Functional Completeness

Instructor:

Deepak Poonia

MTech, IISc Bangalore

GATE CSE AIR 53; AIR 67;
AIR 107; AIR 206; AIR 256

Digital Logic Complete Course:

https://www.goclasses.in/courses/Digital-Logic

Join **GO+ GO Classes Combined Test Series** for BEST quality tests, matching GATE CSE Level:

Visit www.gateoverflow.in website to join Test Series.

1. **Quality Questions:** No Ambiguity in Questions, All Well-framed questions.
2. Correct, **Detailed Explanation**, Covering Variations of questions.
3. **Video Solutions.**

https://gateoverflow.in/blog/14987/gate-overflow-and-go-classes-test-series-gate-cse-2024

Join GO Classes **GATE CSE Complete Course** now:

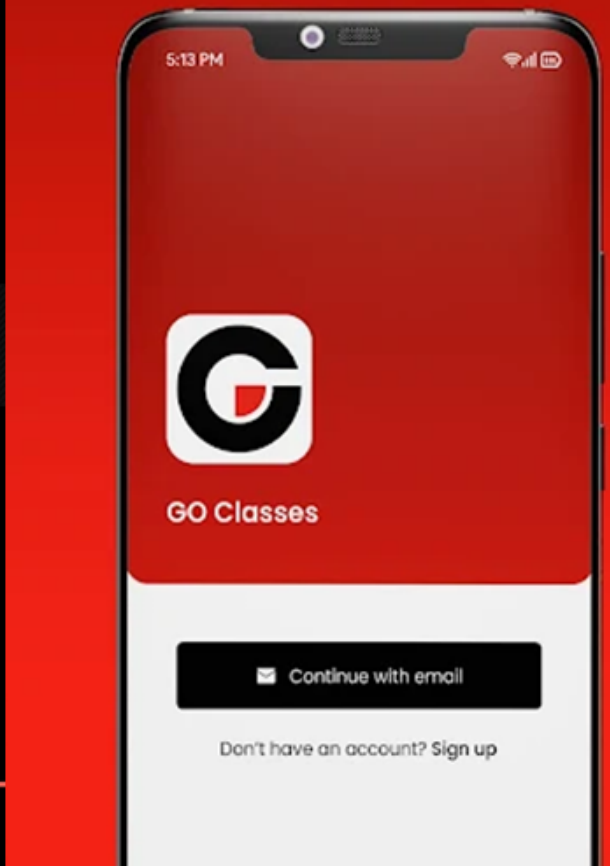
https://www.goclasses.in/s/pages/gatecompletecourse

1. Quality Learning: No Rote-Learning. Understand Everything, from basics, In-depth, with variations.
2. Daily Homeworks, Quality Practice Sets, Weekly Quizzes.
3. Summary Lectures for Quick Revision.
4. Detailed Video Solutions of Previous ALL GATE Questions.
5. Doubt Resolution, Revision, Practice, a lot more.

Download the GO Classes Android App:

https://play.google.com/store/apps/details?id=com.goclasses.courses

Search "GO Classes" on Play Store.



NOTE :

Complete Discrete Mathematics & Complete Engineering

Mathematics Courses, by GO Classes, are **FREE** for ALL learners.

Visit here to watch : https://www.goclasses.in/s/store/

SignUp/Login on Goclasses website for free and start learning.

We are on Telegram. **Contact us** for any help.

Link in the Description!!

Join GO Classes **Doubt Discussion** Telegram Group :

Username:

@GATECSE_GOCLASSES

We are on Telegram. **Contact us** for any help.

Join GO Classes Telegram Channel, Username: **@GOCLASSES_CSE**

Join GO Classes **Doubt Discussion** Telegram Group :

Username: **@GATECSE_Goclasses**

(Any doubt related to Goclasses Courses can also be asked here.)

Join GATEOverflow Doubt Discussion Telegram Group :

Username: **@GateOverflow_CSE**

GO CLASSES


GO CLASSES
GO CLASSES
GO CLASSES
GO CLASSES
GO CLASSES
GO CLASSES

Functional Completeness in Digital Logic

Functional Completeness (FC)

So Far…

- Definition & Idea of Functional Completeness
- Many Examples
- Some Guidelines to determine if a Set of Boolean Functions is FC or Not.

Functional Completeness (FC)

Next:

Source of our Guidelines

Guidelines to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

$X = \{ f_1, f_2, f_3, f_4 \}$

Guidelines to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions. $X = \{ f_1, f_2, f_3, f_4 \}$

1. Minimize f_i.
2. $\forall_{f_i} (f_{i(0)} = 0)$ then X is NOT FC.

Guidelines to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions. $X = \{ f_1, f_2, f_3, f_4 \}$

(3) $\forall_{f_i} \left(f_i(1) = 1 \right)$ then X is NOT FC.

Guidelines to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions. $X = \{f_1, f_2, f_3, f_4\}$

(4) $\forall_{f_i}$ (f_i is Self Dual) then X is NOT FC.

(5) $\forall_{f_i}$ (f_i is Linear) then X is NOT FC.

Guidelines to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions. $X = \{f_1, f_2, f_3, f_4\}$

(6) Else Try to create {NOT, OR} or

or {NOR} OR {NAND} OR {NOT, AND}

OR

Source of our Guidelines:

Emil Post's

Functional Completeness Theorem

Notre Dame Journal of Formal Logic
Volume 31, Number 2, Spring 1990

Post's Functional Completeness Theorem

FRANCIS JEFFRY PELLETIER and NORMAN M. MARTIN*

Abstract The paper provides a new proof, in a style accessible to modern logicians and teachers of elementary logic, of Post's Functional Completeness Theorem. Post's Theorem states the necessary and sufficient conditions for an arbitrary set of (2-valued) truth functional connectives to be expressively complete, that is, to be able to express every (2-valued) truth function or truth table. The theorem is stated in terms of five properties that an arbitrary connective may have, and claims that a set of connectives is expressively complete iff for each of the five properties there is a connective that lacks that property.

Theorem. *A system of boolean functions is functionally complete if and only if this system does not* entirely *belong to any of* T_0, T_1, L, M, S.

The mentioned theorem is called *Post's completeness criterion* and is due to Emil Post. What this criterion says is that in order for a system of boolean functions to be functionally complete, this system should have

- at least one function that does *not* preserve zero (i.e. it is *not* in T_0), and
- at least one function that does *not* preserve one (i.e. it is *not* in T_1), and
- at least one function that is *not* linear (i.e. it is *not* in L), and
- at least one function that is *not* monotone (i.e. it is *not* in M), and
- at least one function that is *not* self-dual (i.e. it is *not* in S).

So far

Guidelines

Now

Theorem → iff statement

THEOREM

to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Input

↳ The Question

Let X be a Set of Boolean Functions.

$X = \{ f_1, f_2, f_3, f_4 - - \}$

THEOREM to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions. $X = \{f_1, f_2, f_3\}$

Property 1 to check:

Let X be a Set of Boolean Functions.

Property 1 to check:

$X = \{ f_1, f_2, f_3 \}$

$\forall_{f_i} \left(f_i(0) = 0 \right)$ then

X is NOT Fc.

Let X be a Set of Boolean Functions.

Property 1 to check:

"0-preserving" boolean function :

$f(a, b, c)$ is 0-preserving iff

$$f(0, 0, 0) = 0.$$

Let X be a Set of Boolean Functions.

Property 1 to check:

0-preserving boolean function :

$f(a_1, a_2, - -, a_n)$ is 0-preserving iff

$$f(0, 0, - -, 0) = 0.$$

Let X be a Set of Boolean Functions.

Property 1 to check:

Result: If all functions in X are 0-preserving then X is NOT Fc.

Q: which of the following functions are 0-preserving ?

(1) $f(a,b) = a + b$

(2) ab

(3) $a \oplus b$

(4) $a \odot b$

(5) $a \rightarrow b$

(6) $\bar{a}$

(7) a

(8) $f(a,b) = 0$

(9) $f(a,b) = 1$

(10) $a \uparrow b$

(11) $a \downarrow b$

Q: which of the following functions are 0-preserving ?

1. $f(a,b) = a + b$
2. ab
3. $a \oplus b$
4. $a \odot b$
5. $a \rightarrow b$
6. $\bar{a}$
7. a
8. $f(a,b) = 0$
9. $\times$
10. $f(a,b) = 1$
11. $a \uparrow b$
12. $a \downarrow b$

$f(a,b) = a \odot b$ is NOT 0-preserving.

because $f(0,0) = 0 \odot 0 = 1 \neq 0$

$f(a,b) = a \rightarrow b$ is NOT 0-preserving

because $f(0,0) = 0 \rightarrow 0 = 1 \neq 0$

Q: which of the following functions are 0-preserving?

$f(0, 0, 0, \ldots, 0) = 0$

Property 1: We say that boolean function f *preserves zero*, if on the 0-input it produces 0. By the 0-input we mean such an input, where every input variable is 0 (this input usually corresponds to the first row of the truth table). We denote the class of zero-preserving boolean functions as T_0 and write $f \in T_0$.

Here are the truth tables for a few boolean functions we are familiar with:

x	$\neg x$
0	1
1	0

x_1	x_2	$x_1 \wedge x_2$
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	$x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1

We see that on 0-input, both $\wedge$ and $\vee$ produce 0, while $\neg$ produces 1. Thus, $\wedge$ and $\vee$ preserve zero, and $\neg$ does not. We write $\wedge \in T_0$ and $\vee \in T_0$, but $\neg \notin T_0$.

THEOREM to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

Property 2 to check:

Let X be a Set of Boolean Functions.

Property 2 to check:

1-Preserving boolean function :

f(a,b,c) is 1-preserving iff

$$f(1,1,1) = 1.$$

Let X be a Set of Boolean Functions.

Property 2 to check:

1-Preserving boolean function :

$f(a_1, a_2, --, a_n)$ is 1-preserving iff

$f(1, 1, --, 1) = 1$.

Let X be a Set of Boolean Functions.

Property 2 to check:

Result 2 :

If **all** functions in X are 1-preserving

then X is **NOT** FC.

1-preserving functions :

$a\,b$, $a + b$, $a \odot b$, a, $a \rightarrow b$, 1

NOT 1-preserving functions :

$a \oplus b$, $\overline{a}$, $a \uparrow b$, $a \downarrow b$, 0

$a \oplus b$ is NOT 1-preserving

because $1 \oplus 1 = 0 \neq 1$.

XNOR is Same as Biimplication.

$a \odot b \equiv a \leftrightarrow b$

Property 2: Similarly to T_0, we say that boolean function f *preserves one*, if on 1-input, it produces 1. The 1-input is the input where all the input variables are 1 (this input usually corresponds to the last row of the truth table). We denote the class of one-preserving boolean functions as T_1 and write $f \in T_1$.

Looking at the truth tables of out three functions, we see that $\neg$ does not preserve one since it produces 0 on 1-input. However, both $\wedge$ and $\vee$ preserve 1, since their values on 1-input are both 1.

x	$\neg x$
0	1
1	0

x_1	x_2	$x_1 \wedge x_2$
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	$x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1

Thus, $\wedge \in T_1$ and $\vee \in T_1$, but $\neg \notin T_1$.

Q: If a function f is 0-preserving then what we can say about f being 1-preserving?

Q: If a function f is 0-preserving then what we can say about f being 1-preserving?

Can NOT say Anything.

$f(a, b, c)$

a	b	c	f
0	0	0	

→ used to check if f is 0-preserving or NOT.

1	1	1	

→ used to check if f is 1-preserving.

function which is 0-preserving & 1-preserving : $a\,b$, $a+b$, a

$a \oplus b \longrightarrow$ 0-preserving ✓ 1-pre ✗

$a \odot b \longrightarrow$ 0-pre ✗ 1-pre ✓

$a \uparrow b \longrightarrow$ 0-pre ✗ 1-pre ✗

No Relationship b/w

0-preserving functions

&

1-preserving functions

THEOREM to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

Property 3 to check:

Let X be a Set of Boolean Functions.

Property 3 to check:

Self Dual property

$X = \{f_1, f_2, f_3\}$

Result 3: If all functions in X are Self Dual then X is NOT FC.

Self Dual function :

$$f = f^d$$

Self Dual functions :

a , $\bar{a}$

NoT self Dual functions :

$a+b$, ab , 0 , 1 , $a \oplus b$, $a \odot b$

$a \uparrow b$, $a \downarrow b$

$f = \bar{a}$

$f^d = \bar{a}$

Self Dual

$g = a \oplus b$

$g^d = a \odot b$

$g \neq g^d$

Not self Dual

THEOREM to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

Property 4 to check:

Let X be a Set of Boolean Functions.

Property 4 to check:

$X = \{f_1, f_2, f_3\}$ → Linear function

Result: If Every function in X is Linear then X is NOT FC.

Property 3: We say that boolean function f is *linear* if one of the following two statements holds for f:

- For every 1-value of f, the number of 1's in the corresponding input is *odd*, and for every 0-value of f, the number of 1's in the corresponding input is *even*.

  or

- For every 1-value of f, the number of 1's in the corresponding input is *even*, and for every 0-value of f, the number of 1's in the corresponding input is *odd*.

If one of these statements holds for f, we say that f is linear[1]. We denote the class of linear boolean functions with L and write $f \in L$.

f is Linear iff

For all f=1 Rows
number of 1's in input is Even

&

for all f=0 Rows
number of i's in input is odd.

OR

for all f=1 Rows
number of i's in input is odd

&

for all f=0 Rows
number of i's in input is Even.

$f = \bar{a}$ is Linear OR not ?

a	$f = \bar{a}$
0	1
1	0

f is Linear.

for all f = 1 Rows
↳ number of 1's in input is Even.

&

for all f = 0 Rows
↳ number of 1's in input is Odd

$f = ab$ is Linear OR not ?

a	b	ab
0	0	0
0	1	0
1	0	0
1	1	1

→ Not Linear

false ↑

for all f=1 Rows

↳ even number of 1's in input

for all f=0 Rows

↳ odd number of 1's in input

$f = a + b$ is Linear or not ?

a	b	a + b
0	0	0
0	1	1
1	0	1
1	1	1

"for all f = 1 Rows"

neither even 1's in input

nor odd 1's in Input.

Not Linear

$f = a \oplus b$ is Linear or not ?

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

for all $f=1$ Rows → odd 1's in input

for all $f=0$ Rows → even 1's in input.

$\oplus$ is Linear.

$f = a \odot b$ is Linear or not ?

a	b	$a \odot b$
0	0	1
0	1	0
1	0	0
1	1	1

for all $f=1$ Rows

↳ even 1's in input

&

for all $f=0$ Rows

↳ odd 1's in input.

$\odot$ is Linear.

$f = a \rightarrow b$ is Linear OR not ?

a	b	$a \rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1

Not Linear

$f = 1$ Rows

Sometimes even 1's in input

Sometimes odd 1's in input.

$f = a \uparrow b$ is Linear or not ?

a	b	$a \uparrow b = \overline{ab}$
0	0	1
0	1	1
1	0	1
1	1	0

$f = 1$ Rows

- Sometime even #'s in input
- Sometime odd #'s in input

$f = a \downarrow b$ is Linear or not ?

a	b	$a \downarrow b = \overline{a+b}$
0	0	1
0	1	0
1	0	0
1	1	0

$f = 1$ Rows

↳ even 1's in input

for all $f = 0$ Rows

↳ odd 1's in input

Now, let us check whether our three functions are linear.

x	$\neg x$
0	1
1	0

x_1	x_2	$x_1 \wedge x_2$
0	0	0
0	1	0
1	0	0
1	1	1

x_1	x_2	$x_1 \vee x_2$
0	0	0
0	1	1
1	0	1
1	1	1

In the case of $\neg$, for every 1-value of this function, the number of 1's among the input variables is always even. Actually, there is only one 1-value of $\neg$, and the number of 1's in the corresponding input is 0, which is an even number. Similarly, for every 0-value of $\neg$ (and there is only one such value), the number of 1's in the input is odd (we have the only input variable, and it is 1 in the corresponding row). Thus, $\neg$ is linear.

Using the same reasoning, we can show that neither $\wedge$ nor $\vee$ is linear. $\wedge$ is not linear because on 0 outputs, the number of 1's in the corresponding inputs is sometimes even and sometimes odd. Thus, neither of two conditions from our definition can hold for $\wedge$. Similarly, $\vee$ is not linear because on 1-outputs, the number of 1's in the corresponding inputs is sometimes even and sometimes odd.

Thus, $\neg \in L$, but $\wedge \notin L$ and $\vee \notin L$.

THEOREM to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

Property 5 to check:

Let X be a Set of Boolean Functions.

Property 5 to check:

Monotonic Property

Result 5: $X = \{f_1, f_2, f_3\}$

If all functions in X are monotonic then X is NOT FC.

So far :

Boolean function

{ (1) 0-preserving
(2) 1-preserving
(3) self Dual
(4) Linear
(5) monotonic }

THEOREM

to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

Emil Post's fc Theorem:

Input: $X = \{ f_1, f_2, f_3, f_4 \}$ = set of Boolean functions

Question to Answer:

X is fc OR Not.

Emil Post's fc Theorem:

Input: $X = \{ f_1, f_2, f_3, f_4 \}$ = set of Boolean functions

Theorem: X is **NOT fc** iff

(1) all f_i are 0-preserving.

OR

(2) all f_i are 1-preserving.

Emil Post's fc Theorem:

Input: $X = \{ f_1, f_2, f_3, f_4 \}$ = set of Boolean functions

Theorem: X is NOT fc iff

(3) all f_i are self Dual

OR

(4) all f_i are Linear. OR

Emil Post's fc Theorem:

Input: $X = \{ f_1, f_2, f_3, f_4 \}$ = set of Boolean functions

Theorem: X is **NOT fc** iff

(5) **all** f_i are monotonic.

Emil Post's FC Theorem:

Input: $X = \{f_1, f_2, f_3, f_4\}$ = set of Boolean functions

Theorem: X is FC iff

(1) at least one of f_i is NOT 0-preserving and

(2) at least one of f_i is NOT 1-preserving.

Emil Post's fc Theorem:

Input: $X = \{f_1, f_2, f_3, f_4\}$ = set of Boolean functions

Theorem: X is fc iff

and

(3) at least one of f_i is Not self Dual

and

Emil Post's fc Theorem:

Input: $X = \{f_1, f_2, f_3, f_4\}$ = set of Boolean functions

Theorem: X is fc iff

and

(i) at least one of f_i is NOT Linear

and

Emil Post's fc Theorem:

Input: $X = \{f_1, f_2, f_3, f_4\}$ = set of Boolean functions

Theorem: X is fc iff

and

(5) at least one of f_i is NOT monotonic.

Given $X = \{f_1, f_2, f_3, \cdots\}$

X is FC iff

atleast one of f_i is NOT 0-pre.

" " " " " NOT 1-pre.

" " " " " NOT self dual

" " " " " NOT Linear

" " " " " NOT monotonic

Given $X = \{f_1, f_2, f_3, \dots\}$

X is **NOT FC** iff

All f_i are 0-preserving

OR

All f_i " 1-preserving

OR

" " " Self Duel

OR

" " " Linear (OR) all f_i are monotonic.

Theorem. *A system of boolean functions is functionally complete if and only if this system does not* entirely *belong to any of* T_0, T_1, L, M, S.

The mentioned theorem is called *Post's completeness criterion* and is due to Emil Post. What this criterion says is that in order for a system of boolean functions to be functionally complete, this system should have

- at least one function that does *not* preserve zero (i.e. it is *not* in T_0), and
- at least one function that does *not* preserve one (i.e. it is *not* in T_1), and
- at least one function that is *not* linear (i.e. it is *not* in L), and
- at least one function that is *not* monotone (i.e. it is *not* in M), and
- at least one function that is *not* self-dual (i.e. it is *not* in S).

Theorem (Post's Functional Completeness Theorem) *A set X of truth functions (of 2-valued logic) is functionally complete if and only if, for each of the five defined classes, there is a member of X which does not belong to that class.*

Functional Completeness (FC)

Next:

Linear Boolean Function

(Revisited)

First Minimize f then Check f is Linear or Not

Property 3: We say that boolean function f is *linear* if one of the following two statements holds for f:

- For every 1-value of f, the number of 1's in the corresponding input is *odd*, and for every 0-value of f, the number of 1's in the corresponding input is *even*.

  or

- For every 1-value of f, the number of 1's in the corresponding input is *even*, and for every 0-value of f, the number of 1's in the corresponding input is *odd*.

If one of these statements holds for f, we say that f is linear[1]. We denote the class of linear boolean functions with L and write $f \in L$.

Important NOTE:

Before checking a Function f is Linear Or Not...

Minimize f.

$f(a,b) = a + ab$

$a + ab = a$

Absorption Law

a	b	a + ab
0	0	0
0	1	0
1	0	1
1	1	1

$f = 1$ Rows

Sometimes even 1's in input

Sometimes odd 1's in input.

$f = a$ → Linear

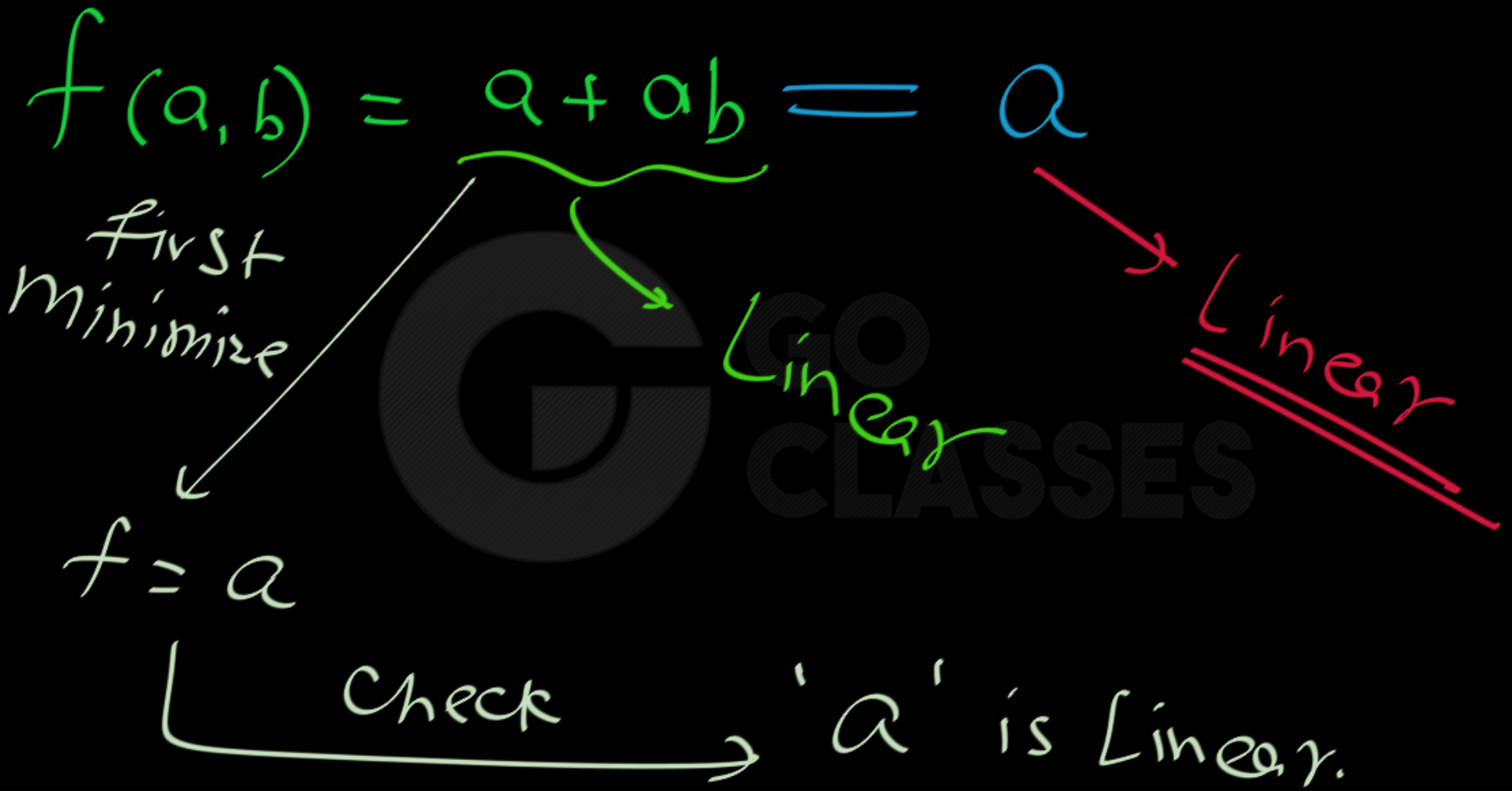
a	f = a
0	0
1	1

$f = a$ is Linear.

for all f = 1 Rows → odd 1's in input

&

for all f = 0 Rows → even 1's in input.


f(a, b) = a + ab = a
first Minimize
Linear
Linear
f = a
Check
'a' is Linear.

$$f(a,b) = a + ab = a$$

f Does not depend on b.

b is called Dummy Variable.

Source:

Emil Post's Paper

Type 3: Counting functions. [Post: "Alternating functions". The set of all functions generated by alternating functions is his L_1.] A counting function is one in which every nondummy position always makes a difference. That is, given any row of a truth table, if you ignore values of dummy positions, a change in the value of one argument (holding all others constant) will create a change in the value of the function. So: each position of such a function either never makes a difference ("dummy position") or else it *always* makes a difference. This means that countability can be easily tested in the following way. First, delete dummy positions. Then a function is counting if one of the following two situations occurs: (a) in every row in which the value of the function is T, there are an even number of T's assigned to the arguments of the function, and in every row in which the function is F, there are an odd number of T's assigned to the arguments of the function; or (b) in every row in which the value of the function is T, there are an odd number of T's assigned to the arguments of the function, and in every row in which the function is F, there are an even number of T's assigned to the arguments of the function. That this is an adequate test can be seen

Linear boolean function

→ Emil Post names it Alternating function

→ one author called it Counting function

Source:

Emil Post's Paper

Type 3: Counting functions. [Post: "Alternating functions". The set of all functions generated by alternating functions is his L_1.] A counting function is one in which every nondummy position always makes a difference. That is, given any row of a truth table, if you ignore values of dummy positions, a change in the value of one argument (holding all others constant) will create a change in the value of the function. So: each position of such a function either never makes a difference ("dummy position") or else it *always* makes a difference. This means that countability can be easily tested in the following way. First, delete dummy positions. Then a function is counting if one of the following two situations occurs: (a) in every row in which the value of the function is T, there are an even number of T's assigned to the arguments of the function, and in every row in which the function is F, there are an odd number of T's assigned to the arguments of the function; or (b) in every row in which the value of the function is T, there are an odd number of T's assigned to the arguments of the function, and in every row in which the function is F, there are an even number of T's assigned to the arguments of the function. That this is an adequate test can be seen

Note :

Given f, To check is it Linear or Not : first minimize f

all Dummy Variables gone

Note :

Given f, To check is it Linear or Not : first minimize f then Create truth table without Dummy Variables. & Check Linearity.

Q: $f(a,b,c) = \bar{a}\,b + a\,\bar{b} + a\,\bar{b}\,c$

Is f Linear or Not.

Q: $f(a,b,c) = \bar{a}b + a\bar{b} + a\bar{b}c$

Absorption Law

Is f Linear or Not.

① Minimize f:

$f = \bar{a}b + a\bar{b}$ (c way Dummy Variable)

$f = \bar{a}b + a\bar{b}$

a	b	f
0	0	0
0	1	1
1	0	1
1	1	0

Linear

Note:

Checking f is Linear or Not :

(1) Minimize f : All Dummy variables will be gone.

(2) Now create Truth Table without Dummy Variables & Check Linearity.

$f(a,b) = 1$ is Linear or Not?

Q: $f(a,b) = 1$ is Linear or Not?

a	b	f
0	0	1
0	1	1
1	0	1
1	1	1

WRONGly you will get "Not Linear" answer.

$f(a,b) = 1$ → Minimize = 1

Create Truth Table without Dummy Variables.

0, 1 are Linear.

Note: Constant functions 0, 1 are Linear.

sfu.ca/~jeffpell/papers/PostPellMartin.pdf

PostPellMartin.pdf 3 / 14 198%

[redacted] Note finally that the constant functions **T** and **F** are each counting functions, since they do not have any nondummy variables.

Functional Completeness (FC)

Next:

Monotonic Boolean Functions

Monotonic Boolean function:

→ Monotone

for function f, if increasing the input Combination by changing simple input from 0 to 1 then f value must not Decreaye.

Monotonic Boolean function :

→ Monotone

for function f, if increasing the input Combination by changing simple input from 0 to 1 then f must Not go to from 1 to 0.

Monotonic Boolean function :

→ Monotone

a	b	c	f
1	0	0	0
1	0	1	0/1

Monotonic Boolean function :

→ Monotone

a	b	c	f
1	0	0	1
1	0	1	1

Monotonic Boolean function :

→ Monotone

a	b	c	f
0	0	0	1
0	1	0	1

Monotonic Boolean function :

→ Monotone

a	b	c	f
x	0	y	0
x	1	y	0/1

Monotonic Boolean function :

→ Monotone

a	b	c	f
x	0	y	1
x	1	y	1

Monotonic Boolean function:

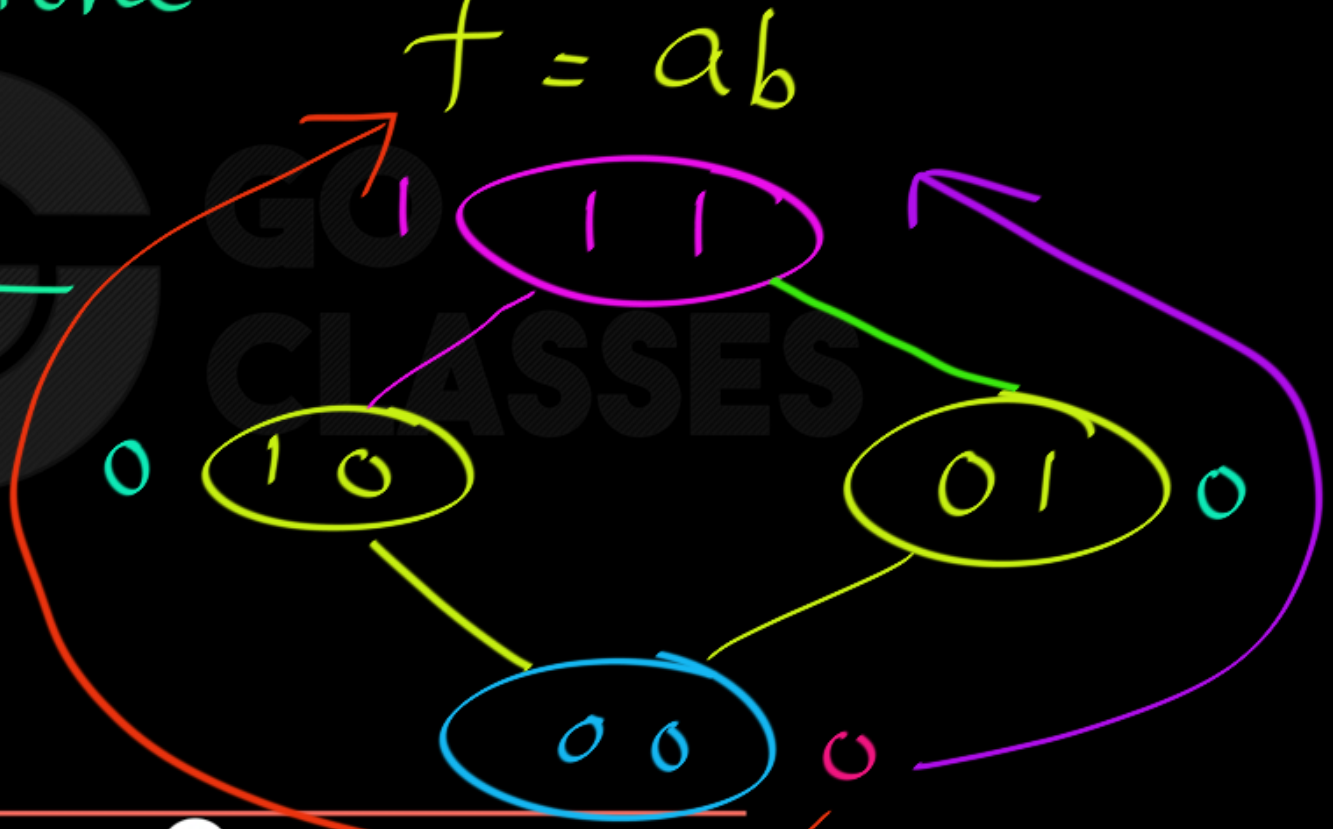
→ Monotone

f is monotonic iff increasing input Combination by changing Simple input from 0 to 1 then f must NoT Decrease.

Monotonic Boolean function → Monotonic

→ Monotone

a	b	f = ab
0	0	0
0	1	0
1	0	0
1	1	1

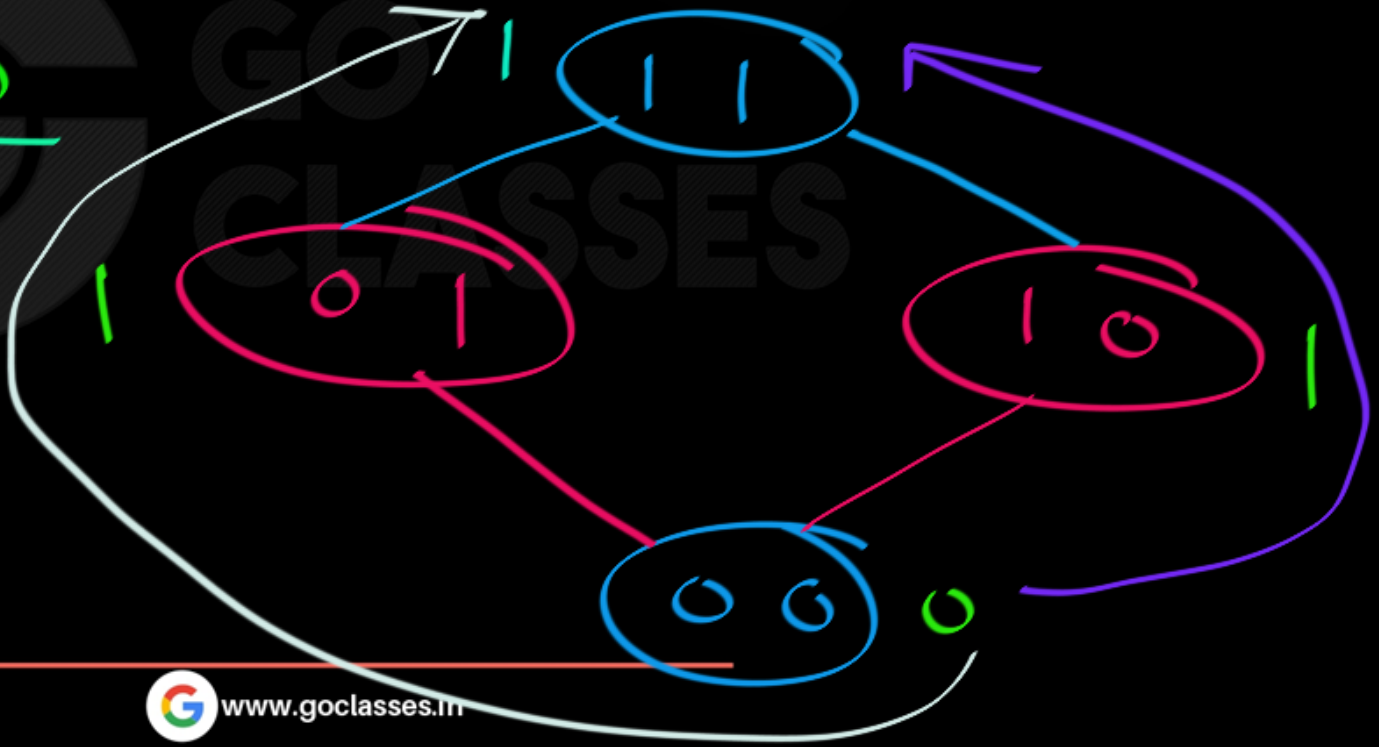


Monotonic Boolean function : Monotonic

Monotone

$f = a + b$

a	b	f = a + b
0	0	0
0	1	1
1	0	1
1	1	1

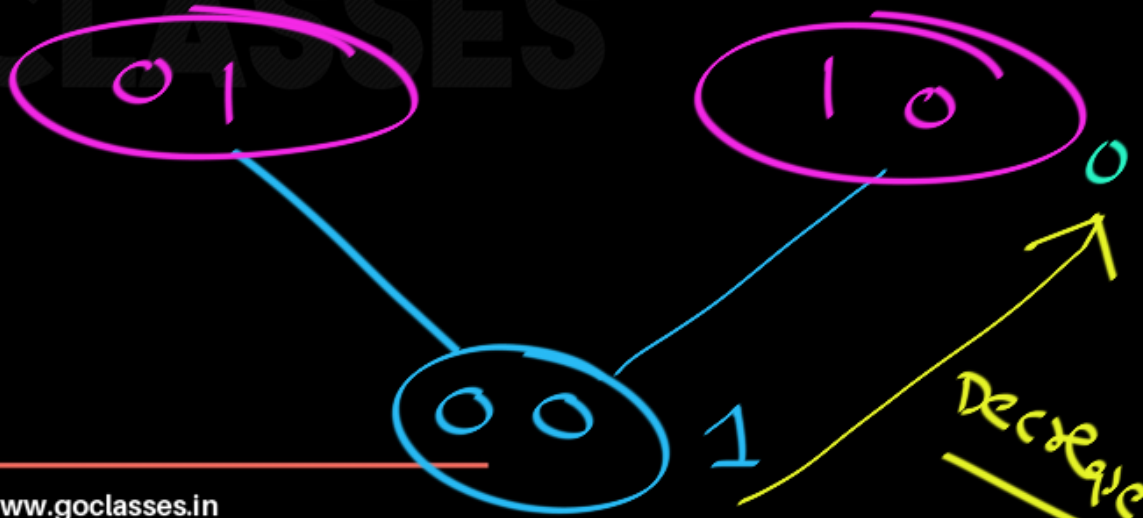


Monotonic Boolean function :

→ Monotone

$f = a \rightarrow b$ → NOT monotonic

a	b	$f = a \rightarrow b$
0	0	1
0	1	1
1	0	0
1	1	1



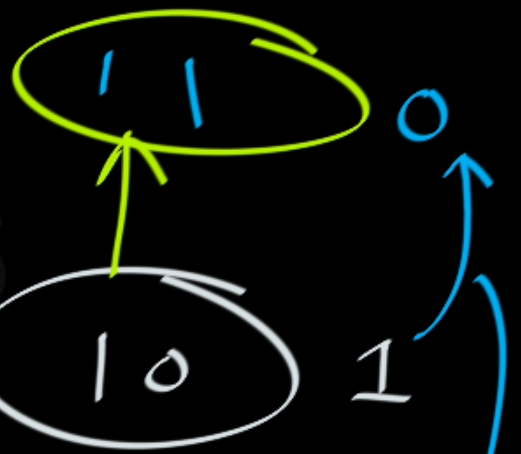
Monotonic Boolean function:

→ Monotone

→ NOT Monotonic

$f = a \oplus b$

a	b	$f = a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0



Monotonic Boolean function:

Monotone

$f = a \odot b$

Not monotonic

a	b	$f = a \odot b$
0	0	1
0	1	0
1	0	0
1	1	1

0 1 0

f value decreasing

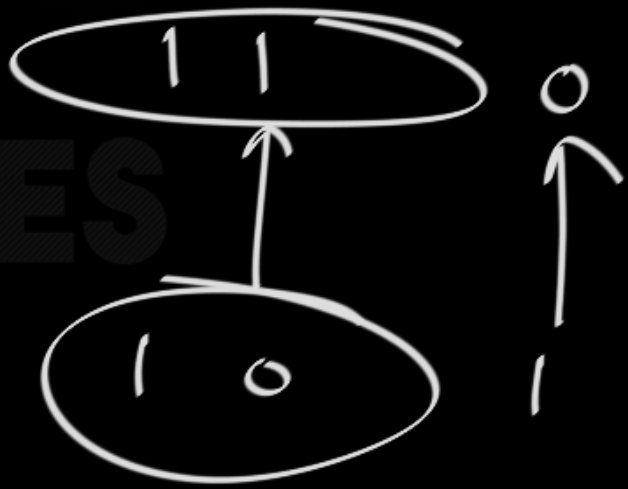
0 0 1

Monotonic Boolean function :

→ Monotone

$f = a \uparrow b$ → NOT monotonic

a	b	$f = a \uparrow b = \overline{ab}$
0	0	1
0	1	1
1	0	1
1	1	0



Monotonic Boolean function:

→ Monotone

$f = a \oplus b \oplus c$

NOT Monotonic

a	b	c	f
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	0
1	1	1	1

0 1 1 → 0

0 0 1 → 1

f value Decrease

Monotonic Boolean function:

→ Monotone

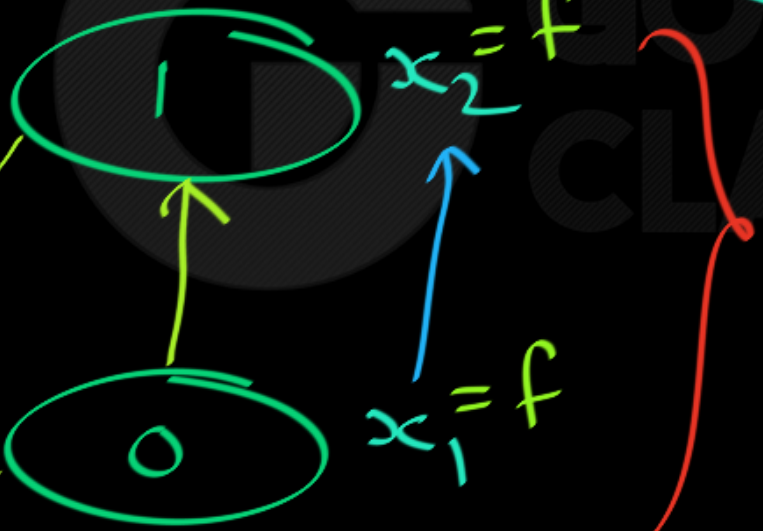
To check if f is monotonic:

1. Create a Lattice for f.
2. On every upward path, f value must NOT Decrease.

f(a)

Lattice :

a	f
0	x_1
1	x_2



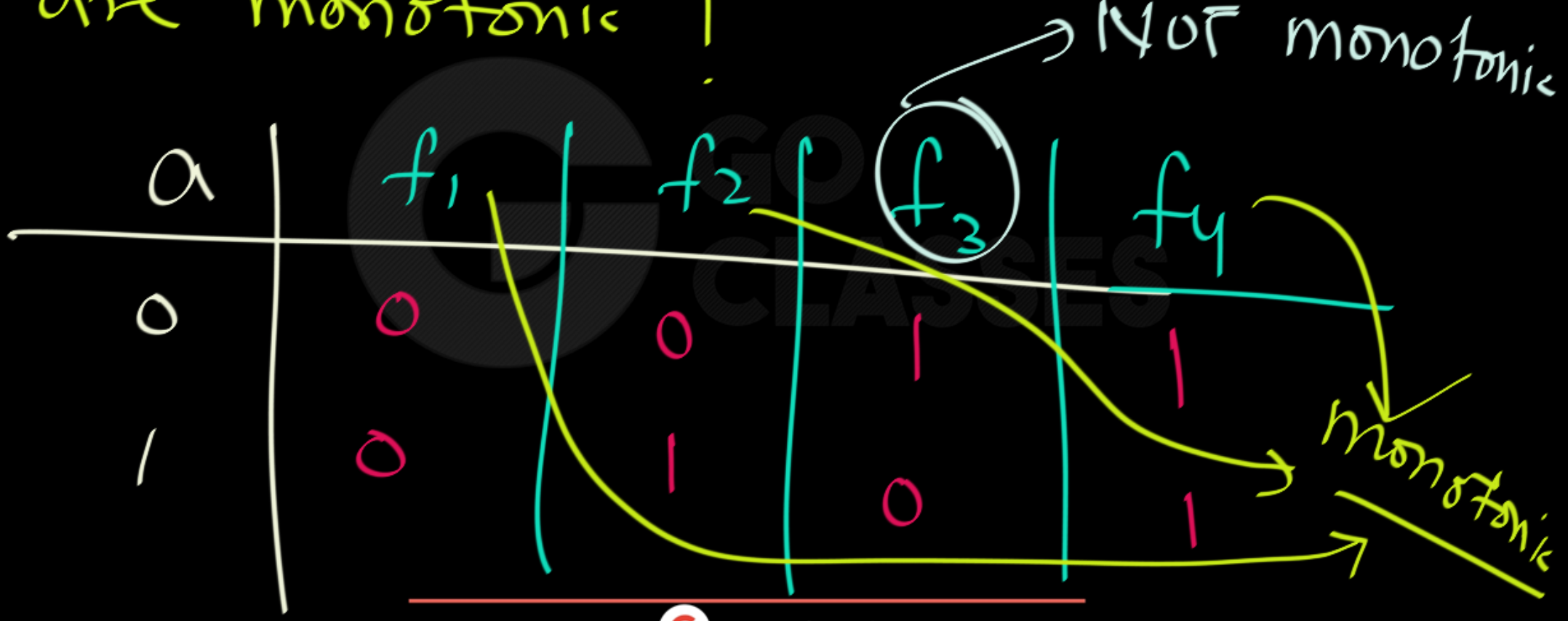
f will monotonic iff on every upward Path, f value NOT Decrease.

Lattice

for f(a):

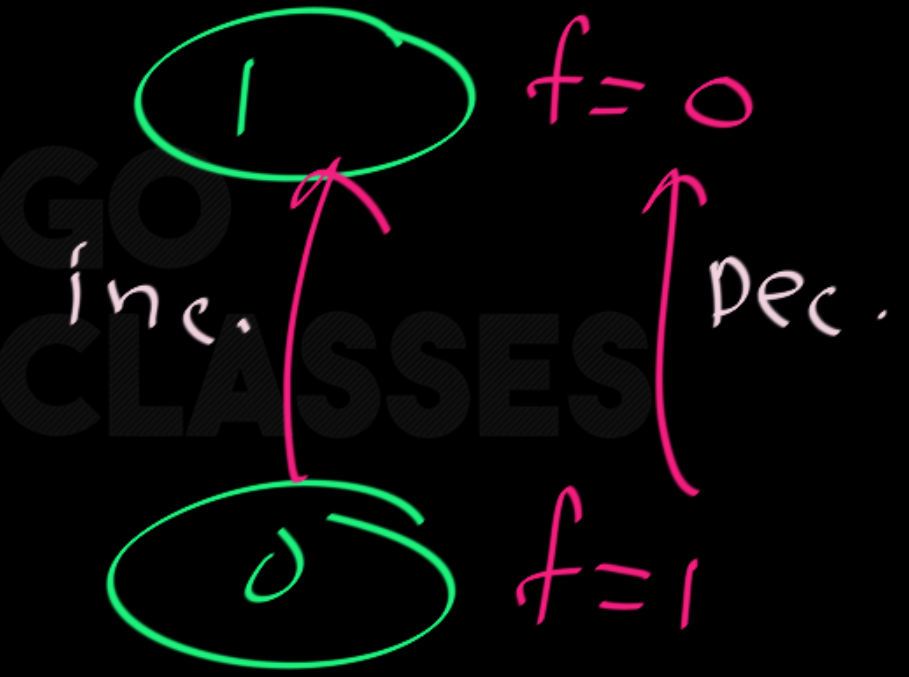
$x_2 \geq x_1$

On a Single Value, which functions are monotonic ?



$f(a) = \overline{a} \Rightarrow$ Not monotonic

a	f
0	1
1	0



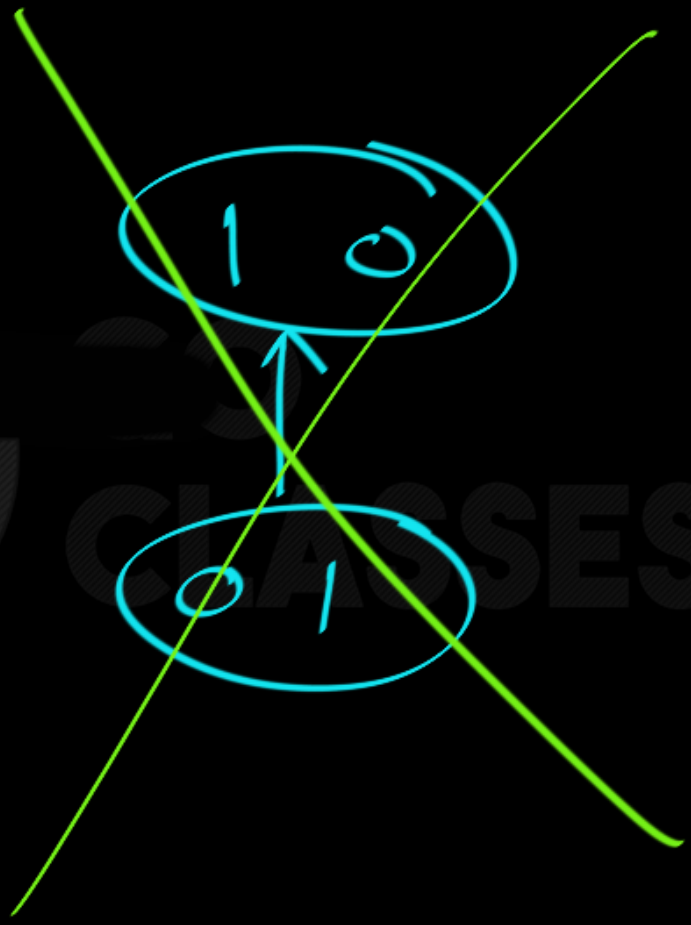
$f(a,b)$

a	b	f
0	0	0
0	1	1
1	0	0
1	1	1

→ Monotonic or Not ?

$f(a,b)$

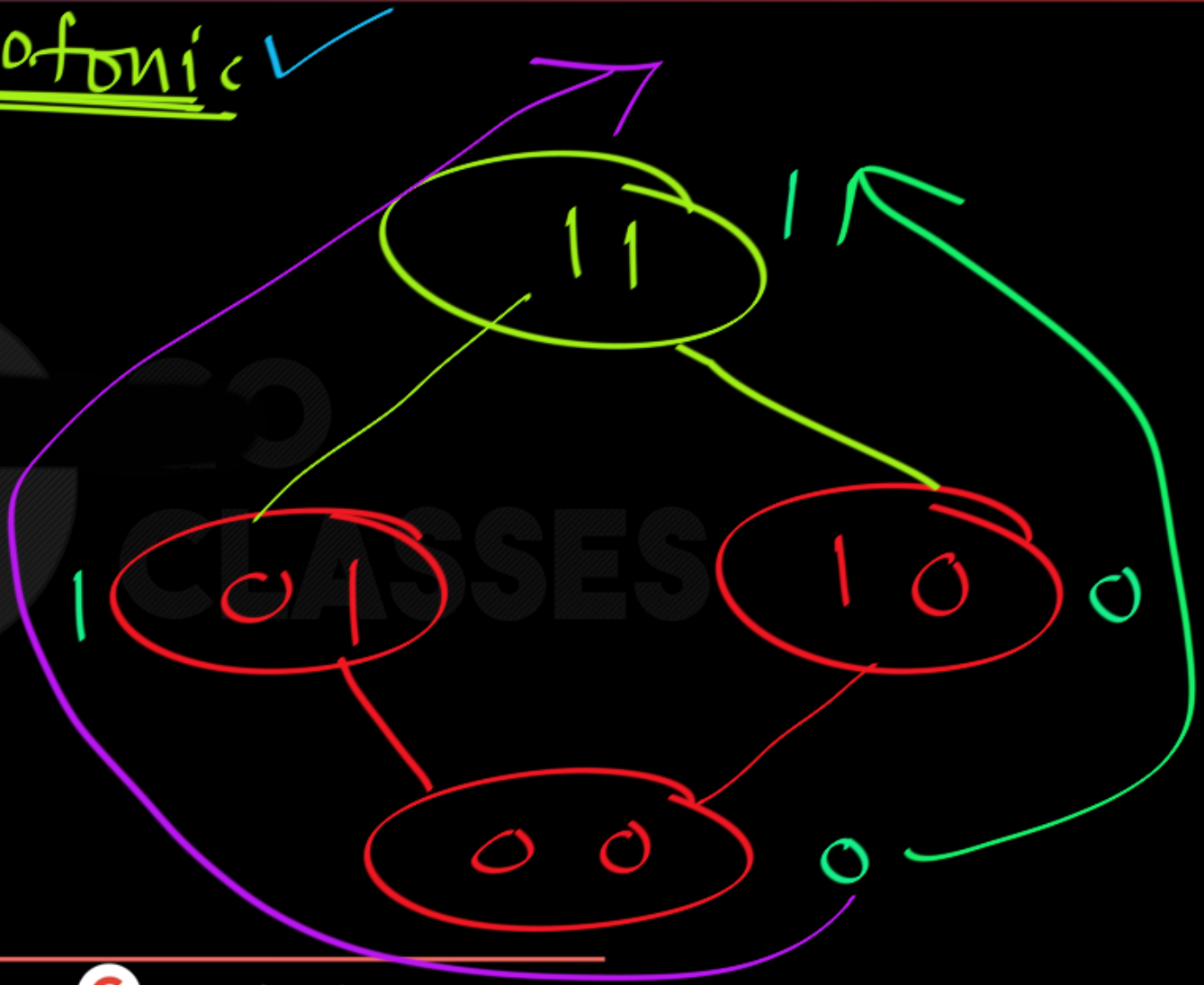
a	b	f
0	0	0
0	1	1
1	0	0
1	1	1



$f(a,b)$

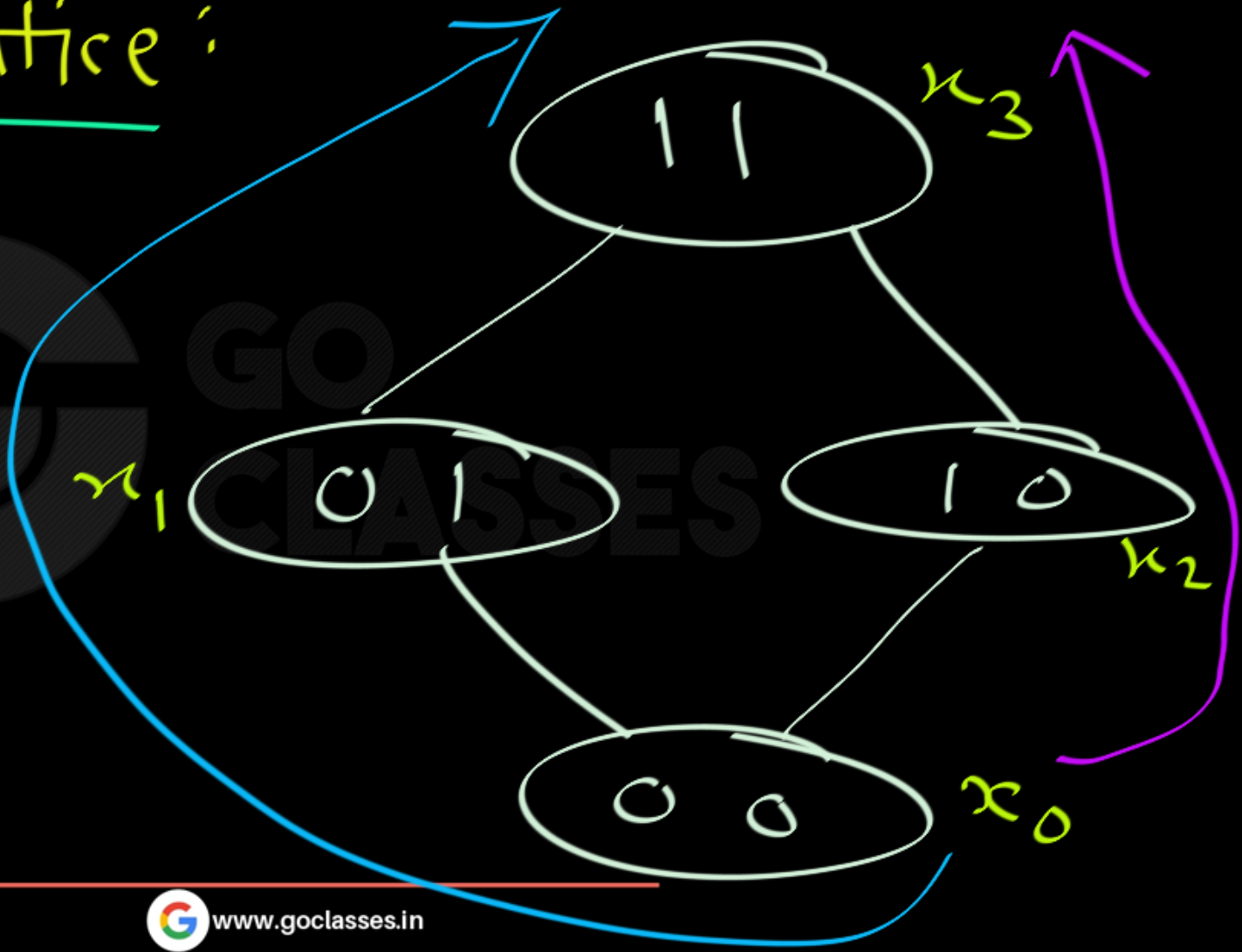
Monotonic ✓

a	b	f
0	0	0
0	1	1
1	0	0
1	1	1

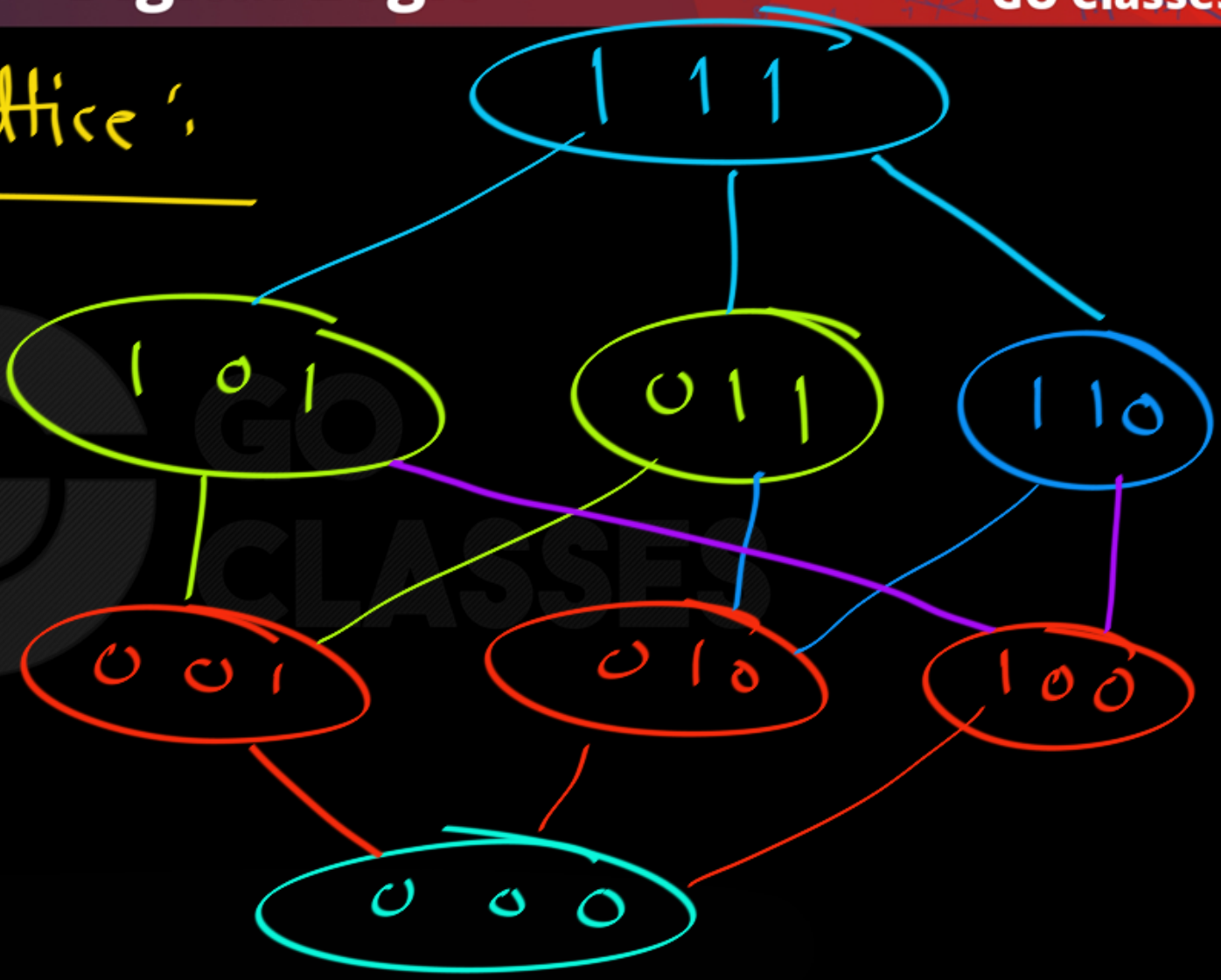


f(a, b) Lattice :

a	b	f
0	0	x_0
0	1	x_1
1	0	x_2
1	1	x_3



f(a,b,c) Lattice:



Property 4: We say that boolean function f is *monotone* if for every input, switching any input variable from 0 to 1 can *only* result in the function's switching its value from 0 to 1, and *never* from 1 to 0. We denote the class of monotone boolean functions with M and write $f \in M$.

Yet the given definition looks slightly intimidating, it is actually very easy to check monotonicity by looking at the following *Hasse diagrams*:

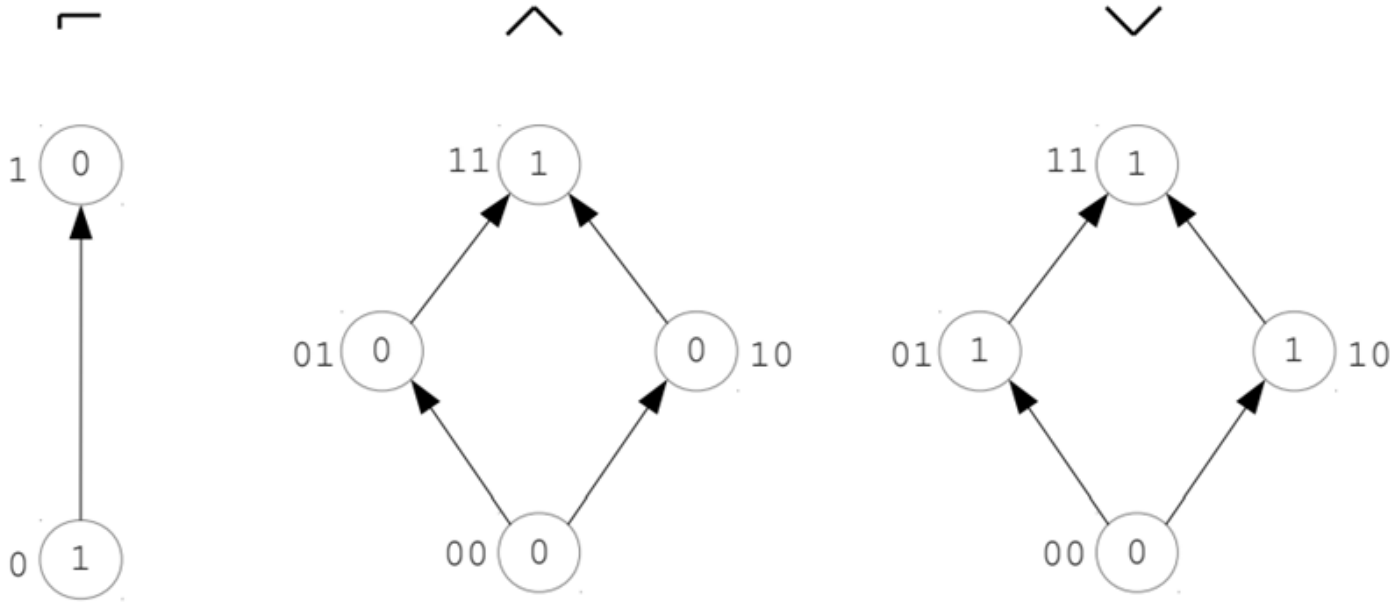


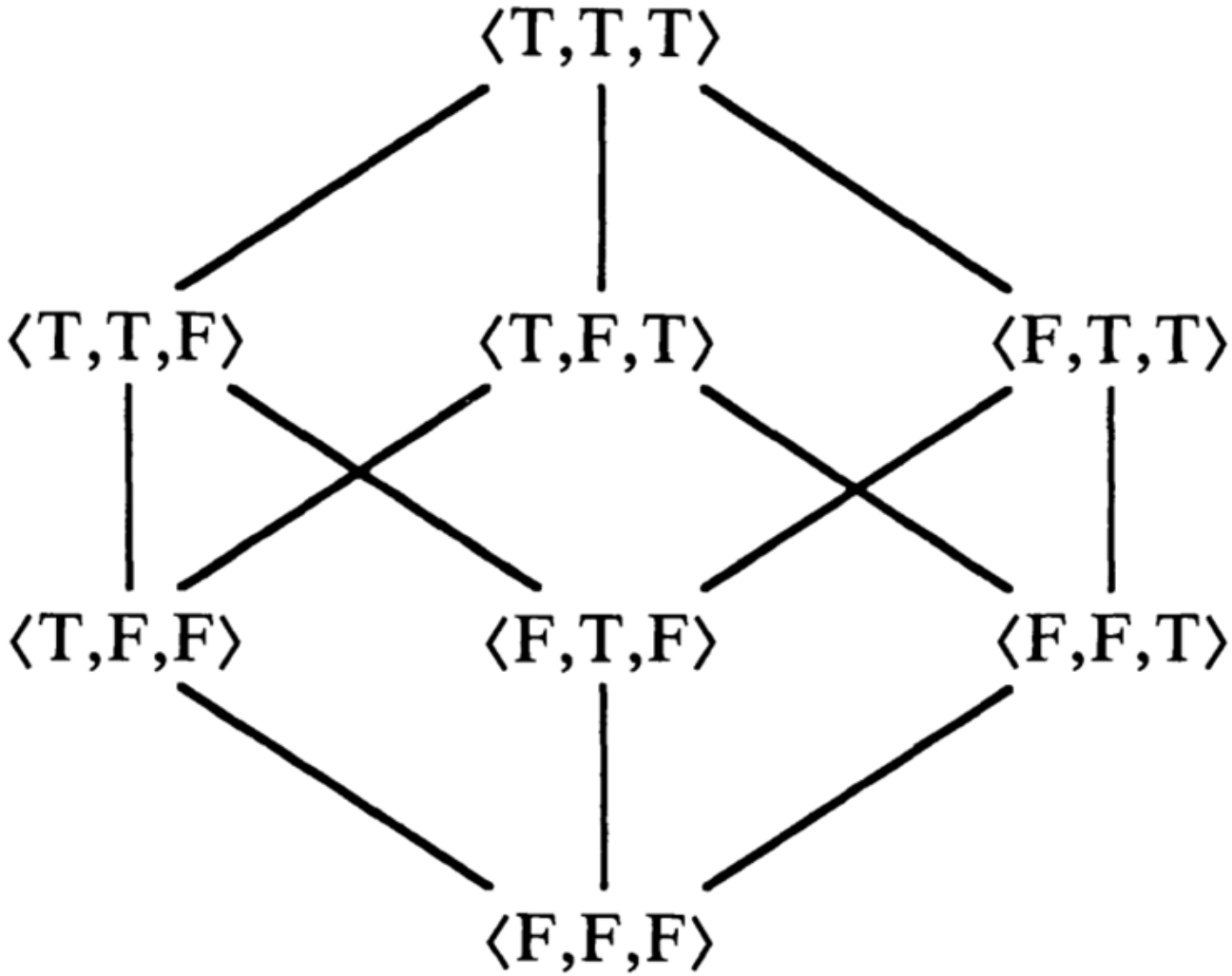
Figure 1: Checking monotonicity with Hasse diagrams

Note: Constant functions 0, 1 are monotonic.

function value can never Decrease.

fixed

Note: Constant functions 0, 1 are monotonic.


⟨T,T,T⟩
⟨T,T,F⟩
⟨T,F,T⟩
⟨F,T,T⟩
⟨T,F,F⟩
⟨F,T,F⟩
⟨F,F,T⟩
⟨F,F,F⟩

Functional Completeness (FC)

Next:

Post's Functional Completeness Theorem (Revisited)

Post's THEOREM

to determine if a Set of Boolean Functions is Functionally Complete Or Not:

Let X be a Set of Boolean Functions.

Theorem. *A system of boolean functions is functionally complete if and only if this system does not* entirely *belong to any of* T_0, T_1, L, M, S.

The mentioned theorem is called *Post's completeness criterion* and is due to Emil Post. What this criterion says is that in order for a system of boolean functions to be functionally complete, this system should have

- at least one function that does *not* preserve zero (i.e. it is *not* in T_0), and
- at least one function that does *not* preserve one (i.e. it is *not* in T_1), and
- at least one function that is *not* linear (i.e. it is *not* in L), and
- at least one function that is *not* monotone (i.e. it is *not* in M), and
- at least one function that is *not* self-dual (i.e. it is *not* in S).

$X = \{ f_1, f_2, f_3, f_4 \}$

(1) If All functions f_i are 0-preserving then X is NOT fc.

$X = \{ f_1, f_2, f_3, f_4 \}$

(2) If All functions f_i are 1-preserving then X is NOT fc.

$X = \{ f_1, f_2, f_3, f_4 \}$

(3) If all function f_i are Self Dual then X is Not FC

$X = \{ f_1, f_2, f_3, f_4 \}$

(4) If all functions f_i are Linear then X is NOT FC

$X = \{ f_1, f_2, f_3, f_4 \}$

(5) If all functions f_i are monotonic then X is Not Fc

$X = \{ f_1, f_2, f_3, f_4 \}$

Else

X is FC.

Theorem (Post's Functional Completeness Theorem) *A set X of truth functions (of 2-valued logic) is functionally complete if and only if, for each of the five defined classes, there is a member of X which does not belong to that class.*

Functional Completeness (FC)

Next:

Some Examples

Checking if a Set of Boolean Functions is

Functionally Complete Or Not

Functional Completeness (FC)

Some Examples

Example 1:

$$\{x \Leftrightarrow \bar{y}, xy, 0\}$$

Using the Post's Functional Completeness Theorem, check out if the given system of Boolean functions is complete, by checking their belonging to the following 5 classes:

1) Class $\mathbf{C_0}$ = The class of all 0-preserving functions;
2) Class $\mathbf{C_1}$ = The class of all 1-preserving functions;
3) Class **L** = The class of all linear functions;
4) Class **S** = The class of all self-dual functions;
5) Class **M** = The class of all monotonic functions.

The belonging / not-belonging must be explained!

$S = \{ x \leftrightarrow \bar{y},\ xy,\ 0 \}$

$x \oplus y = x \odot \bar{y}$

$a \oplus b = \bar{a} \odot b$

$\leftrightarrow$ is $\odot$

$a \leftrightarrow \bar{b} \equiv a \odot \bar{b} \equiv a \oplus b$

$S = \{ x \leftrightarrow \bar{y},\ xy,\ 0 \}$

$S = \{ x \oplus y,\ xy,\ 0 \}$

Property 1 : 0-preserving :

$x \oplus y$ ↘

xy ↘

0 ↘

Every function in S is 0-preserving,

So ; S is

NOT FC.

Functional Completeness (FC)

Some Examples

Example 2:

Problem 8. (4 points)

Consider the logical operator $p \downarrow q$ defined

p	q	$p \downarrow q$
1	1	0
1	0	0
0	1	0
0	0	1

Show that $\{\downarrow\}$ forms a *functionally complete* collection of logical operators.

Problem 8. (4 points)

Consider the logical operator $p \downarrow q$ defined

NOR

p	q	$p \downarrow q$
1	1	0
1	0	0
0	1	0
0	0	1

Show that $\{\downarrow\}$ forms a *functionally complete* collection of logical operators.

$\{\downarrow\} = \{f = a \downarrow b\}$

"NOR" function is FC; So 'NOR' is called Universal function.

NOR gate : a universal gate

NAND gate : " " "

Prove that $\downarrow$ is fc using Post's theorem. $X = \{a \downarrow b\}$

(1) 0-preserving ; ⟶ No.

$f_{(a,b)} = a \downarrow b \Rightarrow f(0,0) \neq 0$

$0 \downarrow 0 = \overline{0+0} = 1$

Prove that $\downarrow$ is fc using Post's theorem. $X = \{a \downarrow b\}$

(2) 1-preserving; ⟶ No

$f(1,1) \neq 1$

$1 \downarrow 1 = \overline{1+1} = 0$

Prove that $\downarrow$ is fc using Post's theorem. $X = \{a \downarrow b\}$

(3) Self Dual :⟶ No.

$f = a \downarrow b = \overline{a + b} = \bar{a}\,\bar{b}$

$f^d = \bar{a} + \bar{b}$

$f \neq f^d$

Prove that $\downarrow$ is fc using Post's theorem. $X = \{a \downarrow b\}$

(4) Linear :⟶ No.

a	b	$a \downarrow b$
0	0	1
0	1	0
1	0	0
1	1	0

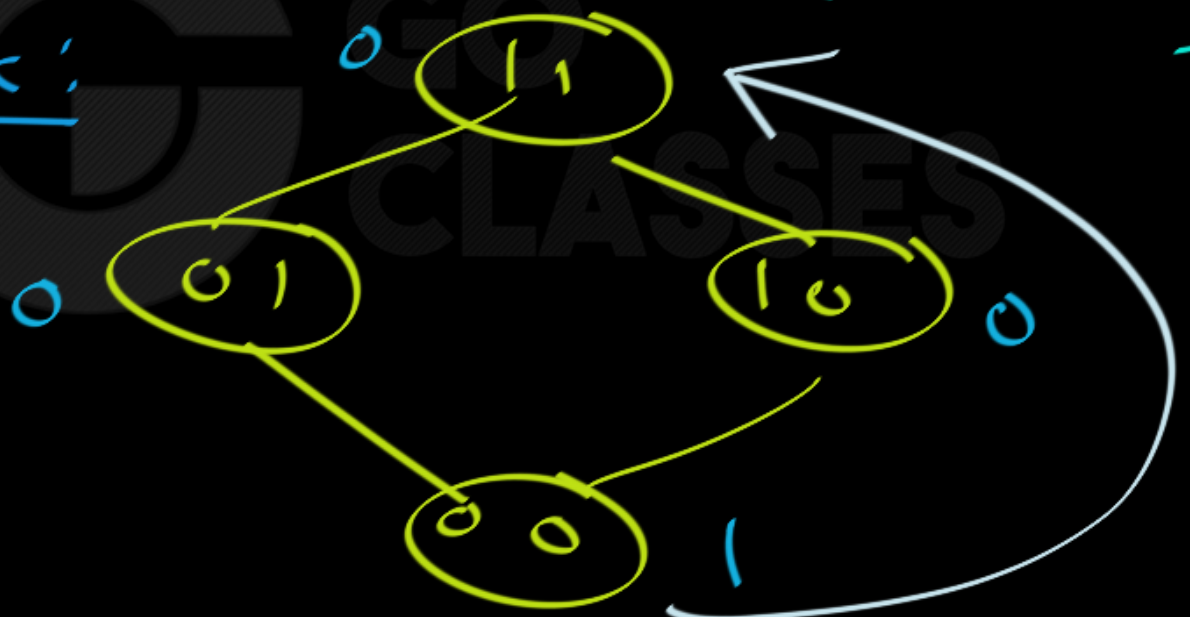
for $f = 1$ Rows ⟶ Even 1's in input

for $f = 0$ Rows ⟶ odd 1's in input

Violation

Prove that $\downarrow$ is fc using Post's theorem. $X = \{a \downarrow b\}$

(5) Monotonic:



No.

$X = \{ \downarrow \}$ ⟶ FC ✓

In X,

Some function ($\downarrow$) is not 0-pre ✗

" " " " 1-pre ✗

" " " " self Dual ✗

" " " " Linear ✗

" " " " monotonic ✗

Functional Completeness (FC)

Some Examples

Example 3:

$$\{x + \bar{y}, x \downarrow y, 1\}$$

Using the Post's Functional Completeness Theorem, check out if the given system of Boolean functions is complete,

by checking their belonging to the following 5 classes:

1) Class C, = The class of all 0-preserving functions;
2) Class C1 = The class of all 1-preserving functions;
3) Class L = The class of all linear functions;
4) Class S = The class of all self-dual functions;
5) Class M = The class of all monotonic functions.

$S = \{ x + \bar{y},\ x \downarrow y,\ 1 \}$

$x \downarrow y$ itself is Fnoupn.

So; S is fc.

Functional Completeness (FC)

Some Examples

Example 4:

Definition 1. A set S of logical connectives is said to be *functionally* complete if all boolean functions on the primitive propositions $P_1, P_2, \ldots, P_n$ can be represented by propositions using only connectives in S.

- Prove that the set of logical connectives $\{\neg, \wedge, \vee\}$ is functionally complete.
- Define the connective $\uparrow$ by the following truth table:

P	Q	$P \uparrow Q$
F	F	T
F	T	T
T	F	T
T	T	F

Prove that $\{\uparrow\}$ is functionally complete.

Definition 1. A set S of logical connectives is said to be *functionally* complete if all boolean functions on the primitive propositions $P_1, P_2, \ldots, P_n$ can be represented by propositions using only connectives in S.

- Prove that the set of logical connectives $\{\neg, \wedge, \vee\}$ is functionally complete. ✓ Already Done
- Define the connective $\uparrow$ by the following truth table:

NAND

P	Q	$P \uparrow Q$
F	F	T
F	T	T
T	F	T
T	T	F

Prove that $\{\uparrow\}$ is functionally complete.

Prove NAND is fc using Post theorem:

$a \uparrow b = \overline{ab} = \bar{a} + \bar{b}$

0-preserving	No
1-Preserving	No
Self Dual	$f = \bar{a} + \bar{b}$; $f^d = \bar{a}\bar{b}$; $f \neq f^d$; (No)
Linear	No
Monotoni.	No

Functional Completeness (FC)

Some Examples

Example 5:

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

4. The truth function $\otimes(\varphi_1, \varphi_2, \varphi_3)$ given by the table in problem (3) is functionally complete.

   a. Show that $\otimes$ obeys the requirements of Post's functional completeness theorem, thus guaranteeing its functional completeness.

   b. Assume you know that $\{\wedge, \vee, \neg\}, \{\wedge, \neg\}, \{\vee, \neg\}, \{\uparrow\}$ (NAND), and $\{\downarrow\}$ (NOR) are each a functionally complete set of connectives. Show that $\{\otimes\}$ is functionally complete, by reducing it to one of these already-known functionally complete sets of connectives. (I.e., show how to define the connectives in some functionally complete set using only the new connective).

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

4. The truth function $\otimes(\varphi_1, \varphi_2, \varphi_3)$ given by the table in problem (3) is functionally complete.

   a. Show that $\otimes$ obeys the requirements of Post's functional completeness theorem, thus guaranteeing its functional completeness.

(1)

0-preserving :

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3) = f$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

Not 0-preserving

$f(0, 0, 0) \neq 0$

(2)

1-preserving :

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3) = f$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

Not 1-preserving

$f(1,1) \neq 1$

(3)

self Dual :

↳ 140

Truth Table is Reverse of standard order

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3) = f$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

f

f	f^d	$\bar{f}$
0	1	1
1	1	1
0	0	0
1	0	1
0	1	0
1	0	1
1	1	1
1	0	0

f^d

(4)

Linear

↓

No.

Violation

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T ✓
T	F	F	F
F	T	T	T ✓
F	T	F	F
F	F	T	F
F	F	F	T ✓

for all f=1 Rows

→ Even 1's in input

for all f=0 Rows

→ Odd 1's in input

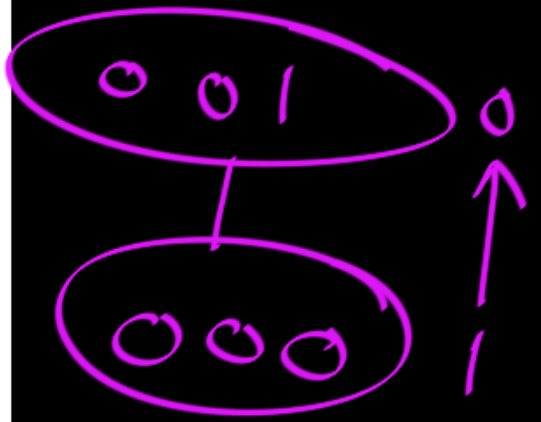
Violation

(5)

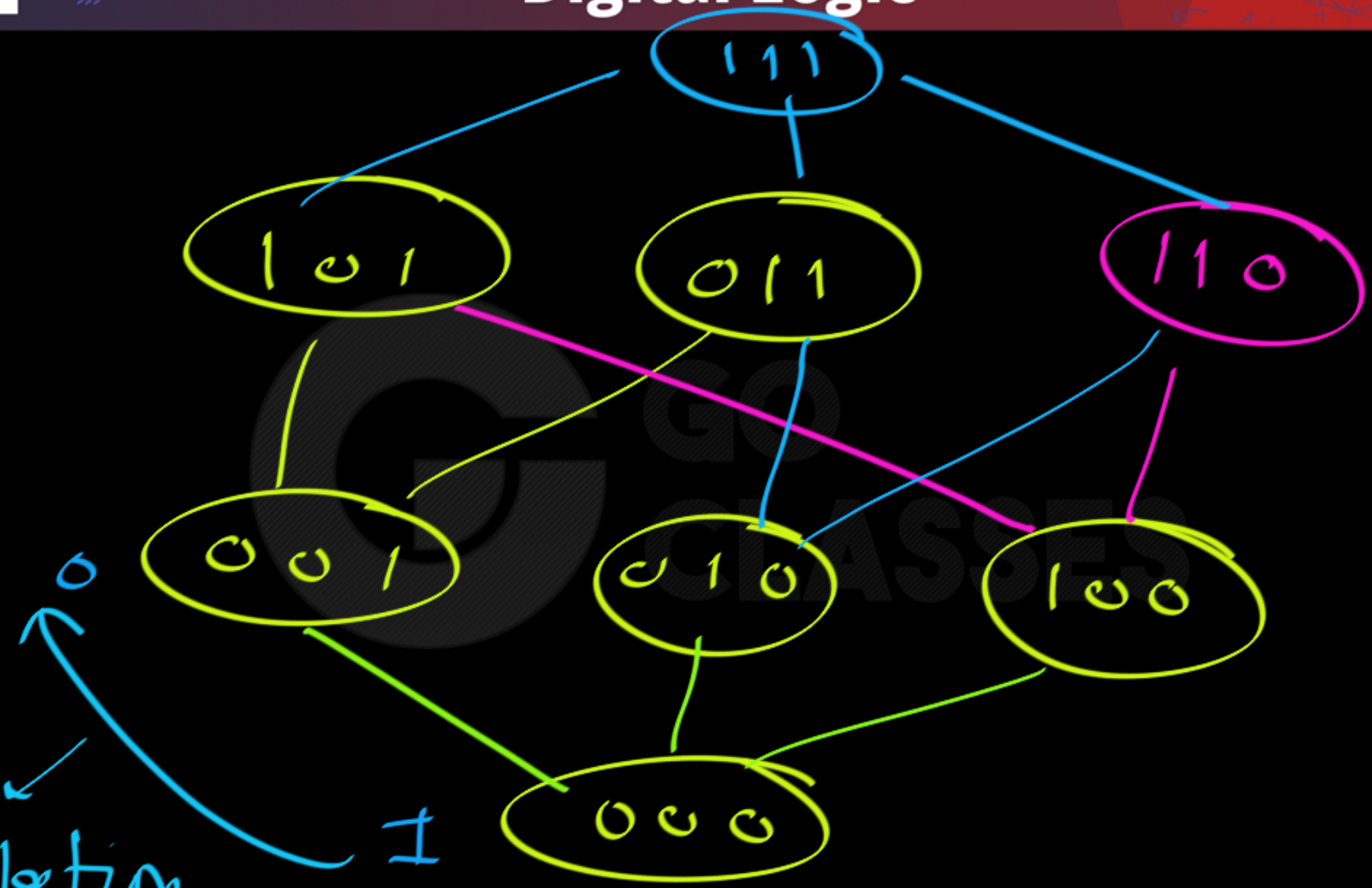
monotonic :

No

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T



Violation


111
1c1
c11
11c
cc1
c10
1cc
ccc
0
1
Violation

So; (X) function :

- NOT 0-preserving
- NOT 1-preserving
- NOT self Dual
- NOT Linear
- NOT Monotonic

So (X) is fc.

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

So FC

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

NOTE:
Before Checking Linear,
Always minimize then create Truth Table without Dummy Variables.

Note:

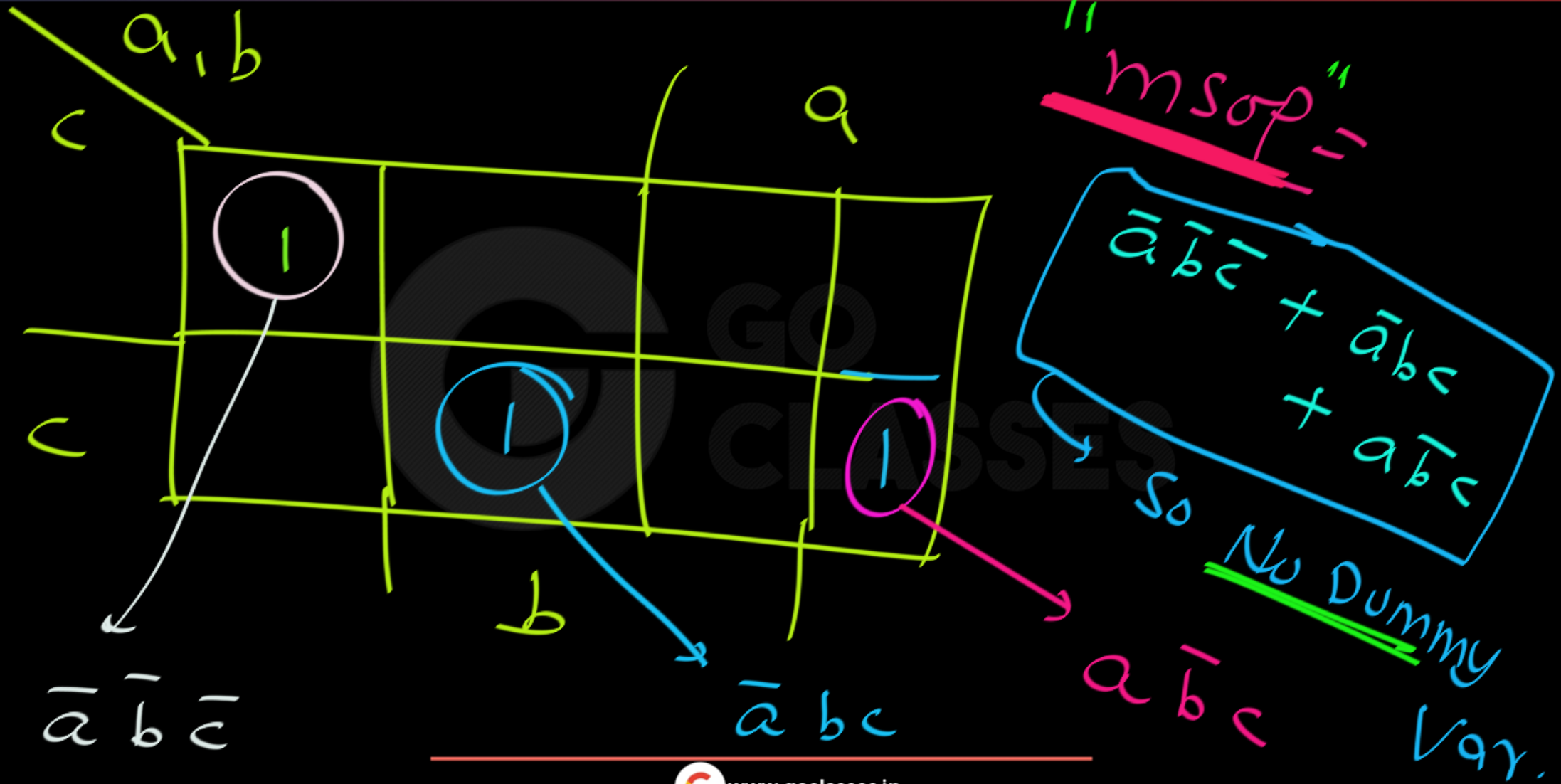
To check if f is Linear; we just have to make sure that there is NO Dummy variable.

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

5

3

0


a, b
c
a
c
1
1
1
b
$\bar{a}\bar{b}\bar{c}$
$\bar{a}bc$
$a\bar{b}c$
"msop" =
$\bar{a}\bar{b}\bar{c} + \bar{a}bc + a\bar{b}c$
So No Dummy Var.

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

Since No Dummy Variable in ⊗

So Now we can Linearity this Table.

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

FC

φ_1	φ_2	φ_3	$\otimes(\varphi_1, \varphi, \varphi_3)$
T	T	T	F
T	T	F	F
T	F	T	T
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	F
F	F	F	T

4. The truth function $\otimes(\varphi_1, \varphi_2, \varphi_3)$ given by the table in problem (3) is functionally complete.

   a. Show that $\otimes$ obeys the requirements of Post's functional completeness theorem, thus guaranteeing its functional completeness.

   b. Assume you know that $\{\wedge, \vee, \neg\}, \{\wedge, \neg\}, \{\vee, \neg\}, \{\uparrow\}$ (NAND), and $\{\downarrow\}$ (NOR) are each a functionally complete set of connectives. Show that $\{\otimes\}$ is functionally complete, by reducing it to one of these already-known functionally complete sets of connectives. (I.e., show how to define the connectives in some functionally complete set using only the new connective).

from ⊗ , if we can create

$\bar{a}$, ab ; then ⊗ is FC.

Note:

Given

$f(a,b,c) = \bar{a}\bar{b}\bar{c} + \bar{a}b\bar{c} + a\bar{b}\bar{c}$

↳ we can use.

We can use: f, a, b, c

We cannot use: $0, 1, \bar{a}, \bar{b}$

(X) ⟶ "minimized" SOP ;

(X) ≡ f

$f_{(a,b,c)} = \bar{a}\bar{b}\bar{c} + \bar{a}bc + a\bar{b}c$

Create $\bar{a}$;

$f(a,a,a) = \bar{a}$ ✓

So "NOT" is created.

Can use a, b, c

Cannot use 0, 1.

Now, since we have created "NOT" so we can use $\bar{a}$, $\bar{b}$, $\bar{c}$ now.

(X) ⟶ "minimized" SOP ;

(X) ≡ f

$$f_{(a,b,c)} = \bar{a}\bar{b}\bar{c} + \bar{a}bc + a\bar{b}c$$

Create ~~AND~~ NOR :

$$f(a,a,c) = \bar{a}\bar{a}\bar{c} + \bar{a}ac + a\bar{a}c$$

$$= \bar{a}\bar{c} = \overline{a+c}$$

NOR

(X) ⟶ "minimized" SOP ; (X) ≡ f

$f_{(a,b,c)} = \bar{a}\bar{b}\bar{c} + \bar{a}bc + a\bar{b}c$

we can create NOR so $f_{(a,b,c)}$ is FC.

$f(a,a,c) = \bar{a}\bar{c} = a \downarrow c$

So; this previous method is Basically Hit & Trial.

So; You can apply Post' fc Theorem.

No Hit & Trial.

To Check fc :

GoClasses Student "Toolbox":

(1) Post fc Theorem ✓

→ Advices to use.

or

(2) This Hit & Trial ✓

Functional Completeness (FC)

Some Examples

Example 6:

1. Is the following set of connectives functionally complete? Justify your answer.

$$\{\wedge, \vee, \rightarrow, \leftrightarrow\}$$

1. Is the following set of connectives functionally complete? Justify your answer.

$\{\wedge, \vee, \rightarrow, \leftrightarrow\}$

No.

If You Try Hit & Trial

& You Try to create {NOT, AND}

then You will "Never be Sure!!"

1. Is the following set of connectives functionally complete? Justify your answer.

$$\{\wedge, \vee, \rightarrow, \leftrightarrow\}$$

(1) 0-preserving:

$\wedge$ ✓

$\vee$ ✓

$\rightarrow$ ✗

$\leftrightarrow$ ✗

At least one function is NOT 0-preserving.

1. Is the following set of connectives functionally complete? Justify your answer.

$S = \{\wedge, \vee, \rightarrow, \leftrightarrow\}$

(2) 1 – preserving :

$\wedge$ 1

$\vee$ 1

$\rightarrow$ 1

$\leftrightarrow$ 1

All functions are 1-preserving

S is NOT FC

1. Is the following set of connectives functionally complete? Justify your answer.

$$\{\wedge, \vee, \rightarrow, \leftrightarrow\}$$

If You Try to create "NOT" $\rightarrow \bar{a}$

Best of Luck.

Because You will NEVER be Sure.

Functional Completeness (FC)

Some Examples

Example 7:

Question 6 (1 point) ✓ *Saved*

Which of the following sets of connectives are functionally complete?

[redacted]

$\neg, \vee$

$\rightarrow, \neg, \wedge$

$\rightarrow, \leftrightarrow, \vee$

$\wedge, \vee$

Question 6 (1 point) ✓ *Saved*

Which of the following sets of connectives are functionally complete?

- [x] $\neg, \vee$ — FC
- [x] $\rightarrow, \neg, \wedge$ — FC
- [ ] $\rightarrow, \leftrightarrow, \vee$ — All 1-preserving — NOT FC
- [ ] $\wedge, \vee$ — all 0-preserving — NOT FC

ISI2017-PCB-CS-7-a

asked in **Digital Logic** Sep 20, 2018 • edited Nov 8, 2019 by **go_editor**

378 views

1

Show that $\{1, A\bar{B}\}$ is functionality complete, i.e., any Boolean function with variables A and B can be expressed using these two primitives.

isi2017-pcb-cs digital-logic functional-completeness descriptive

$S = \{ Y, A\overline{B} \}$ $\quad f(A, B) = A\overline{B}$

① method 1 : Create NOT ($\overline{B}$) ; AND (A, B)

Create $\overline{B}$:

$f(1, B) = \overline{B}$ ⟶ So we can Create NOT.

$S = \{1, A\overline{B}\}$ $f(A,B) = A\overline{B}$

① method 1 : Create NOT ($\overline{a}$) ;

Create AND ;

$f(A, f(1,B)) = AB$ → So AND created.

$\overline{B}$

$S = \{1, A\overline{B}\}$ → F_{CL}

$\{NOT, AND\}$ can be created → F_C

$S = \{1, A\bar{B}\}$

Method 2 : Post's FC Theorem :

(1) 0-preserving :

	$A\bar{B}$
1	✗

So in S, some function is NOT 0-preserving.

$S = \{1, A\bar{B}\}$

Method 2 : Post's FC Theorem :

(2) 1-preserving :

	$A\bar{B}$
1	✓

So in S, some function is NOT 1-preserving.

$S = \{1, A\bar{B}\}$

Method 2 : Post's Fc Theorem :

(3) Self Dual :

1 ✗

$A\bar{B}$ ✗

At least function in S is NOT self Dual

$f = 1$

$f^d = 0$

$g = A\bar{B}$

$g^d = A + \bar{B}$

$S = \{1, A\bar{B}\}$ → Already minimized

Constant function is Linear.

→ No Dummy variable.

Method 2 :

(4) Linear :

1 ✓

$A\bar{B}$ ✗

A	B	$A\bar{B}$
0	0	0
0	1	0
1	0	1
1	1	0

→ Not Linear

f = 0 Rows

Sometime even 1's in input

Sometimes odd 1's in input

$f = A\bar{B}$ is NOT Linear.

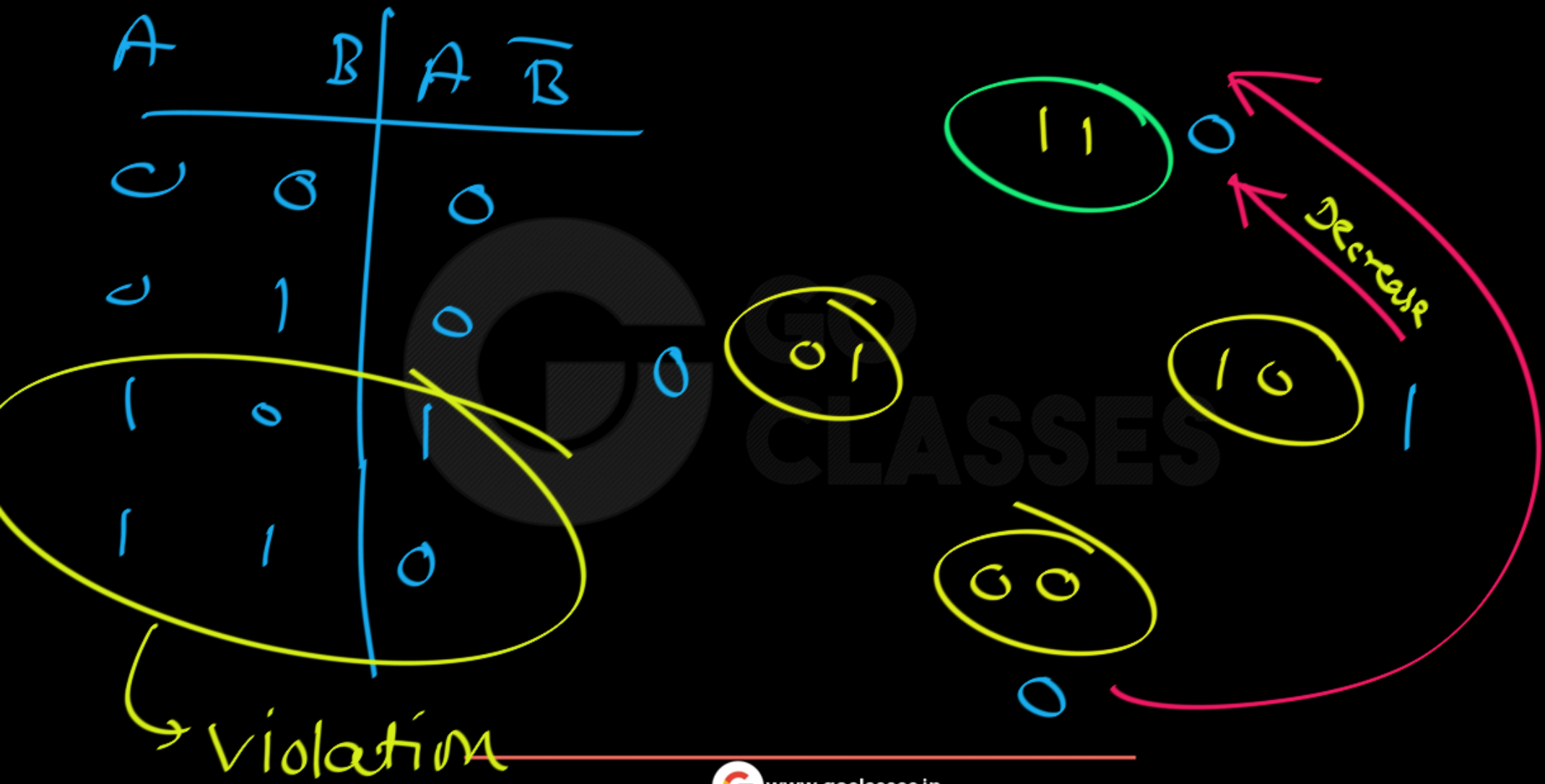
$S = \{1, A\bar{B}\}$

Method 2 : Post's Fc Theorem :

(5) Monotonic :

1 ✓

$A\bar{B}$ ✗ → At least one function is S is NOT monotone.


A
B
$A\ \overline{B}$
0 0 0
0 1 0
1 0 1
1 1 0
Violation
11 0
01
10 1
00 0
Decrease

$S = \{1, A\overline{B}\}$ ⟶ Fc

Some function is NOT 0-preserving : 1

" " " NOT 1-preserving : $A\overline{B}$

" " " NOT self Dual : $1, A\overline{B}$

" " " NOT Linear : $A\overline{B}$

" " " NOT monotonic : $A\overline{B}$

Functional Completeness (FC)

Some Examples

Example 8:

The operations $\Longrightarrow$ and $\textcircled{\scriptsize D}$ are shown in the table below. Show that the set of operations $\{\textcircled{\scriptsize D}, \Longrightarrow\}$ is functionally complete.

x	y	$x \Longrightarrow y$	$x \textcircled{\scriptsize D} y$
0	0	1	0
0	1	1	1
1	0	0	1
1	1	1	0

Functional Completeness (FC)

Next:

A Misconception:

"Partially Functionally Complete"

Functional Completeness (FC)

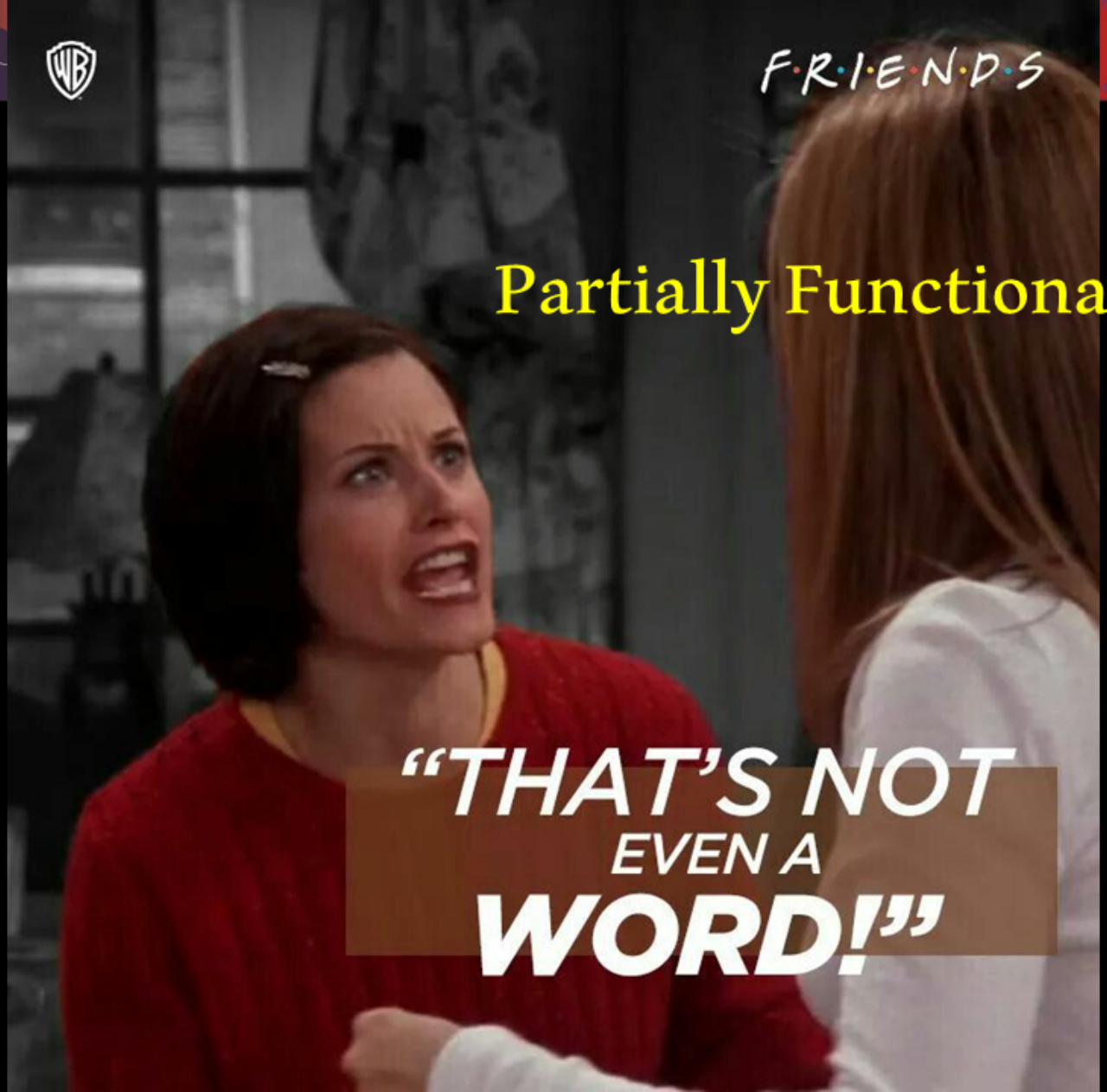
Next:

- A Misconception: Partially Functionally Complete
- Boolean Functions Vs Digital Circuits

Functional Completeness (FC):

A Misconception:

"Partially Functionally Complete"



Partially Functionally Complete

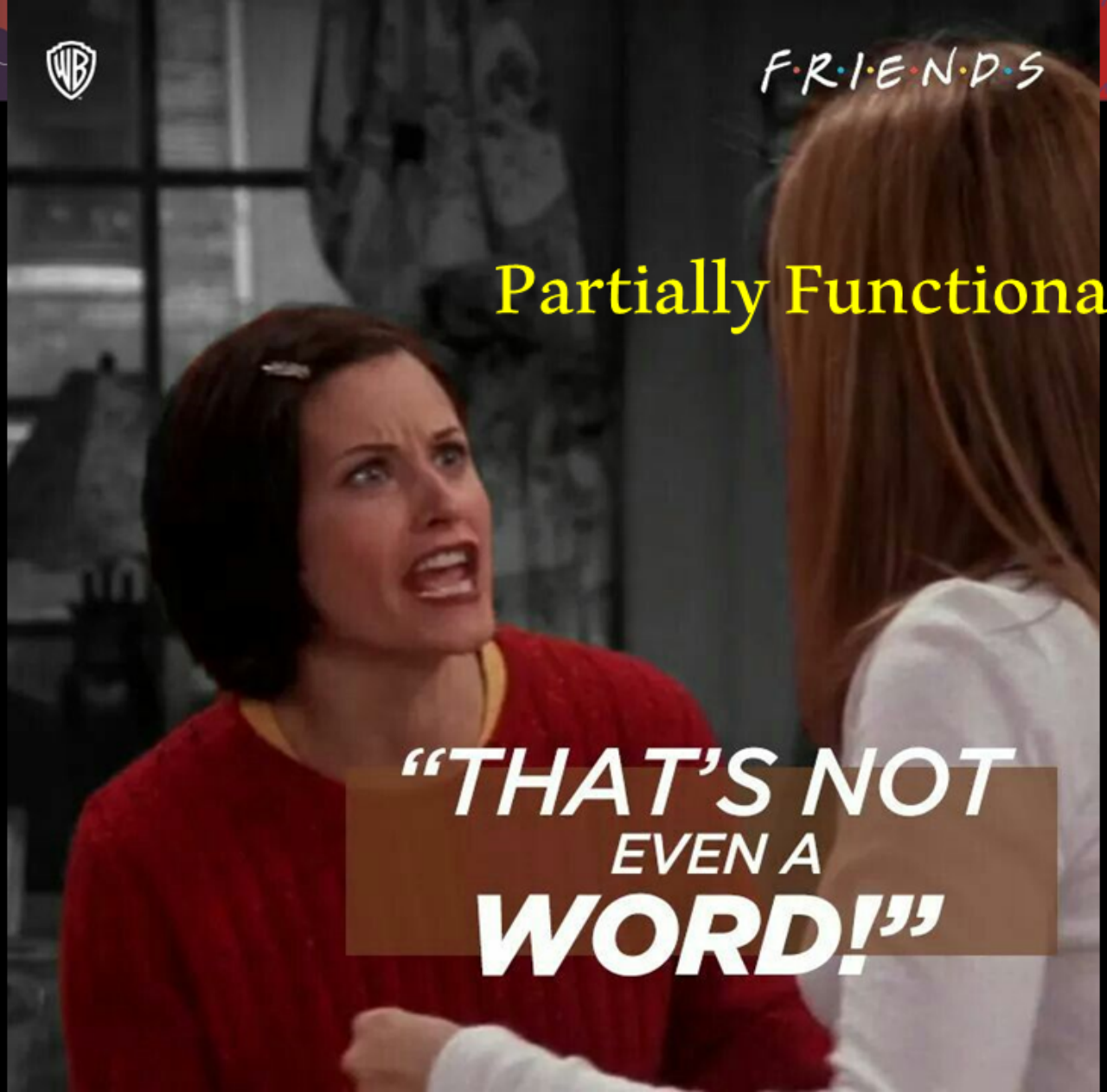
Partially Functionally Complete:

There is NO "Standard" Resource Containing such Concept.

Partially Functionally Complete:

There is NO "Standard" Resource Containing such Concept.

"Partially Functionally Complete" is a Concept MADE BY Some Indian Coaching Teachers & Copied by others Coaching Teachers, Instead of following Standard Resources.



Partially Functionally Complete

Functional Completeness (FC):

An Important Note:

Boolean Functions Vs Digital Circuits

Boolean functions

$a+b$ (OR)

$a+b$, $a.b$, $a \oplus b$, $a \odot b$, $\bar{a}$, $a \rightarrow b$, a, $a \uparrow b$, $a \downarrow b$

Digital Circuits

OR gate

gates, mux, Decoder, Encoder

Boolean functions

By Default, You Can not use 0, 1 unless it is given to us.

Digital Circuits

By Default, 0, 1 signals are Available to us.

Boolean function

implication function

$f(a,b) = a \rightarrow b = \bar{a} + b$

Digital Circuit

implication gate

Which of them is functionally Complete ??

Boolean function

implication function

$f(a,b) = a \rightarrow b = \bar{a} + b$

$X = \{ a \rightarrow b \}$

↳ NOT FC

Digital Circuit

implication gate

$X = \{ \bar{a} + b, 0, 1 \}$

↳ FCL

$\{a \rightarrow b, 0, 1\}$ is fc.

Create NOT :

$a \rightarrow 0 \equiv \bar{a}$

Create OR :

$\bar{a} \rightarrow b \equiv a + b$

$X = \{ a \rightarrow b \}$ is NOT fc.

1-preserving

NOT fc.

Boolean functions

→ By Default, we Can't use 0, 1. Unless Already Given.

Digital Circuits

→ By Default, 0, 1 signals are Available.

Boolean functions

→ By Default, we Can't use 0, 1. Unless Already Given.

Digital Circuits

→ By Default, 0, 1 signals are Available.

Source : Charles Roth Book ; & GATE PYQs.

A set of logic operations is said to be *functionally complete* if any Boolean function can be expressed in terms of this set of operations. The set AND, OR, and NOT is obviously functionally complete because any function can be expressed in sum-of-products form, and a sum-of-products expression uses only the AND, OR, and NOT operations. Similarly, a set of logic gates is functionally complete if all switching functions can be realized using this set of gates. Because the set of operations AND, OR, and NOT is functionally complete, any set of logic gates which can realize AND, OR, and NOT is also functionally complete. AND and NOT are a functionally complete set of gates because OR can also be realized using AND and NOT:

[redacted]

(We will always assume that 0 and 1 are available as gate inputs). Next, attempt to [redacted]

Source: Fundamentals of Logic Design, Charles H. Roth, Jr.

The following procedure can be used to determine if a given set of gates is functionally complete. First, write out a minimum sum-of-products expression for the function realized by each gate. If no complement appears in any of these expressions, then NOT cannot be realized, and the set is not functionally complete. If a complement appears in one of the expressions, then NOT can generally be realized by an appropriate choice of inputs to the corresponding gate. (We will always assume that 0 and 1 are available as gate inputs). Next, attempt to realize AND or OR, keeping in mind that NOT is now available. Once AND or

Source: Fundamentals of Logic Design, Charles H. Roth, Jr.

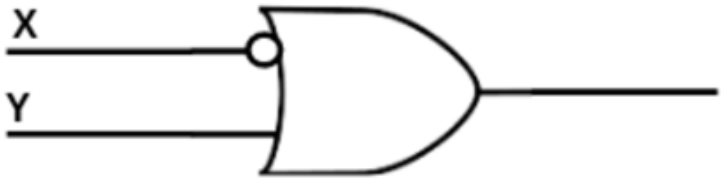
GATE CSE 1998 | Question: 5

asked in Digital Logic Sep 26, 2014 • edited Apr 21, 2019 by Pooja Khatri

3,234 views

22

The implication gate, shown below has two inputs (x and y); the output is 1 except when $x = 1$ and $y = 0$, realize $f = \bar{x}y + x\bar{y}$ using only four implication gates.



Digital Circuit

Show that the implication gate is functionally complete.

Implication function is NOT fc.

$\{a \rightarrow b\}$

Implicate gate is fc.

$\{0, 1, a \rightarrow b\}$ → fc.

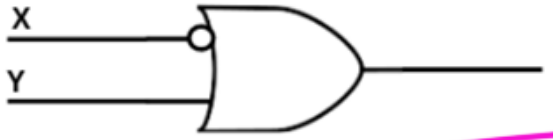
GATE CSE 1998 | Question: 5

asked in Digital Logic Sep 26, 2014 • edited Apr 21, 2019 by Pooja Khatri

3,234 views

22

The implication gate, shown below has two inputs (x and y); the output is 1 except when $x = 1$ and $y = 0$, realize $f = \bar{x}y + x\bar{y}$ using only four implication gates.



by Default 0, 1 Available

Show that the implication gate is functionally complete.

ISI2017-PCB-CS-7-a

asked in Digital Logic Sep 20, 2018 • edited Nov 8, 2019 by go_editor

378 views

1

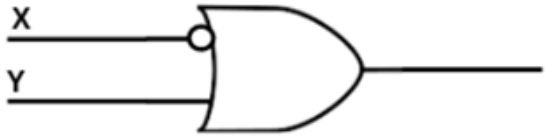
Show that $\{1, A\bar{B}\}$ is functionality complete, i.e., any Boolean function with variables A and B can be expressed using these two primitives.

GATE CSE 1998 | Question: 5

asked in **Digital Logic** Sep 26, 2014 • edited Apr 21, 2019 by **Pooja Khatri** 3,234 views

22

The implication gate, shown below has two inputs (x and y); the output is 1 except when $x = 1$ and $y = 0$, realize $f = \bar{x}y + x\bar{y}$ using only four implication gates.



Show that the implication gate is functionally complete.

ISI2017-PCB-CS-7-a

asked in **Digital Logic** Sep 20, 2018 • edited Nov 8, 2019 by **go_editor** 378 views

1

Show that $\{1, A\bar{B}\}$ is functionality complete, i.e., any Boolean function with variables A and B can be expressed using these two primitives.

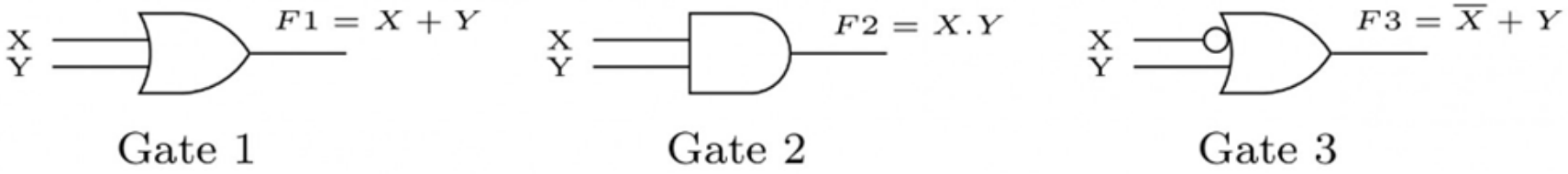
GATE ECE 2015 Set 3 | Question: 37

asked in **Number Representations** Mar 28, 2018 • recategorized Nov 27, 2022 by **gatecse**

112 views

2

A universal logic gate can implement any Boolean function by connecting sufficient number of them appropriately. Three gates are shown.



Which one of the following statements is TRUE?

A. Gate 1 is a universal gate
B. Gate 2 is a universal gate
C. Gate 3 is a universal gate
D. None of the gates shown is a universal gate

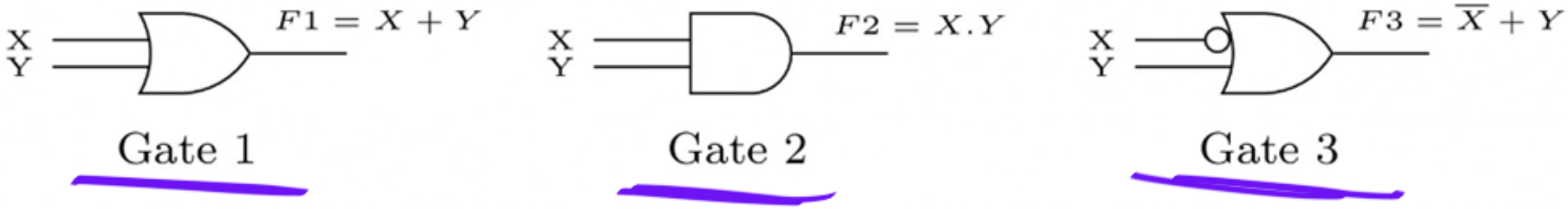
GATE ECE 2015 Set 3 | Question: 37

asked in **Number Representations** Mar 28, 2018 • recategorized Nov 27, 2022 by **gatecse**

112 views

2

A universal logic gate can implement any Boolean function by connecting sufficient number of them appropriately. Three gates are shown.



Which one of the following statements is TRUE?

A. Gate 1 is a universal gate
B. Gate 2 is a universal gate
C. Gate 3 is a universal gate
D. None of the gates shown is a universal gate

0, 1 signals are By default available.

$\{a+b, 0, 1\} \rightarrow$ NOT f_c.

$\{a_b, 0, 1\} \rightarrow$ '1' '1'.

$\{a \rightarrow b, 0, 1\} \rightarrow f_c$

GATE CSE 1999 | Question: 2.9

asked in **Digital Logic** Sep 23, 2014 • edited Nov 6, 2018 by **kenzou**

13,631 views

36

Which of the following sets of component(s) is/are sufficient to implement any arbitrary Boolean function?

A. XOR gates, NOT gates

B. 2 to 1 multiplexers

C. AND gates, XOR gates

D. Three-input gates that output $(A. B) + C$ for the inputs A, B and C.

gate1999 digital-logic normal functional-completeness multiple-selects

GATE CSE 1999 | Question: 2.9

HW → Post's FC Theorem.

asked in **Digital Logic** Sep 23, 2014 • edited Nov 6, 2018 by **kenzou**

13,631 views

36

Which of the following sets of component(s) is/are sufficient to implement any arbitrary Boolean function?

A. XOR gates, NOT gates , 0, 1

B. 2 to 1 multiplexers , 0, 1

C. AND gates, XOR gates , 0, 1

D. Three-input gates that output $(A.B) + C$ for the inputs A, B and C. , 0, 1

gate1999 digital-logic normal functional-completeness multiple-selects

2 * 1 mux is FC

True.

0, 1 Signals are by Default available in the inputs.

GATE CSE 1993 | Question: 9

asked in **Digital Logic** Sep 30, 2014 • retagged 1 day ago by **Deepak Poonia** 2,268 views

16

Assume that only half adders are available in your laboratory. Show that any binary function can be implemented using half adders only.

gate1993 digital-logic combinational-circuit adder descriptive functional-completeness

edit flag close hide answer
comment Follow Pip Box Lists
Mark Read Feature Hide Answers
Delete with Reason Bad

Kathleen

share this

2 Comments

Chhotu commented Dec 28, 2017

But it is partially functionally complete.

10

GATE CSE 1993 | Question: 9

asked in Digital Logic Sep 30, 2014 • retagged 1 day ago by Deepak Poonia

2,268 views

16

Assume that only half adders are available in your laboratory. Show that any binary function can be implemented using half adders only.

gate1993 digital-logic combinational-circuit adder descriptive functional-completeness

edit flag close hide answer

comment Follow Pip Box Lists

Mark Read Feature Hide Answers

Delete with Reason Bad

share this

Kathleen

2 Comments

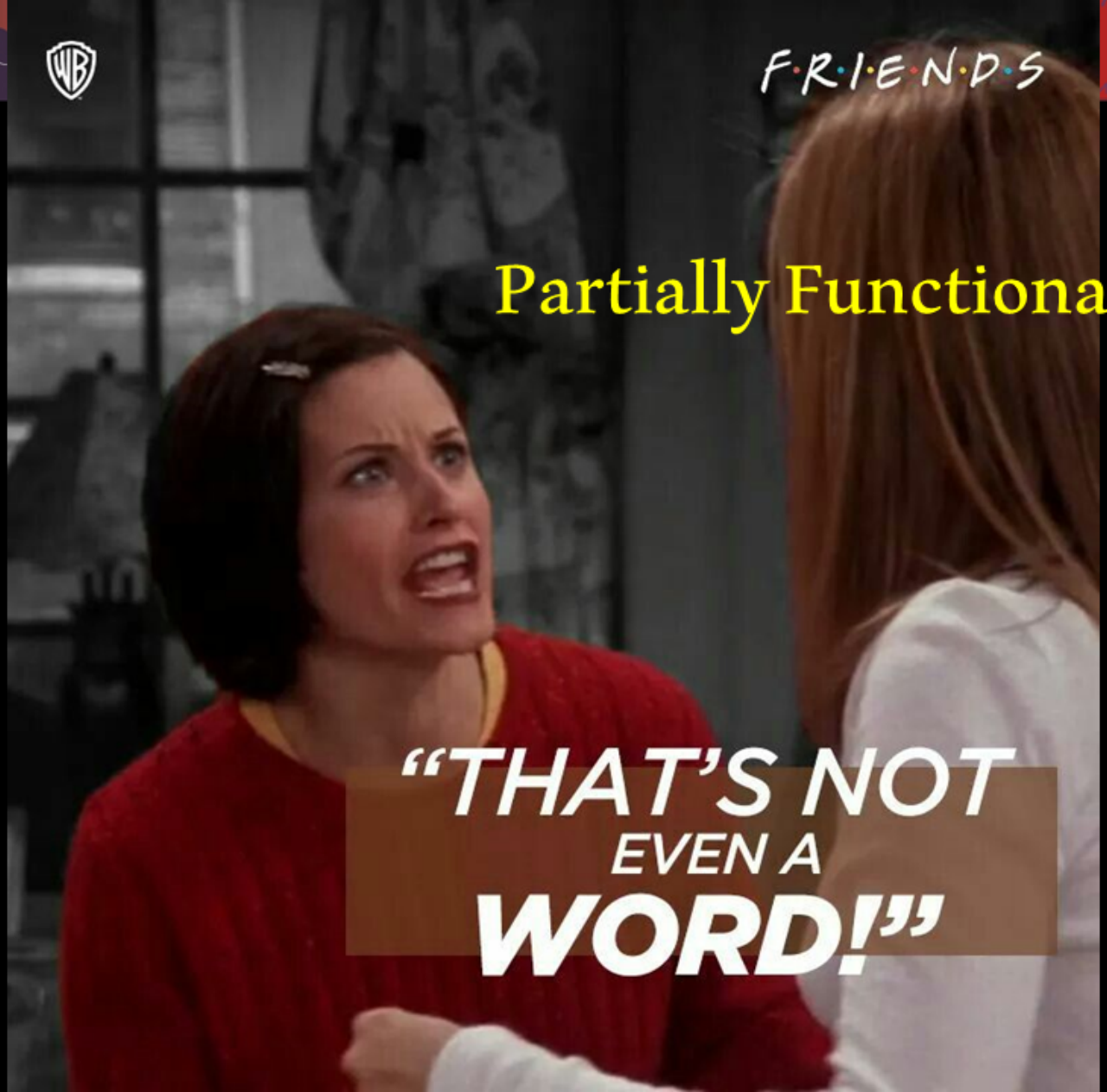
Chhotu commented Dec 28, 2017

But it is partially functionally complete.

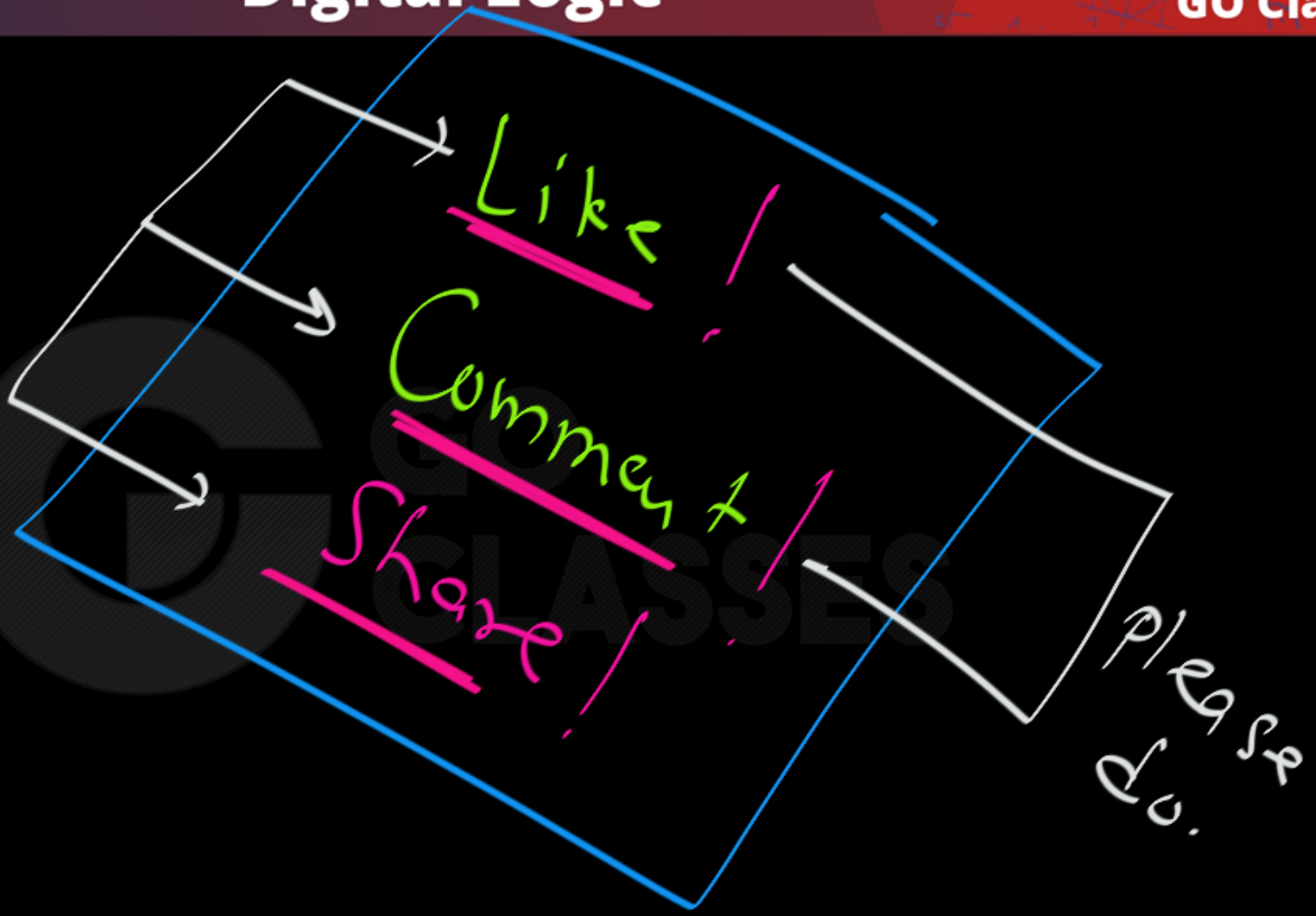
10

0,1 by default available.

my misconception



Partially Functionally Complete


Like !
Comment !
Share !
Please
Do.