minterms on variables $x_1, \ldots, x_k$

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

- An acceptance for minterm expansion on truth assignment of variables happens due to acceptance of exactly one minterm
- If m_i and m_j are two distinct minterms on variables $x_1, \ldots, x_k$
- Let m_i and m_j differ on x_p

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

- An acceptance for minterm expansion on truth assignment of variables happens due to acceptance of exactly one minterm
- If m_i and m_j are two distinct minterms on variables $x_1, \ldots, x_k$
- Let m_i and m_j differ on x_p
- Let x_p occur as literal x_{pi} in m_i and x_{pj} in m_j

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

- An acceptance for minterm expansion on truth assignment of variables happens due to acceptance of exactly one minterm
- If m_i and m_j are two distinct minterms on variables $x_1, \ldots, x_k$
- Let m_i and m_j differ on x_p
- Let x_p occur as literal x_{pi} in m_i and x_{pj} in m_j
- Then $x_{pi} = \overline{x_{pj}}$, so if m_i accepts then m_j doesn't accept and vice versa

Alternate argument for minterm expansion

Acceptance for complements: $\overline{x} = 1$ iff $x = 0$

Acceptance for products: $xy = 1$ iff $x = 1$ and $y = 1$

Acceptance for sums: $u + v = 1$ iff $u = 1$ or $v = 1$

Minterm expansion: sum of distinct minterms

Acceptance for minterm expansion:

- An acceptance for minterm expansion on truth assignment of variables happens due to acceptance of exactly one minterm
- If m_i and m_j are two distinct minterms on variables $x_1, \ldots, x_k$
- Let m_i and m_j differ on x_p
- Let x_p occur as literal x_{pi} in m_i and x_{pj} in m_j
- Then $x_{pi} = \overline{x_{pj}}$, so if m_i accepts then m_j doesn't accept and vice versa
- This ensures that the minterm expansion is unique

Number of Boolean functions

By lattice:

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)
- Thus there are $2^n = 2^{2^k}$ distinct Boolean functions

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)
- Thus there are $2^n = 2^{2^k}$ distinct Boolean functions

By minterm expansion:

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)
- Thus there are $2^n = 2^{2^k}$ distinct Boolean functions

By minterm expansion:

- A Boolean function on k variables has $n = 2^k$ possible minterms

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)
- Thus there are $2^n = 2^{2^k}$ distinct Boolean functions

By minterm expansion:

- A Boolean function on k variables has $n = 2^k$ possible minterms
- A minterm expansion results in a unique acceptance

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)
- Thus there are $2^n = 2^{2^k}$ distinct Boolean functions

By minterm expansion:

- A Boolean function on k variables has $n = 2^k$ possible minterms
- A minterm expansion results in a unique acceptance
- The minterms may be chosen in $\sum_{k=0}^{k=n} \binom{n}{k} = 2^n = 2^{2^k}$ ways

Number of Boolean functions

By lattice:

- A Boolean lattice for a Boolean function of k variables has $n = 2^k$ atoms as minterms
- A Boolean lattice with n atoms has 2^n elements by the Stone representation theorem
- Each non-zero element has a unique representation in terms of the atoms (minterms)
- Thus there are $2^n = 2^{2^k}$ distinct Boolean functions

By minterm expansion:

- A Boolean function on k variables has $n = 2^k$ possible minterms
- A minterm expansion results in a unique acceptance
- The minterms may be chosen in $\sum_{k=0}^{k=n} \binom{n}{k} = 2^n = 2^{2^k}$ ways
- Each choice denotes a distinct Boolean function

Boolean expression manipulation

- $xy + \overline{x}z + yz = xy + \overline{x}z$
- $(x + y)(\overline{x} + z)(y + z) = (x + y)(\overline{x} + z)$
- $T = (x + y)\overline{[\overline{x}(\overline{y} + \overline{z})]} + \overline{x}\ \overline{y} + \overline{x}\ \overline{z}$
- $xy + \overline{x}\ \overline{y} + yz = xy + \overline{x}\ \overline{y} + \overline{x}z$

Exclusive OR

- $a \oplus b = b \oplus a$

Exclusive OR

- $a \oplus b = b \oplus a$
- $(a \oplus b) \oplus c = a \oplus (b \oplus c) = a \oplus b \oplus c$

Exclusive OR

- $a \oplus b = b \oplus a$
- $(a \oplus b) \oplus c = a \oplus (b \oplus c) = a \oplus b \oplus c$
- $a(b \oplus c) = (ab) \oplus (ac)$

Exclusive OR

- $a \oplus b = b \oplus a$
- $(a \oplus b) \oplus c = a \oplus (b \oplus c) = a \oplus b \oplus c$
- $a(b \oplus c) = (ab) \oplus (ac)$
- if $a \oplus b = c$ then $\begin{cases} a \oplus c = b \\ b \oplus c = a \\ a \oplus b \oplus c = 0 \end{cases}$

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)
- MOS transistor, relay, pneumatic valve

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)
- MOS transistor, relay, pneumatic valve
- Normally closed (primed: $\overline{x}$) or normally open (unprimed: x)

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)
- MOS transistor, relay, pneumatic valve
- Normally closed (primed: $\overline{x}$) or normally open (unprimed: x)
- Series connection denoted by AND

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)
- MOS transistor, relay, pneumatic valve
- Normally closed (primed: $\overline{x}$) or normally open (unprimed: x)
- Series connection denoted by AND
- Parallel connection denoted by OR

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)
- MOS transistor, relay, pneumatic valve
- Normally closed (primed: $\overline{x}$) or normally open (unprimed: x)
- Series connection denoted by AND
- Parallel connection denoted by OR
- $T = x\overline{y} + (\overline{x} + y)z = x\overline{y} + \overline{x\overline{y}}z = x\overline{y} + z$

Series-parallel switching circuits

- A transmission device may be treated as a gate (pass or block)
- MOS transistor, relay, pneumatic valve
- Normally closed (primed: $\overline{x}$) or normally open (unprimed: x)
- Series connection denoted by AND
- Parallel connection denoted by OR
- $T = x\overline{y} + (\overline{x} + y)z = x\overline{y} + \overline{x\overline{y}}z = x\overline{y} + z$
- CMOS NAND, NOR

Shannon decomposition

- $f(x_1, x_2, \ldots, x_n) = x_1 \cdot f(1, x_2, \ldots, x_n) + \overline{x_1} \cdot f(0, x_2, \ldots, x_n)$

Shannon decomposition

- $f(x_1, x_2, \ldots, x_n) = x_1 \cdot f(1, x_2, \ldots, x_n) + \overline{x_1} \cdot f(0, x_2, \ldots, x_n)$
- $f(x_1, x_2, \ldots, x_n) = (\overline{x_1} + f(1, x_2, \ldots, x_n)) \cdot (x_1 + f(0, x_2, \ldots, x_n))$

Shannon decomposition

- $f(x_1, x_2, \ldots, x_n) = x_1 \cdot f(1, x_2, \ldots, x_n) + \overline{x_1} \cdot f(0, x_2, \ldots, x_n)$
- $f(x_1, x_2, \ldots, x_n) = (\overline{x_1} + f(1, x_2, \ldots, x_n)) \cdot (x_1 + f(0, x_2, \ldots, x_n))$
- Multiplexer realisation by Shannon decomposition or Shannon expansion

Shannon decomposition

- $f(x_1, x_2, \ldots, x_n) = x_1 \cdot f(1, x_2, \ldots, x_n) + \overline{x_1} \cdot f(0, x_2, \ldots, x_n)$
- $f(x_1, x_2, \ldots, x_n) = (\overline{x_1} + f(1, x_2, \ldots, x_n)) \cdot (x_1 + f(0, x_2, \ldots, x_n))$
- Multiplexer realisation by Shannon decomposition or Shannon expansion
- Repeated application to obtain CNF or DNF of a given Boolean function

Functional completeness

- Treated in Emil Post's functional completeness theorem

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions
  T: T-preserving $f(T, T, \ldots, T) = T$

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions

  T: **T-preserving** $f(T, T, \ldots, T) = T$

  F: **F-preserving** $f(F, F, \ldots, F) = F$

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions

  T: **T-preserving** $f(T, T, \ldots, T) = T$

  F: **F-preserving** $f(F, F, \ldots, F) = F$

  L: **counting** $f(z_1, z_2, \ldots, x_p, \ldots, z_n) \neq f(z'_1, z'_2, \ldots, y_p, \ldots, z'_n)$ if $x_p \neq y_p$ and $z_\imath = z'_\imath$ if position-$\imath$ isn't dummy

  position-$\imath$ is dummy if

  $f(z_1, \ldots, z_\imath, \ldots, z_n) = f(z_1, \ldots, \neg z_\imath, \ldots, z_n)$

  i.e. f is invariant to changes in a dummy position

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions

  T: **T-preserving** $f(T, T, \ldots, T) = T$

  F: **F-preserving** $f(F, F, \ldots, F) = F$

  L: **counting** $f(z_1, z_2, \ldots, x_p, \ldots, z_n) \neq f(z_1', z_2', \ldots, y_p, \ldots, z_n')$ if
  $x_p \neq y_p$ and $z_\imath = z_\imath'$ if position-$\imath$ isn't dummy
  position-$\imath$ is dummy if
  $f(z_1, \ldots, z_\imath, \ldots, z_n) = f(z_1, \ldots, \neg z_\imath, \ldots, z_n)$
  i.e. f is invariant to changes in a dummy position

  M: **monotonic** let $X = \langle x_1, x_2, \ldots, x_n \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$,
  then $X \leq Y \Rightarrow f(X) \leq f(Y)$, where
  $X \leq Y \equiv \forall \imath x_\imath \leq y_\imath$, and $F \leq T$

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions

  T: **T-preserving** $f(T, T, \ldots, T) = T$

  F: **F-preserving** $f(F, F, \ldots, F) = F$

  L: **counting** $f(z_1, z_2, \ldots, x_p, \ldots, z_n) \neq f(z_1', z_2', \ldots, y_p, \ldots, z_n')$ if $x_p \neq y_p$ and $z_\imath = z_\imath'$ if position-$\imath$ isn't dummy
  position-$\imath$ is dummy if
  $f(z_1, \ldots, z_\imath, \ldots, z_n) = f(z_1, \ldots, \neg z_\imath, \ldots, z_n)$
  i.e. f is invariant to changes in a dummy position

  M: **monotonic** let $X = \langle x_1, x_2, \ldots, x_n \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$,
  then $X \leq Y \Rightarrow f(X) \leq f(Y)$, where
  $X \leq Y \equiv \forall \imath x_\imath \leq y_\imath$, and $F \leq T$

  S: **self-dual** $f(x_1, x_2, \ldots, x_n) = \neg f(\neg x_1, \neg x_2, \ldots, \neg x_n)$

Functional completeness

- Treated in Emil Post's functional completeness theorem
- Expressed in terms of five classes of Boolean functions

  T: **T-preserving** $f(T, T, \ldots, T) = T$

  F: **F-preserving** $f(F, F, \ldots, F) = F$

  L: **counting** $f(z_1, z_2, \ldots, x_p, \ldots, z_n) \neq f(z'_1, z'_2, \ldots, y_p, \ldots, z'_n)$ if $x_p \neq y_p$ and $z_\imath = z'_\imath$ if position-$\imath$ isn't dummy
  position-$\imath$ is dummy if
  $f(z_1, \ldots, z_\imath, \ldots, z_n) = f(z_1, \ldots, \neg z_\imath, \ldots, z_n)$
  i.e. f is invariant to changes in a dummy position

  M: **monotonic** let $X = \langle x_1, x_2, \ldots, x_n \rangle$ and $Y = \langle y_1, y_2, \ldots, y_n \rangle$,
  then $X \leq Y \Rightarrow f(X) \leq f(Y)$, where
  $X \leq Y \equiv \forall \imath x_\imath \leq y_\imath$, and $F \leq T$

  S: **self-dual** $f(x_1, x_2, \ldots, x_n) = \neg f(\neg x_1, \neg x_2, \ldots, \neg x_n)$
- A set $\mathbb{F}$ of Boolean connectives is functionally complete if and only if for each of the five defined classes, there is a member of $\mathbb{F}$ which does not belong to that class

Classification of Some Boolean functions

x	y	T	F	$\neg_2$	$\wedge$	$\vee$	$\rightarrow$	$\oplus$	$\leftrightarrow$	$\uparrow$	$\downarrow$	$[x, s, y]$		$\notin$
0	0	1	0	1	0	0	1	0	1	1	1	0	0	
0	1	1	0	0	0	1	1	1	0	1	0	0	1	
1	0	1	0	1	0	1	0	1	0	1	0	1	0	
1	1	1	0	0	1	1	1	0	1	0	0	1	1	
T		✓	✗	✗	✓	✓	✓	✗	✓	✗	✗	✓		f_1
F		✗	✓	✗	✓	✓	✗	✓	✗	✗	✗	✓		f_2
L		✓	✓	✓	✗	✗	✗	✓	✓	✗	✗	✗		f_3
M		✓	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗		f_4
S		✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗		f_5

Classification of Some Boolean functions

x	y	T	F	$\neg_2$	$\wedge$	$\vee$	$\rightarrow$	$\oplus$	$\leftrightarrow$	$\uparrow$	$\downarrow$	$[x,s,y]$		$\not\in$
0	0	1	0	1	0	0	1	0	1	1	1	0	0	
0	1	1	0	0	0	1	1	1	0	1	0	0	1	
1	0	1	0	1	0	1	0	1	0	1	0	1	0	
1	1	1	0	0	1	1	1	0	1	0	0	1	1	
T		✓	✗	✗	✓	✓	✓	✗	✓	✗	✗	✓		f_1
F		✗	✓	✗	✓	✓	✗	✓	✗	✗	✗	✓		f_2
L		✓	✓	✓	✗	✗	✗	✓	✓	✗	✗	✗		f_3
M		✓	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗		f_4
S		✗	✗	✓	✗	✗	✗	✗	✗	✗	✗	✗		f_5

- By FCT, $\mathbb{F}_1 = \{\uparrow\}$ and $\mathbb{F}_2 = \{\downarrow\}$ are both functionally complete
- $\mathbb{F}_3 = \{[x,s,y], T, F\}$ is also functionally complete (why?)
- What are some other functionally complete sets of functions?
- All rows of a counting (L class) function have the same parity (disregarding the dummy columns)

Defining $\neg$ using f_1, f_2 and f_3

- Let $f_\imath^\star(p) = f_\imath(x_1, x_2, \ldots, x_n)|_{x_1=x_2=\cdots=x_n=p}$, $\imath \in \{1, 2\}$

Defining $\neg$ using f_1, f_2 and f_3

- Let $f_\imath^\star(p) = f_\imath(x_1, x_2, \ldots, x_n)|_{x_1 = x_2 = \cdots = x_n = p}$, $\imath \in \{1, 2\}$
- Since f_1 is not T-preserving, $f_1(T) = F$, similarly $f_2(F) = T$ as f_2 is not F-preserving, leading to the following incomplete truth table

p	$f_1^\star(p)$	$f_2^\star(p)$
T	F	?
F	?	T

Defining $\neg$ using f_1, f_2 and f_3

- Let $f_\imath^\star(p) = f_\imath(x_1, x_2, \ldots, x_n)|_{x_1=x_2=\cdots=x_n=p}$, $\imath \in \{1, 2\}$
- Since f_1 is not T-preserving, $f_1(T) = F$, similarly $f_2(F) = T$ as f_2 is not F-preserving, leading to the following incomplete truth table

p	$f_1^\star(p)$	$f_2^\star(p)$
T	F	?
F	?	T

- If $f_1^\star(F) = T$ or $f_2^\star(T) = F$, $\neg$ is immediately realised

Defining ¬ using f_1, f_2 and f_3

- Let $f_\imath^\star(p) = f_\imath(x_1, x_2, \ldots, x_n)|_{x_1=x_2=\cdots=x_n=p}$, $\imath \in \{1, 2\}$
- Since f_1 is not T-preserving, $f_1(T) = F$, similarly $f_2(F) = T$ as f_2 is not F-preserving, leading to the following incomplete truth table

p	$f_1^\star(p)$	$f_2^\star(p)$
T	F	?
F	?	T

- If $f_1^\star(F) = T$ or $f_2^\star(T) = F$, ¬ is immediately realised
- Otherwise, the truth table below must realise F and T (but not ¬)

p	$f_1^\star(p)$	$f_2^\star(p)$
T	F	T
F	F	T

so $f_1^\star(\_) = F, f_2^\star(\_) = T$

Defining $\neg$ using f_1, f_2 and f_3

- Let $f_\imath^\star(p) = f_\imath(x_1, x_2, \ldots, x_n)|_{x_1=x_2=\cdots=x_n=p}$, $\imath \in \{1, 2\}$
- Since f_1 is not T-preserving, $f_1(T) = F$, similarly $f_2(F) = T$ as f_2 is not F-preserving, leading to the following incomplete truth table

p	$f_1^\star(p)$	$f_2^\star(p)$
T	F	?
F	?	T

- If $f_1^\star(F) = T$ or $f_2^\star(T) = F$, $\neg$ is immediately realised
- Otherwise, the truth table below must realise F and T (but not $\neg$)

p	$f_1^\star(p)$	$f_2^\star(p)$
T	F	T
F	F	T

so $f_1^\star(\_) = F, f_2^\star(\_) = T$

- Now, since f_3 is non-monotonic, it will have two rows

x_1	x_2	$\ldots$	x_k	$\ldots$	x_n	f_3
z_1	z_2	$\ldots$	F	$\ldots$	z_n	T
z_1	z_2	$\ldots$	T	$\ldots$	z_n	F

leading to

$p(= x_k)$	$f_3'(p)$
F	T
T	F

where $z_\imath = F = f_1^\star(\_)$ or $z_\imath = T = f_2^\star(\_)$, $\imath \neq k$

Defining T and F using f_1, f_2, f_3 and f_5

- Since f_5 isn't self complementing its truth table should have rows

$$\begin{array}{ccccc|c} x_1 & x_2 & \ldots & \ldots & x_n & f_5 \\ \hline z_1 & z_2 & \ldots & \ldots & z_n & z \\ \neg z_1 & \neg z_2 & \ldots & \ldots & \neg z_n & z \end{array}$$ leading to $\boxed{f_5' = z}$ (see below)

Defining T and F using f_1, f_2, f_3 and f_5

- Since f_5 isn't self complementing its truth table should have rows

x_1	x_2	...	...	x_n	f_5
z_1	z_2	...	...	z_n	z
$\neg z_1$	$\neg z_2$	...	...	$\neg z_n$	z

leading to $\boxed{f_5' = z}$ (see below)

- In the top row, if $z_\imath = T$, replace it by p, otherwise with $\neg p = f_3'(p)$

Defining T and F using f_1, f_2, f_3 and f_5

- Since f_5 isn't self complementing its truth table should have rows

x_1	x_2	$\ldots$	$\ldots$	x_n	f_5
z_1	z_2	$\ldots$	$\ldots$	z_n	z
$\neg z_1$	$\neg z_2$	$\ldots$	$\ldots$	$\neg z_n$	z

leading to $\boxed{f_5' = z}$ (see below)

- In the top row, if $z_\imath = T$, replace it by p, otherwise with $\neg p = f_3'(p)$
- Note that $\neg z_\imath = f_3'(z_\imath)$, the output ($f_5'$) is constant, either T or F

Defining T and F using f_1, f_2, f_3 and f_5

- Since f_5 isn't self complementing its truth table should have rows

x_1	x_2	$\ldots$	$\ldots$	x_n	f_5
z_1	z_2	$\ldots$	$\ldots$	z_n	z
$\neg z_1$	$\neg z_2$	$\ldots$	$\ldots$	$\neg z_n$	z

leading to $\boxed{f_5' = z}$ (see below)

- In the top row, if $z_\imath = T$, replace it by p, otherwise with $\neg p = f_3'(p)$
- Note that $\neg z_\imath = f_3'(z_\imath)$, the output ($f_5'$) is constant, either T or F
- The other constant truth value may be obtained as $\boxed{f_3'(f_5')}$

Defining T and F using f_1, f_2, f_3 and f_5

- Since f_5 isn't self complementing its truth table should have rows

$$\begin{array}{ccccc|c} x_1 & x_2 & \ldots & \ldots & x_n & f_5 \\ \hline z_1 & z_2 & \ldots & \ldots & z_n & z \\ \neg z_1 & \neg z_2 & \ldots & \ldots & \neg z_n & z \end{array}$$ leading to $\boxed{f_5' = z}$ (see below)

- In the top row, if $z_\imath = T$, replace it by p, otherwise with $\neg p = f_3'(p)$
- Note that $\neg z_\imath = f_3'(z_\imath)$, the output ($f_5'$) is constant, either T or F
- The other constant truth value may be obtained as $\boxed{f_3'(f_5')}$
- Thus, both T and F may be generated using f_1, f_2, f_3 and f_5

Defining $g(p, q)$ with an odd number of Ts

- f_4 is not counting, so its TT will have (at least) two inputs $\langle x_1, \ldots, x_n\rangle$ and $\langle y_1, \ldots, y_n\rangle$ st
  - $f_4(u_1, \ldots, u_\imath, \ldots, u_n) = f_4(u_1, \ldots, \neg u_\imath, \ldots, u_n)$ as f_4 is not counting
  - $f_4(v_1, \ldots, v_\imath, \ldots, v_n) \neq f_4(v_1, \ldots, \neg v_\imath, \ldots, v_n)$ position-$\imath$ isn't dummy
- Parity of Ts in the pairs of rows will be different, these rows will be used to define $g(p, q)$
- The four rows and also the column reduction scheme ($\imath \neq \jmath$), z_1, z_2 are either T or F

$x_1 \ldots$	$x_\imath$	$\ldots x_n$	$u_\jmath = v_\jmath = T$	$u_\jmath = v_\jmath = F$	$u_\jmath = F$ $v_\jmath = T$	$u_\jmath = T$ $v_\jmath = F$	$x_\imath = q$	$g(p, q)$
$u_1 \ldots$	$u_\imath$	$\ldots u_n$	T	F	$\neg p$	p	q	z_1
$u_1 \ldots$	$\neg u_\imath$	$\ldots u_n$	T	F	$\neg p$	p	$\neg q$	z_1
$v_1 \ldots$	$v_\imath$	$\ldots v_n$	T	F	p	$\neg p$	q	z_2
$v_1 \ldots$	$\neg v_\imath$	$\ldots v_n$	T	F	p	$\neg p$	$\neg q$	$\neg z_2$

- All TTs with odd number of Ts can now be generated by enumerating the symbolic TT

g	p	q	g_1	g_2	g_3	g_4	g	p	q	g_5	g_6	g_7	g_8
z_1	T	T	T	T	F	F	z_2	T	T	T	F	T	F
z_1	T	F	T	T	F	F	$\neg z_2$	T	F	F	T	F	T
z_2	F	T	T	F	T	F	z_1	F	T	T	T	F	F
$\neg z_2$	F	F	F	T	F	T	z_1	F	F	T	T	F	F
Common Boolean fn			$\vee$	$\leftarrow$	$\not\leftarrow$	$\downarrow$				$\rightarrow$	$\uparrow$	$\wedge$	$\not\rightarrow$

- Each g_i with T, F and complementation, as required, is functionally complete