

CNS 188a Overview

- Boolean algebra as an axiomatic system
- Boolean functions and their representations using Boolean formulas and spectral methods
- Implementing Boolean functions with **relay** circuits, circuits of **AON** (AND, OR, NOT) gates and **LT** (Linear Threshold) gates
- Analyzing the complexity (size and depth) of circuits
- Relations (as opposed to functions) and their implementation in circuits
- Feedback and convergence in LT circuits

Basic Axioms and Properties

- **A1. Identities:**

$$a + 0 = a \quad \text{and} \quad a \cdot 1 = a$$

- **A2. Complements:**

$$a + \bar{a} = 1 \quad \text{and} \quad a \cdot \bar{a} = 0$$

- **A3. Commutativity:**

$$a + b = b + a \quad \text{and} \quad a \cdot b = b \cdot a$$

- **A4. Distributivity:**

$$a + b \cdot c = (a + b) \cdot (a + c) \quad \text{and} \quad a \cdot (b + c) = a \cdot b + a \cdot c$$

- **L1. Self Absorption:**

$$a + a = a \quad \text{and} \quad a \cdot a = a$$

- **L2: Simple Absorption:**

$$a + 1 = 1 \quad \text{and} \quad a \cdot 0 = 0$$

- **T3. Associativity:**

$$(a + b) + c = a + (b + c)$$

$$(a \cdot b) \cdot c = a \cdot (b \cdot c)$$

- **T4. DeMorgan Laws:**

$$\overline{(a + b)} = \bar{a} \cdot \bar{b}$$

$$\overline{(a \cdot b)} = \bar{a} + \bar{b}$$

- **T0. Duality:**

Correctness is maintained when interchange $+$ and $\cdot$, as well as 0 and 1 .

- **T1. Distinct Complement:**

Every element has another element that is its unique complement.

- **T2. Absorption:**

$$a + ab = a \quad \text{and} \quad a \cdot (a + b) = a$$

Current State

Representation Theorem (Stone 1936):

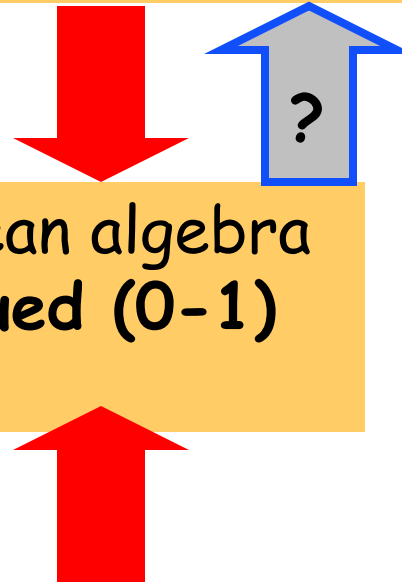
Every *finite* Boolean algebra is isomorphic to a Boolean algebra of subsets of some *finite* set S .

0-1 Theorem:

An identity is true for **any** finite Boolean algebra if and only if it is true for a **two-valued (0-1)** Boolean algebra.

DNF Representation Theorem:

DNF is a representation: two Boolean functions are equal if and only if their DNFs is identical.



DNF Theorem

DNF Theorem:

Every Boolean function can be expressed in DNF.

$$\begin{aligned}\overline{(a \cdot \bar{b} + a \cdot c)} + \bar{a} &= \overline{(a \cdot \bar{b})} \cdot \overline{(a \cdot c)} + \bar{a} \\ &= (\bar{a} + b)(\bar{a} + \bar{c}) + \bar{a} \\ &= \bar{a} + b \cdot \bar{c} + \bar{a} \\ &= \bar{a} + b \cdot \bar{c}\end{aligned}$$

Absorption:

- A2. Complements:
 $a + \bar{a} = 1$ and $a \cdot \bar{a} = 0$

- T4. DeMorgan Laws:

$$\begin{aligned}\overline{(a + b)} &= \bar{a} \cdot \bar{b} \\ \overline{(a \cdot b)} &= \bar{a} + \bar{b}\end{aligned}$$

- A4. Distributivity:

$$a + b \cdot c = (a + b) \cdot (a + c) \quad \text{and} \quad a \cdot (b + c) = a \cdot b + a \cdot c$$

$$\begin{aligned}&= \bar{a} \cdot (b + \bar{b}) \cdot (c + \bar{c}) + b \cdot \bar{c} \cdot (a + \bar{a}) \\ &= \bar{a} \cdot b \cdot c + \bar{a} \cdot b \cdot \bar{c} + \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot \bar{b} \cdot \bar{c} + a \cdot b \cdot \bar{c} + \bar{a} \cdot b \cdot \bar{c} \\ &= \bar{a} \cdot b \cdot c + \bar{a} \cdot b \cdot \bar{c} + \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot \bar{b} \cdot \bar{c} + a \cdot b \cdot \bar{c}\end{aligned}$$

DNF Theorem

DNF Theorem:

Every Boolean function can be expressed in DNF.

Proof: By the algorithm

Apply DeMorgan Theorem (T4) until each negation is applied to a single variable

Apply distributive axiom (A4) to get a sum of terms

Augment a missing variable a to a term using (A2) multiplying by $a + \bar{a}$

Use self absorption (L1) to eliminate duplicate terms

DNF from Look Up Tables

Idea: construct a DNF by adding the normal terms that correspond to $f=1$.

ab	XOR(a,b)
00	0
01	1
10	1
11	0

$$a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$

DNF from Look Up Tables

$\oplus$	$\emptyset$	$\{a\}$	$\{b\}$	$\{a, b\}$
$\emptyset$	$\emptyset$	$\{a\}$	$\{b\}$	$\{a, b\}$
$\{a\}$	$\{a\}$	$\emptyset$	$\{a, b\}$	$\{b\}$
$\{b\}$	$\{b\}$	$\{a, b\}$	$\emptyset$	$\{a\}$
$\{a, b\}$	$\{a, b\}$	$\{b\}$	$\{a\}$	$\emptyset$

Use only the **blue** entries

$$a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$

$$a_i^{x_i} = \begin{cases} a_i & \text{if } x_i = 1 \\ \bar{a}_i & \text{if } x_i = 0 \end{cases}$$

$$f(a_1, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(x_1, x_2, \dots, x_n) a_1^{x_1} a_2^{x_2} \cdots a_n^{x_n}$$

Boole (Shannon) Decomposition

$$f(a_1, a_2, \dots, a_n) = \bar{a}_1 \cdot f(0, a_2, \dots, a_n) + a_1 \cdot f(1, a_2, \dots, a_n)$$

Proof:

DNF representation

$$f(a_1, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(x_1, x_2, \dots, x_n) a_1^{x_1} a_2^{x_2} \cdots a_n^{x_n}$$

$0^{x_1} = 1 \rightarrow x_1 = 0$

$$f(0, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(x_1, x_2, \dots, x_n) 0^{x_1} a_2^{x_2} \cdots a_n^{x_n}$$

$$f(0, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(0, x_2, \dots, x_n) a_2^{x_2} \cdots a_n^{x_n}$$

Boole (Shannon) Decomposition

$$f(a_1, a_2, \dots, a_n) = \bar{a}_1 \cdot f(0, a_2, \dots, a_n) + a_1 \cdot f(1, a_2, \dots, a_n)$$

Proof:

DNF representation

$$f(a_1, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(x_1, x_2, \dots, x_n) a_1^{x_1} a_2^{x_2} \cdots a_n^{x_n}$$

$$f(0, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(0, x_2, \dots, x_n) a_2^{x_2} \cdots a_n^{x_n}$$

$$\bar{a}_1 \cdot f(0, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(0, x_2, \dots, x_n) \bar{a}_1 \cdot a_2^{x_2} \cdots a_n^{x_n}$$

DNF part that corresponds to $x_1 = 0$

Boole (Shannon) Decomposition

$$f(a_1, a_2, \dots, a_n) = \bar{a}_1 \cdot f(0, a_2, \dots, a_n) + a_1 \cdot f(1, a_2, \dots, a_n)$$

Proof:

DNF representation

$$f(a_1, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(x_1, x_2, \dots, x_n) a_1^{x_1} a_2^{x_2} \cdots a_n^{x_n}$$

$$\bar{a}_1 \cdot f(0, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(0, x_2, \dots, x_n) \bar{a}_1 \cdot a_2^{x_2} \cdots a_n^{x_n}$$

DNF part

$$x_1 = 0$$

$$a_1 \cdot f(1, a_2, \dots, a_n) = \sum_{X \in \{0,1\}^n} f(1, x_2, \dots, x_n) a_1 \cdot a_2^{x_2} \cdots a_n^{x_n}$$

DNF part

$$x_1 = 1$$

Q

Why is the Decomposition Important?

Idea: can help in compressing representations of Boolean functions (Decision Trees, Binary Decision Diagrams (BDD))

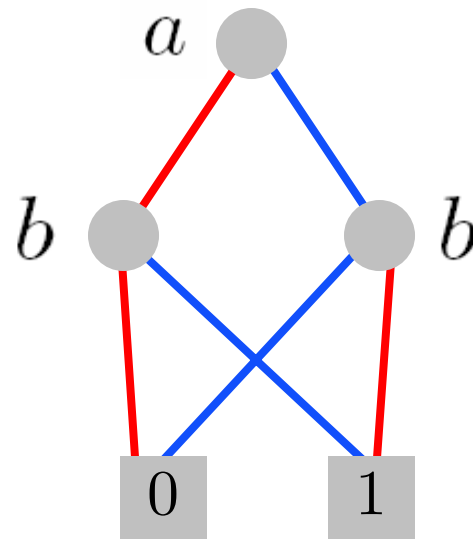
Key in design, optimization and verification of circuits.

ab	XOR(a,b)
00	0
01	1
10	1
11	0

0 = 

1 = 

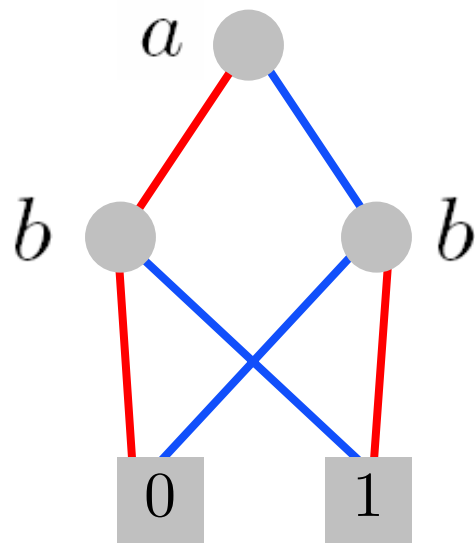
$$a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$



Why is the Decomposition Important?

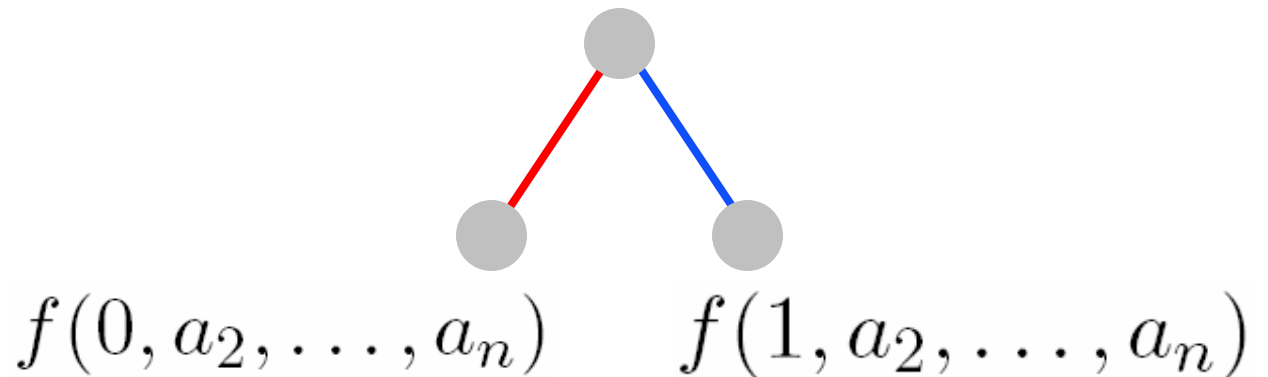
Decomposition results in a Binary Decision Tree

$$f(a_1, a_2, \dots, a_n) = \bar{a}_1 \cdot f(0, a_2, \dots, a_n) + a_1 \cdot f(1, a_2, \dots, a_n)$$



$0 =$ 
 $1 =$ 

$$f(a_1, a_2, \dots, a_n)$$



"The further back you look,
the further forward you can see"
Winston Churchill

Gottfried Leibniz
1646-1716



- Logic and binary system
- Calculus

Shannon
1916-2001



Boolean algebra to
electrical circuits
1938

George Boole
1815-1864



Connected logic
with algebra
1847

Edward Huntington
1874-1952



Boolean axiomatic
system, **1904**

Marshal Stone
1903-1989



Boolean
representation
theorem, **1936**

Gottfried Leibniz
1646-1716



Leibniz - Binary System

And accordingly, numbers are expressed as follows:

0	0
1	1
10	2
11	3
100	4
101	5
110	6
111	7
1000	8
1001	9
1010	10
1011	11
1100	12
1101	13
1110	14
1111	15

10000	16
10001	17
10010	18
10011	19
10100	20
10101	21
10110	22
10111	23
11000	24
11001	25
11010	26
11011	27
11100	28
11101	29
11110	30
11111	31
100000	32
etc.	etc

Gottfried Leibniz
1646-1716



Leibniz - Binary System

Binary addition algorithm

§72 As for addition, it is simply done by counting and making periods when there are numbers to add together, adding up each column as usual, which will be done as follows: count the unities of the column for example, for 29, look how this number is written in the table, to wit, by 11101; thus you write 1 under the column and put periods under the second, third and fourth column thereafter. These periods denote that it is necessary to count out one unity further in the column following.

Gottfried Leibniz
1646-1716

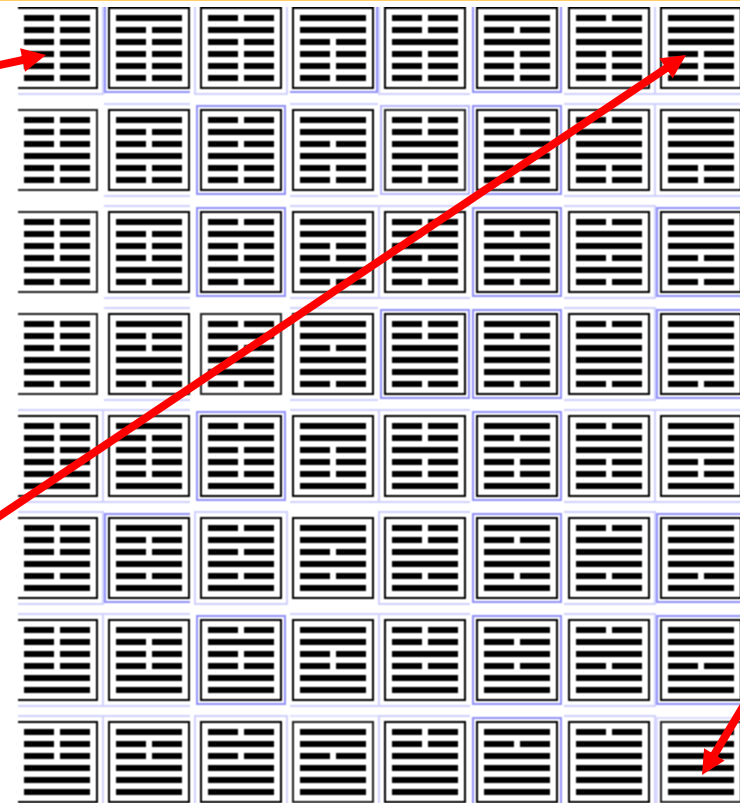


Leibniz - Binary System

Wen Wang (who flourished in about 1150 BC) is traditionally thought to have been author of the present hexagrams



Figures of Fohi			
0	--	0	0
1	—	1	1
00	==	0	0
01	==	1	1
10	==	10	2
11	==	11	3
000	===	0	0
001	===	1	1
010	===	10	2
011	===	11	3
100	===	100	4
101	===	101	5
110	===	110	6
111	===	111	7

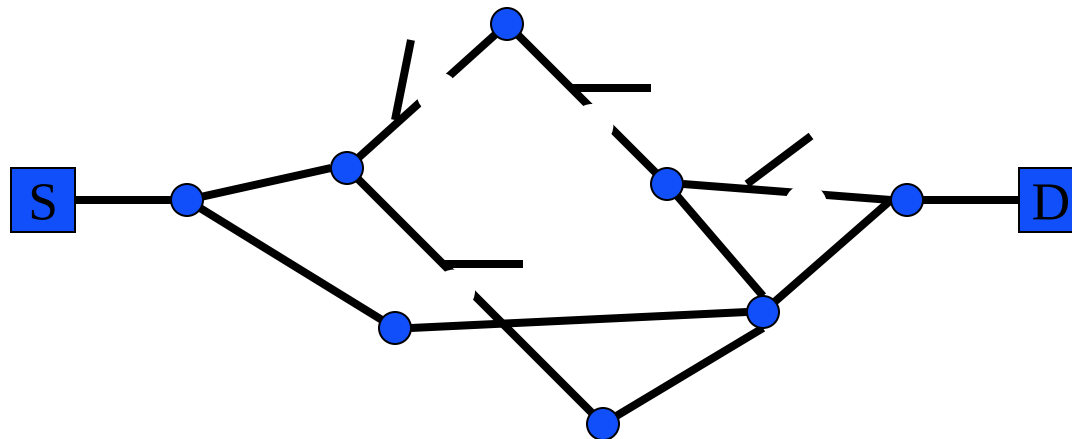


63

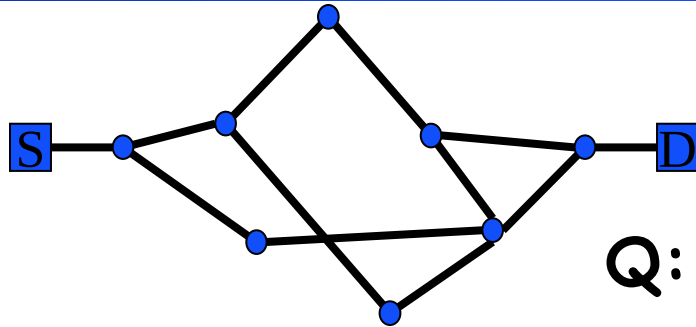
Connection Between Boolean Calculus and Physical Circuits Shannon 1938

The basic framework:

- A graph with edges and vertices
- Source and destination vertices (**S** and **D**)
- Boolean variables control the flow in the edges
- The function is **1** if **S** and **D** are **connected**
- The function is **0** if **S** and **D** are **disconnected**

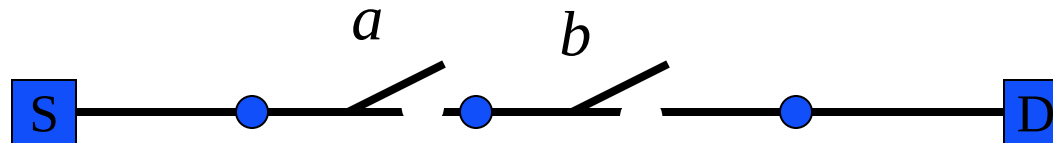


Connection Between Boolean Calculus and Physical Circuits Shannon 1938

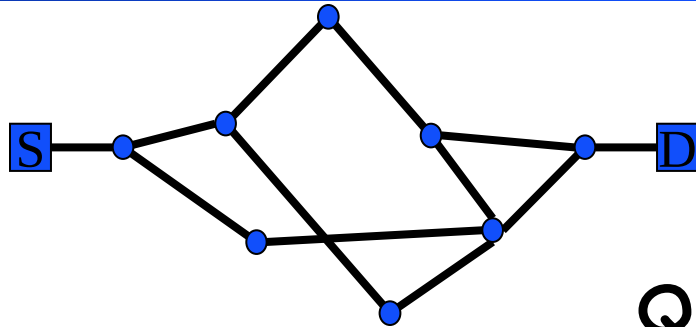


Q: How to design an AND gate?

$$a \cdot b$$

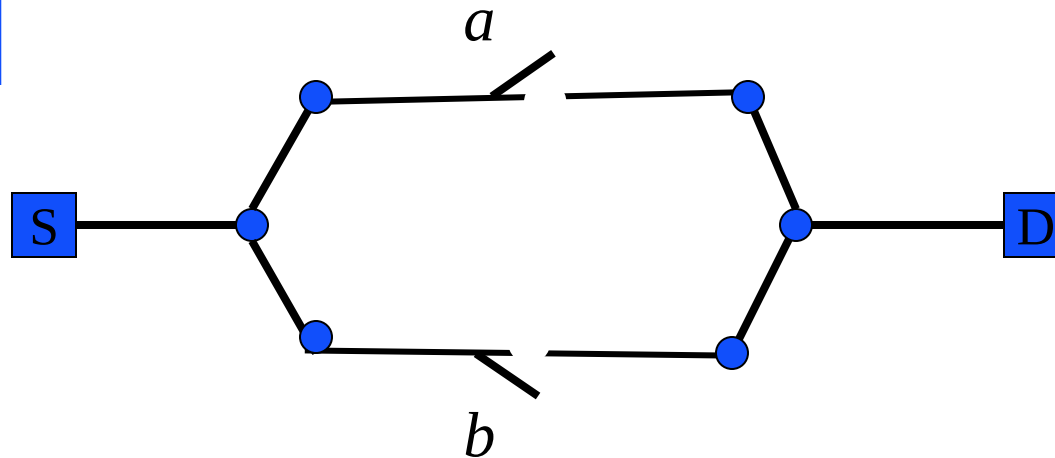


Connection Between Boolean Calculus and Physical Circuits Shannon 1938



Q: How to design an OR gate?

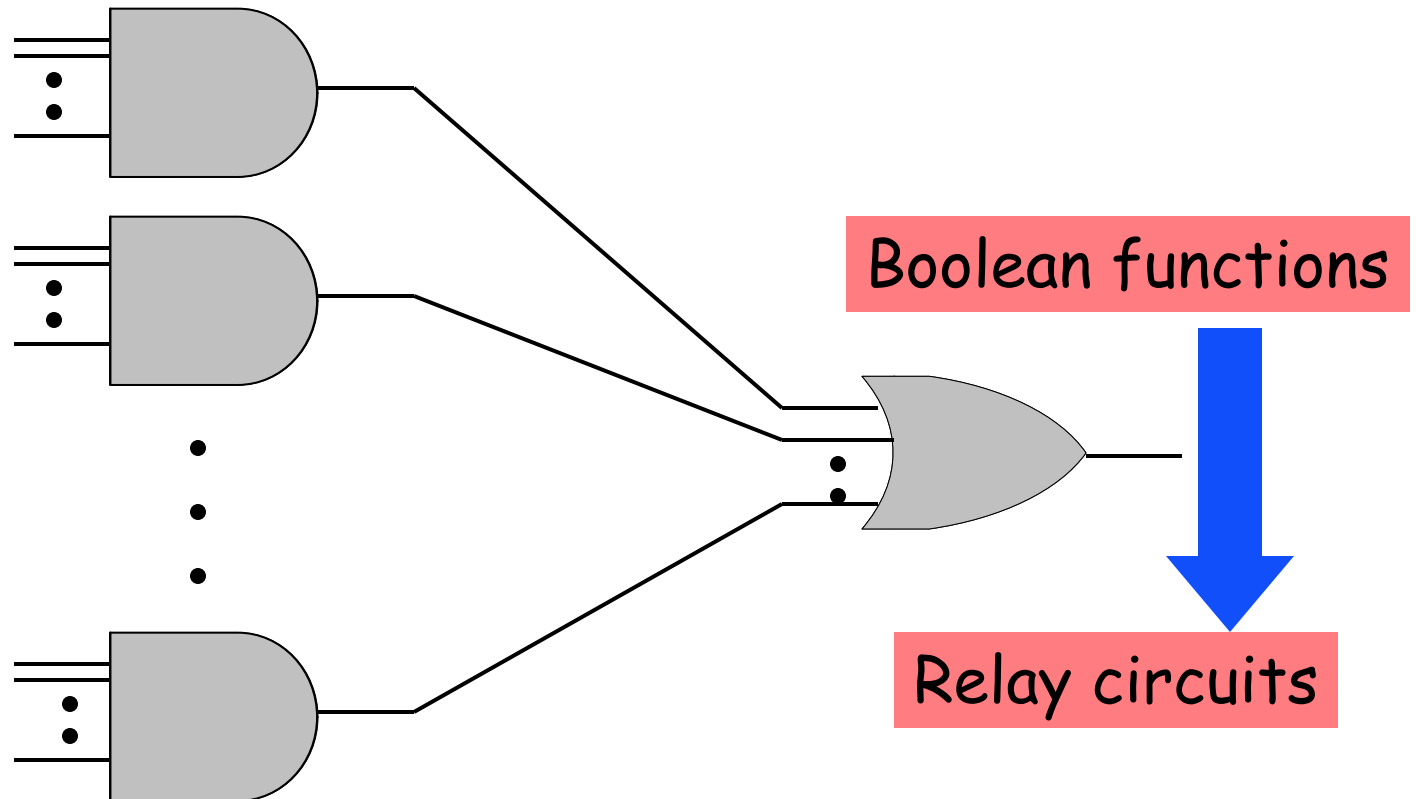
$$a + b$$



Every 0-1 Boolean Function Can be Implemented Using A Depth Two Circuit

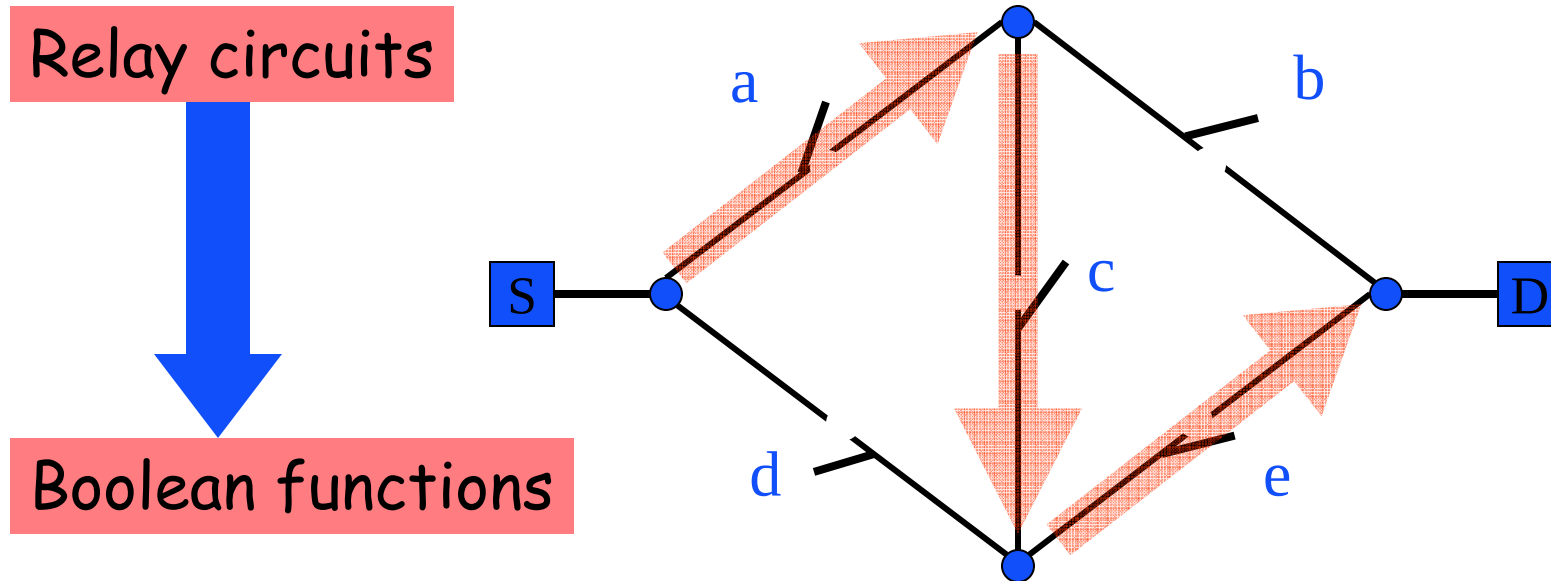
How?

Implement the DNF representation: *OR* of many *AND*s



Connection Between Boolean Calculus and Physical Circuits Shannon 1938

Q: relay circuits to Boolean calculus?



$$f = ab + de + ace + dcb$$

Connection Between Boolean Calculus and Physical Circuits Shannon 1938

Boolean functions

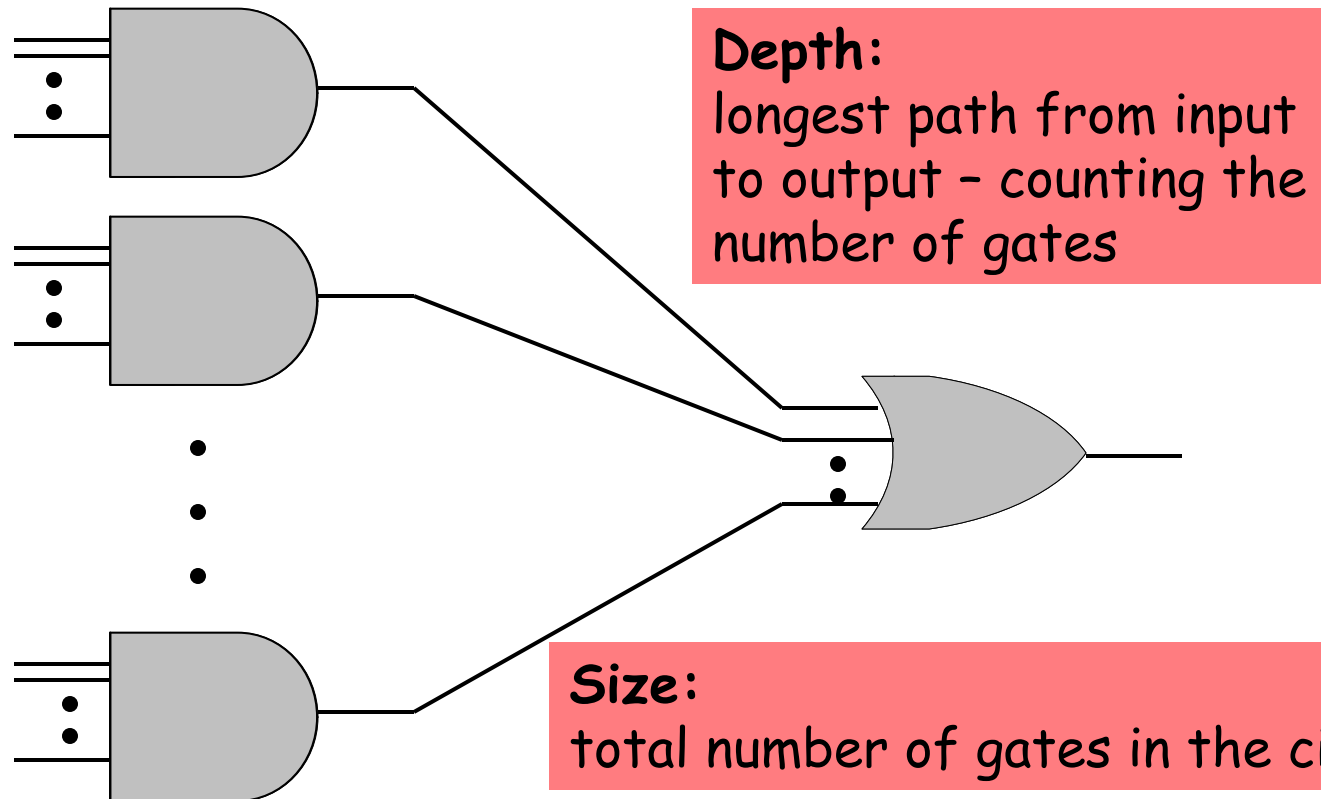


Relay circuits

Every 0-1 Boolean Function Can be Implemented Using A **Depth Two** Circuit

How?

Implement the DNF representation: *OR* of many *AND*s

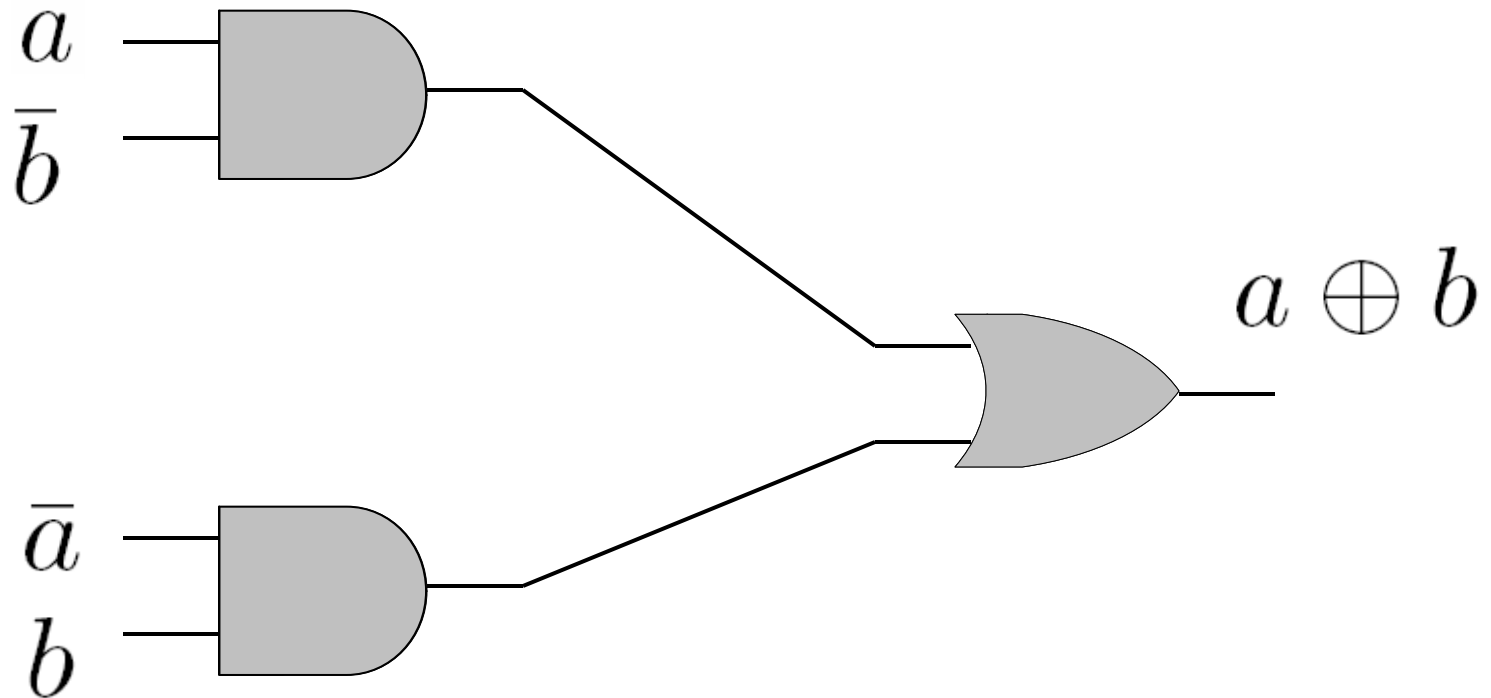


XOR of Two Variables

$$a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$

Depth = 2

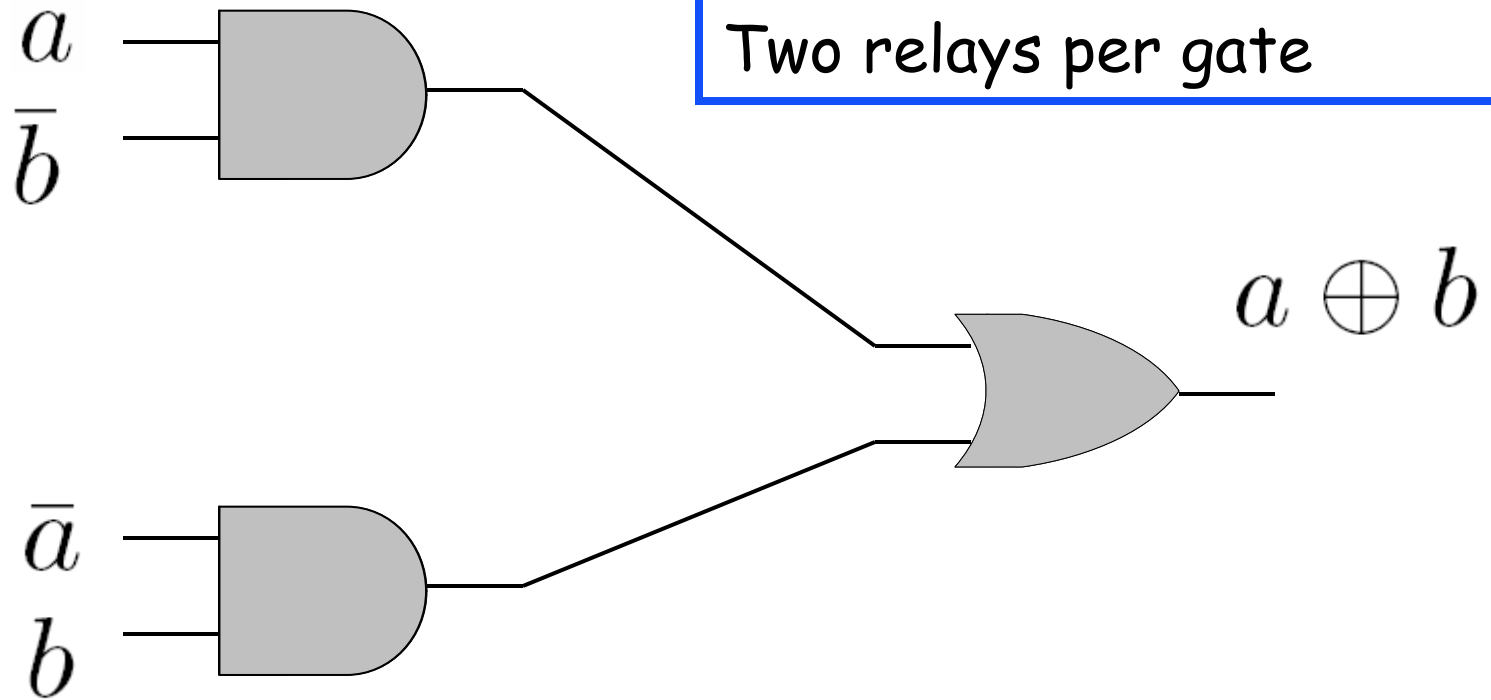
Size = 3



XOR of two Variables with Relays?

$$a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$

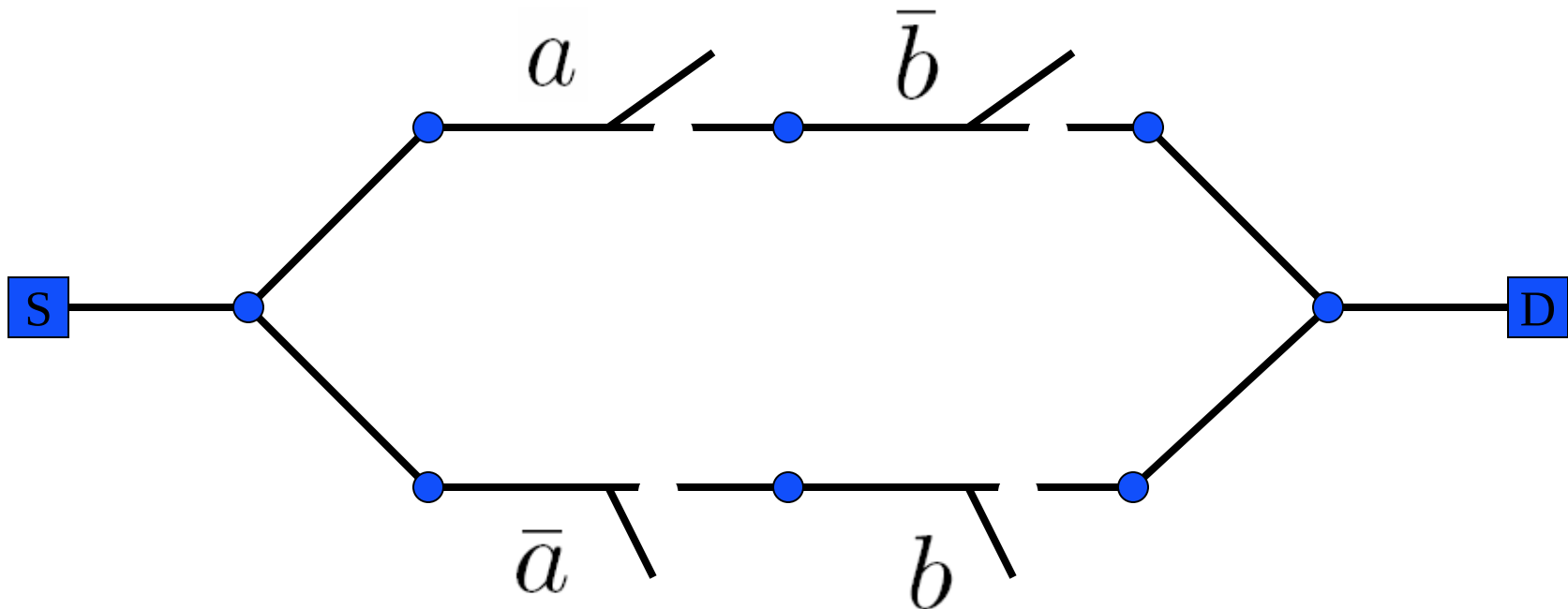
We need to use 6 relays
Two relays per gate



XOR of two Variables with Relays?

$$a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$$

Q: Can we compute XOR with less than 6 relays?



XOR of More Variables

$$XOR(x_1, x_2, \dots, x_n) = \begin{cases} 0 & \text{if } |X| \text{ (number of 1's in } X) \text{ is even} \\ 1 & \text{if } |X| \text{ is odd} \end{cases}$$

$$a \oplus b \oplus c = \bar{a} \cdot \bar{b} \cdot c + \bar{a} \cdot b \cdot \bar{c} + a \cdot \bar{b} \cdot \bar{c} + a \cdot b \cdot c$$

abc	XOR(a,b,c)
000	0
001	1
010	1
011	0
100	1
101	0
110	0
111	1

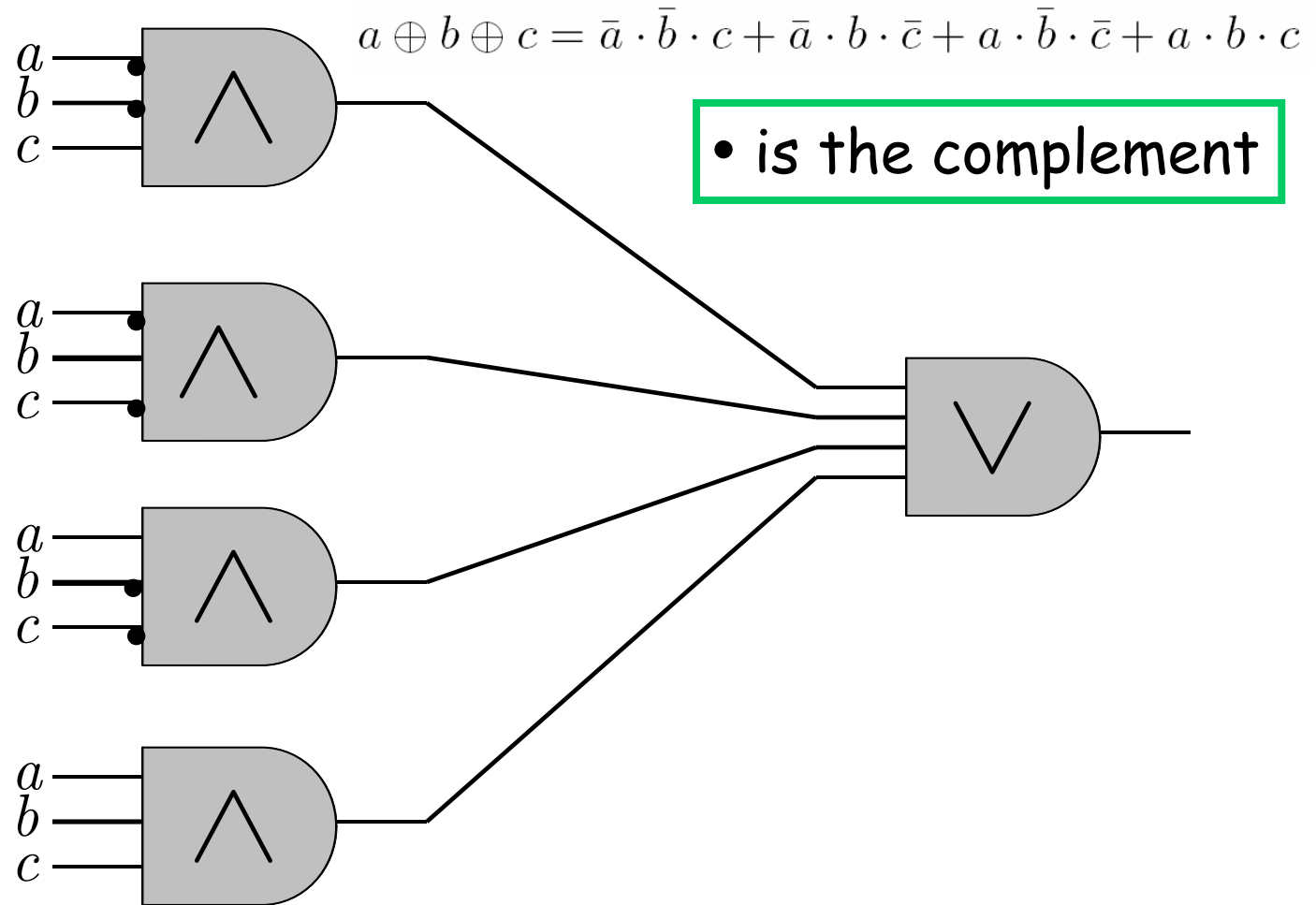
How many gates for XOR with AON?

$$\mathbf{AON} = \mathbf{A}_{\text{ND}} \mathbf{O}_{\text{R}} \mathbf{N}_{\text{OT}}$$

XOR of Three Variables with AON

Depth = 2

Size = 5



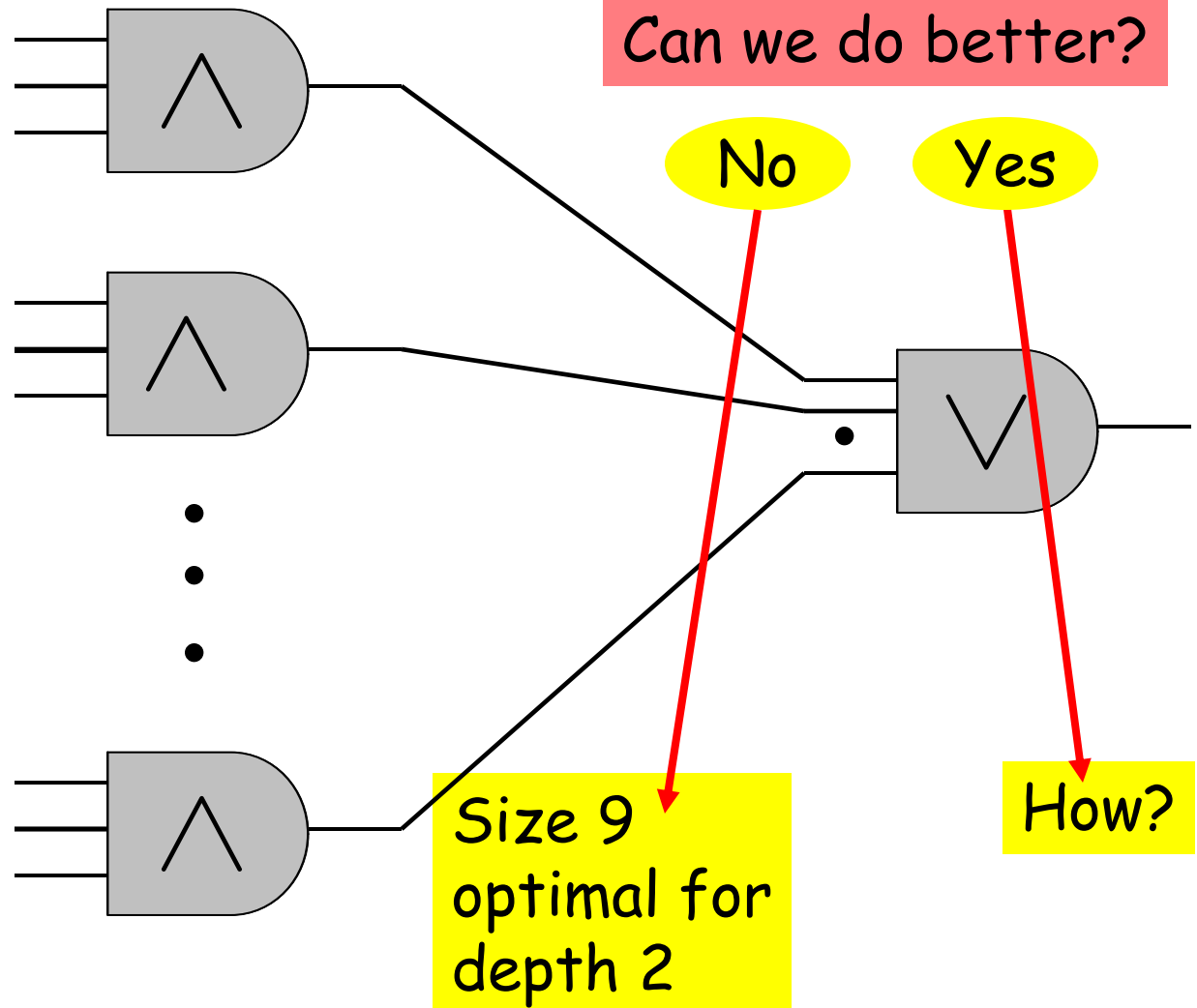
XOR of Four Variables with AON

Depth = 2

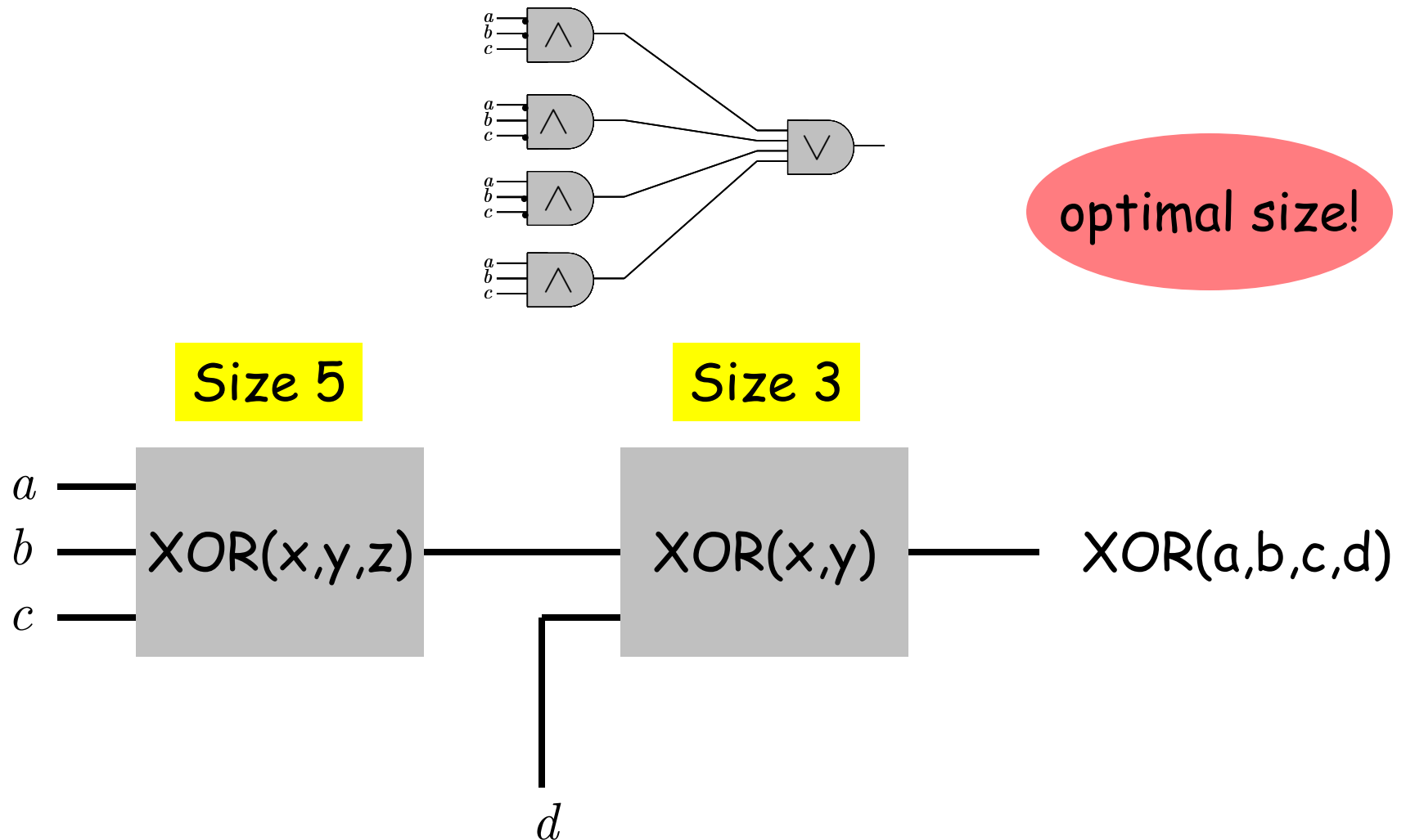
Size = 9

4 *AND*s for $|X| = 1$

4 *AND*s for $|X| = 3$

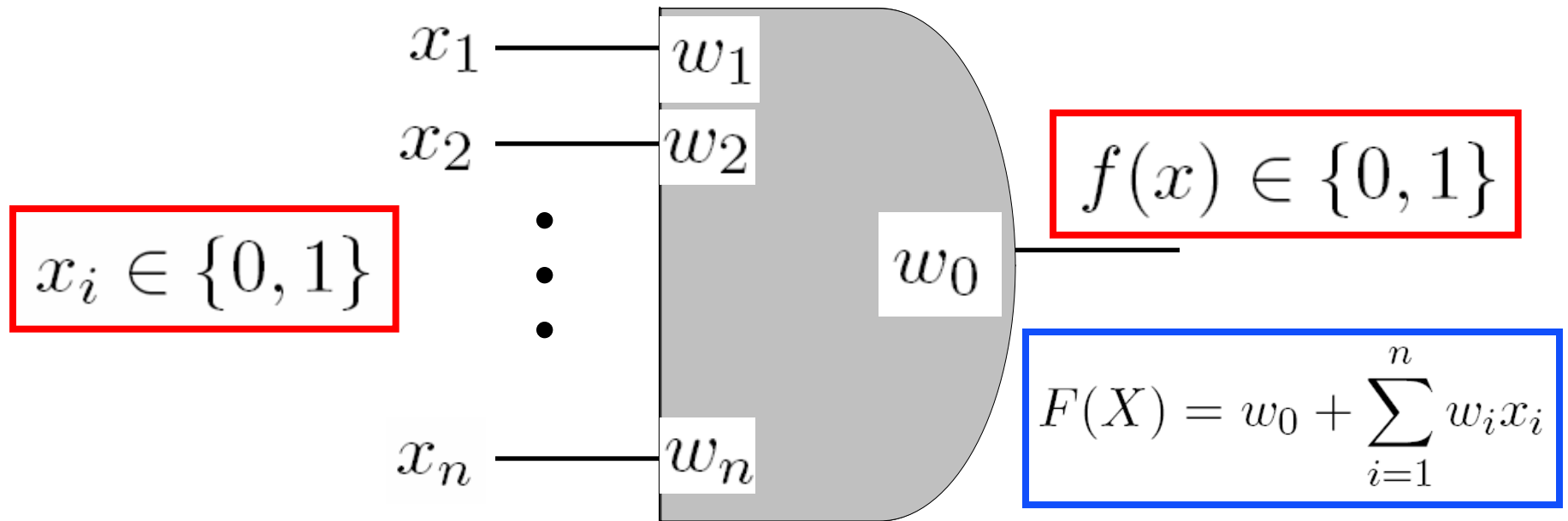


Size 8 XOR AON Circuit of Four Variables



Neuron - Neural Gate

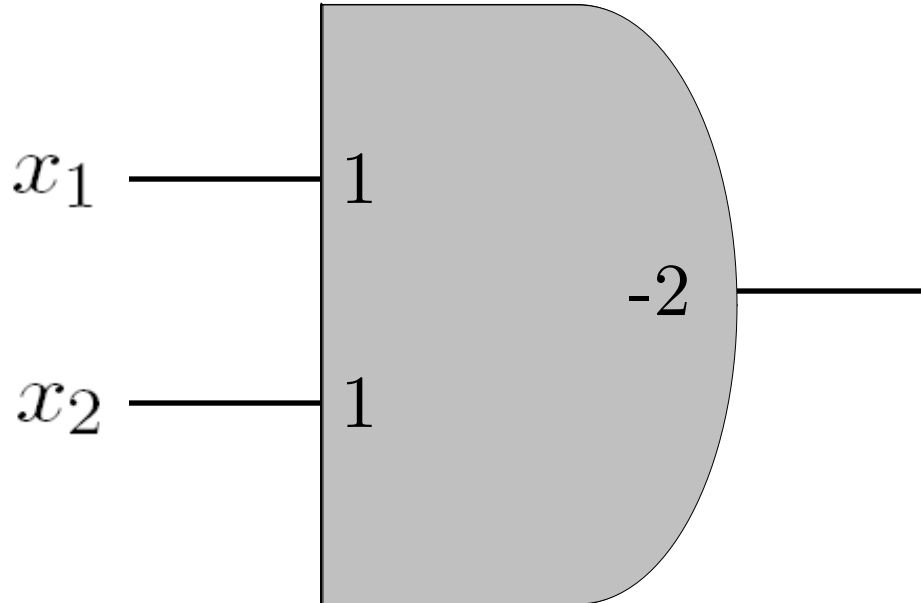
Linear Threshold (LT) gate



$$f(X) = \text{sgn}(F(X)) = \begin{cases} 0 & \text{if } F(X) < 0 \\ 1 & \text{if } F(X) \geq 0 \end{cases}$$

Can We Compute an AND Function with an LT Gate?

$$AND(x_1, x_2)$$



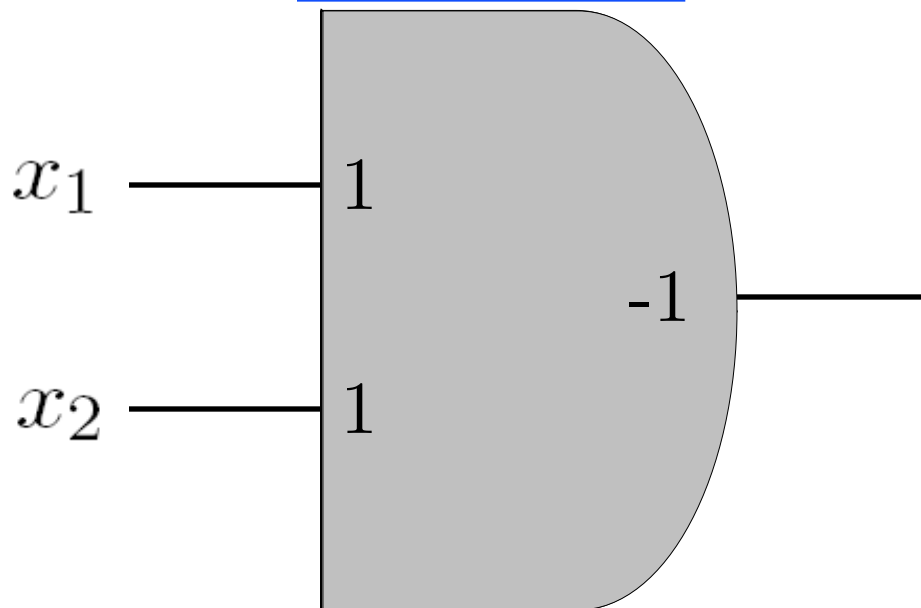
$$F(x_1, x_2) = -2 + x_1 + x_2$$

$$F(X) = w_0 + \sum_{i=1}^n w_i x_i$$

x_1	x_2	$F(X)$	$f(X)$
0	0	-2	0
0	1	-1	0
1	0	-1	0
1	1	0	1

Can We Compute an OR Function with an LT Gate?

$$OR(x_1, x_2)$$

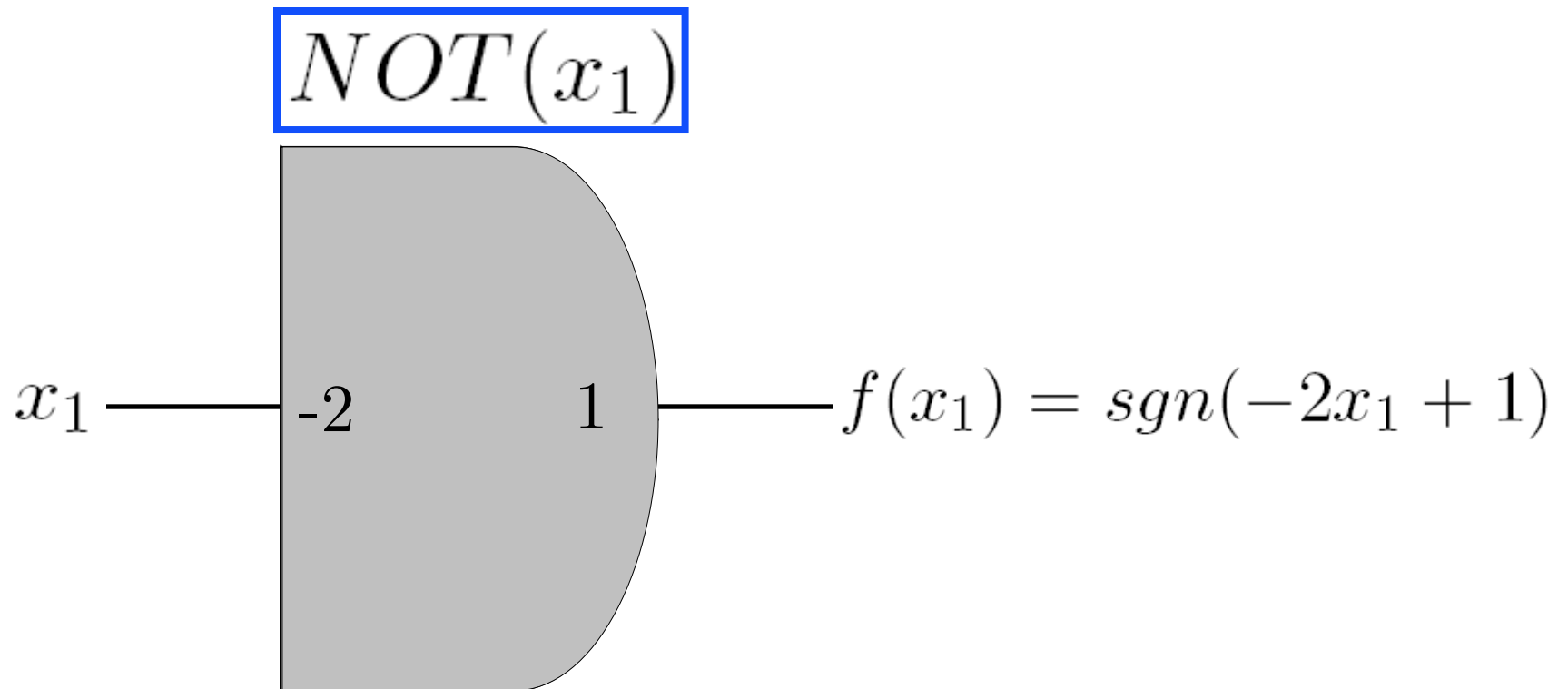


$$F(x_1, x_2) = -1 + x_1 + x_2$$

$$F(X) = w_0 + \sum_{i=1}^n w_i x_i$$

x_1	x_2	$F(X)$	$f(X)$
0	0	-1	0
0	1	0	1
1	0	0	1
1	1	1	1

Can We Compute a NOT with an LT Gate?



Can we compute **NOT**
without sgn ?

$$\bar{x}_1 = NOT(x_1) = 1 - x_1$$

More Variables for AND?

$$F(x_1, x_2) = -2 + x_1 + x_2$$

$$F(x_1, x_2, \dots, x_n) = -n + x_1 + x_2 + \dots + x_n$$

$$|X| = \sum_{i=1}^n x_i$$

$$-n + |X|$$

$$F(X) = \begin{cases} 0 & \text{if } |X| = n \\ < 0 & \text{if } |X| < n \end{cases}$$

Hence $\text{sgn}(F(X))$ is an AND

More Variables for OR?

$$F(x_1, x_2) = -1 + x_1 + x_2$$

$$F(x_1, x_2, \dots, x_n) = -1 + x_1 + x_2 + \dots + x_n$$

$$|X| = \sum_{i=1}^n x_i$$

$$-1 + |X|$$

$$F(X) = \begin{cases} -1 & \text{if } |X| = 0 \\ \geq 0 & \text{if } |X| \geq 1 \end{cases}$$

Hence $\text{sgn}(F(X))$ is an OR

AON vs LT Circuits

Theorem (LT/AON):

LT circuits are at least as powerful as AON circuits:

(i) **Size** of an **AON** circuit for a Boolean function f is greater/equal **size** of an optimal **LT** circuit for the function f

(ii) **Depth** of an **AON** circuit for a Boolean function f is greater/equal **depth** of an optimal **LT** circuit for the function f

AON vs LT Circuits

Theorem (LT/AON):

LT circuits are at least as powerful as AON circuits:

(i) **Size** of an **AON** circuit for a Boolean function f is greater/equal
size of an optimal **LT** circuit for the function f

(ii) **Depth** of an **AON** circuit for a Boolean function f is greater/equal
depth of an optimal **LT** circuit for the function f

Proof:

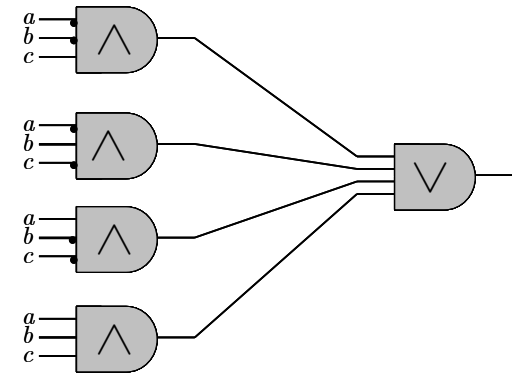
By the previous reductions:
AND, *OR* and *NOT* can be implemented by a single LT gate

LT gates are MORE Powerful

XOR of three variables with AON

Depth = 2

Size = 5



XOR of four variables with AON

Depth = 2

Size = 9

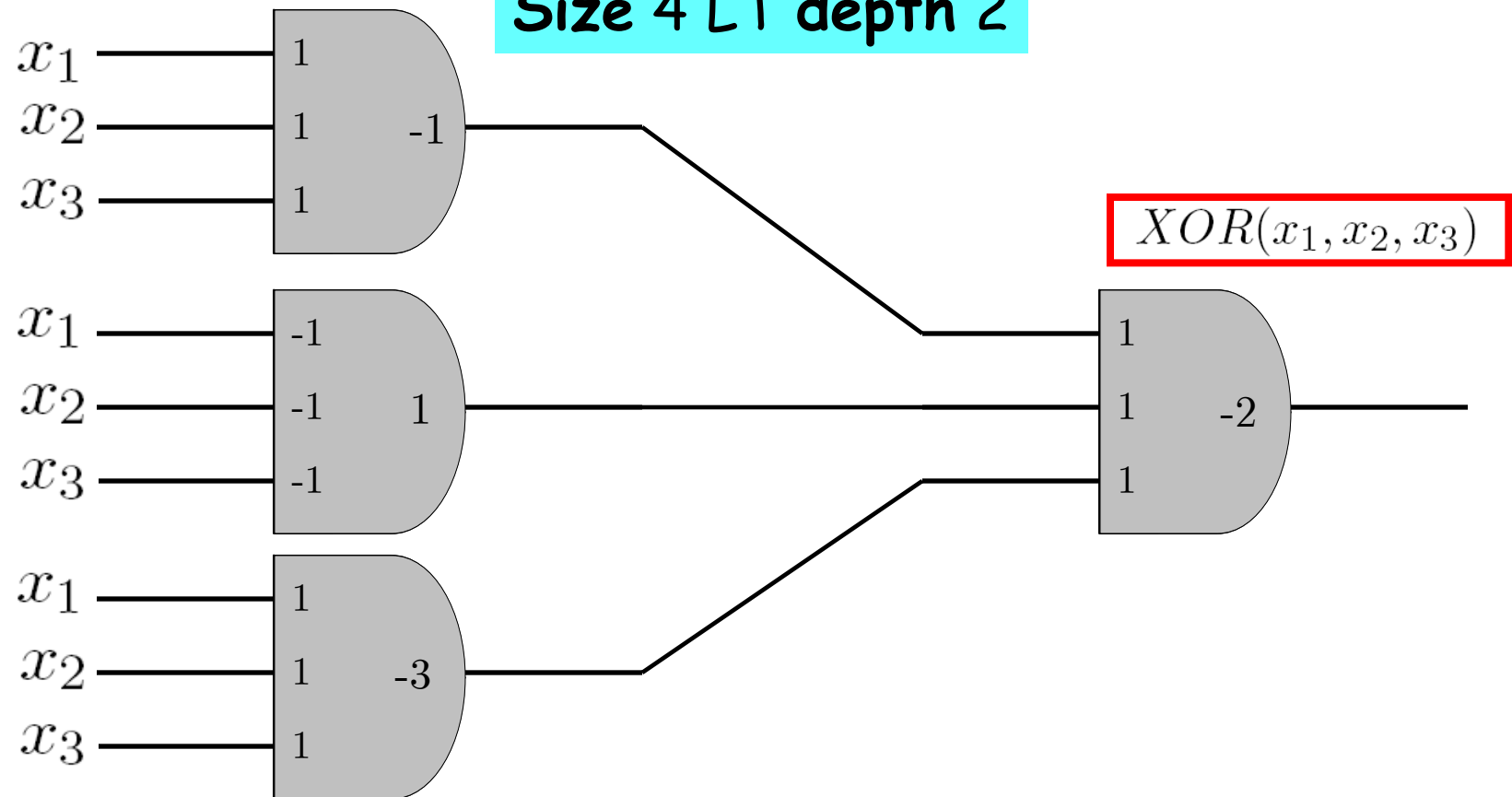
Size is optimal for depth 2

XOR of four variables with AON,
size 8 is optimal for any depth

LT gates are MORE Powerful

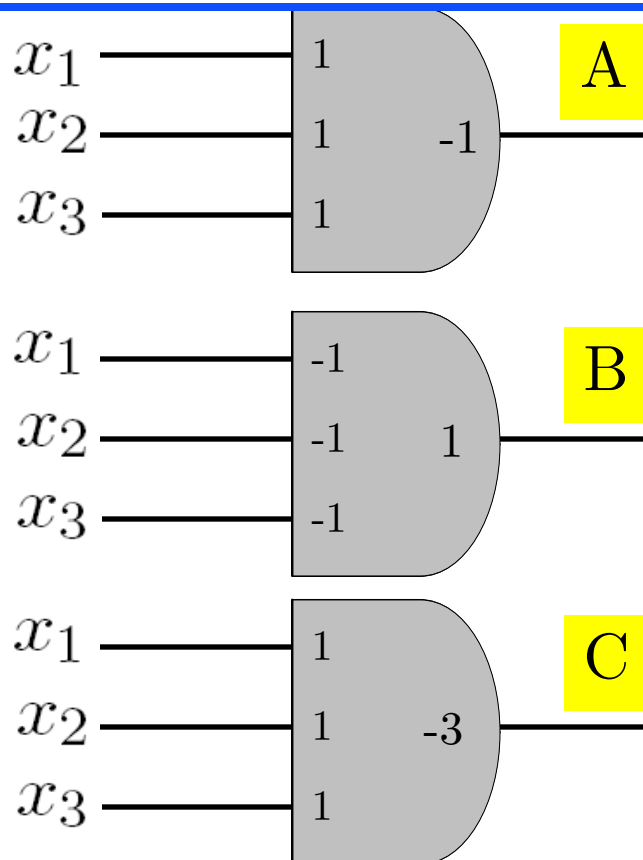
Size 5 is optimal for AON depth 2

Size 4 LT depth 2



LT gates are MORE Powerful

LT-1 = LT layered, inputs go to first layer only

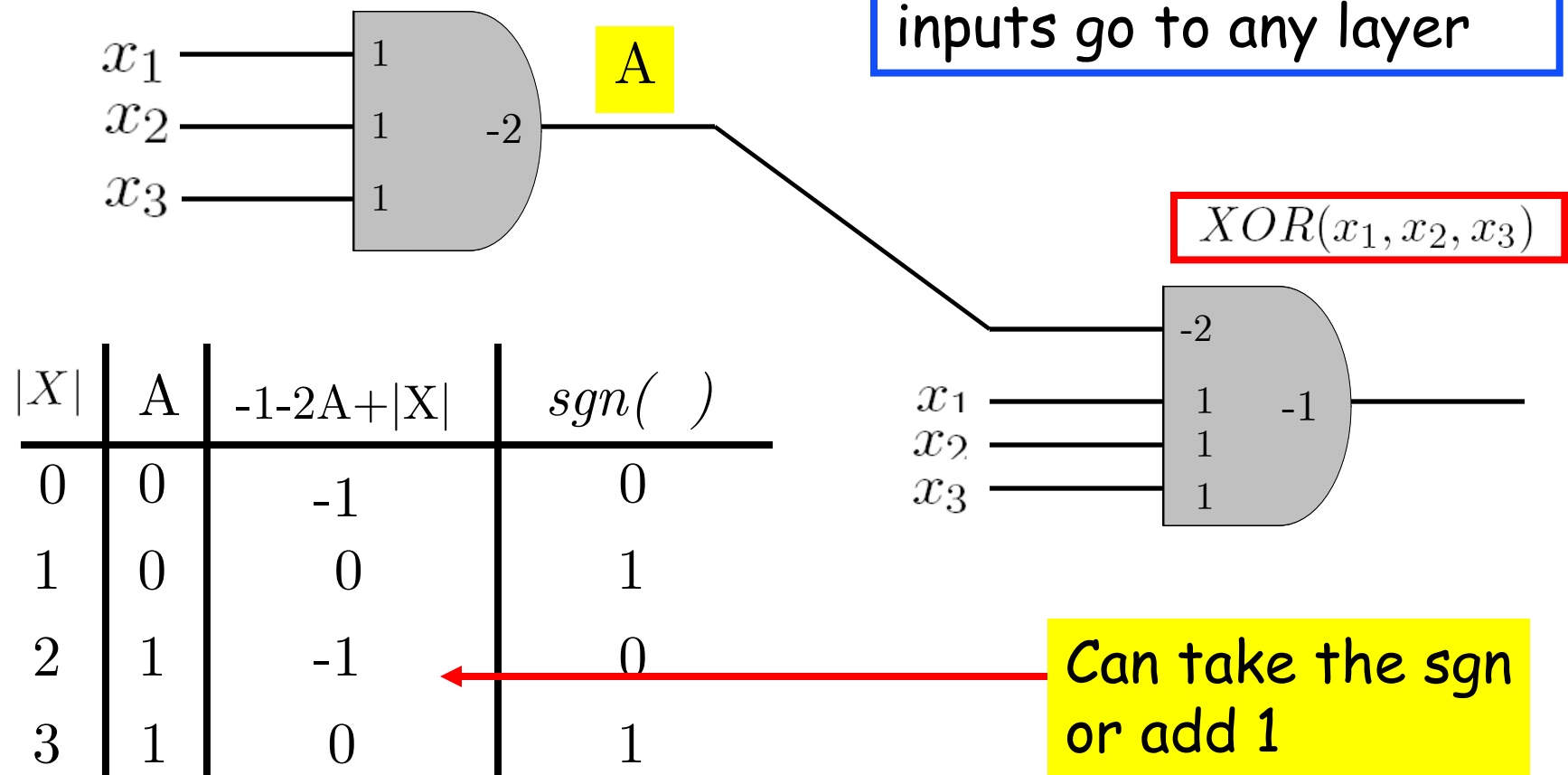


$ X $	A	B	C	$A+B+C$	$-2+A+B+C$
0	0	1	0	1	-1
1	1	1	0	2	0
2	1	0	0	1	-1
3	1	0	1	2	0

$XOR(x_1, x_2, x_3)$

Can take the sgn or add 1

LT gates are EVEN MORE Powerful



XOR Function: Size of LT vs AON in Depth 2

	$XOR(x_1, x_2, x_3)$	$XOR(x_1, x_2, \dots, x_n)$
AON	5	$2^{n-1} + 1^*$
LT-I	4	$n + 1^*$
LT-nl	2	$\left\lfloor \frac{n}{2} \right\rfloor + 1^*$

* = it is optimal

Exponential gap in size

XOR with a Single LT Gate

	$XOR(x_1, x_2, x_3)$	$XOR(x_1, x_2, \dots, x_n)$
AON	5	$2^{n-1} + 1$
LT-I	4	$n + 1$
LT-nl	2	$\left\lfloor \frac{n}{2} \right\rfloor + 1$

Is it possible to compute $XOR(x_1, x_2)$ with a single LT element?

Is this result general? more variables?

XOR with a Single LT Gate

Is it possible to compute $XOR(x_1, x_2)$ with a single LT element?

Answer : NO

Proof: By contradiction
assume it is possible and reach a contradiction

$$XOR(x_1, x_2) = \text{sgn}(w_0 + w_1x_1 + w_2x_2)$$

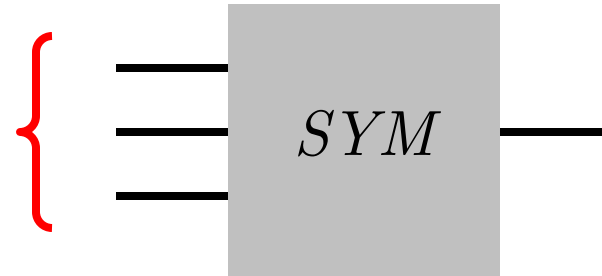
$x_1 = 0 \quad x_2 = 0$	$\longrightarrow$	$w_0 < 0$	$\longrightarrow$	$2w_0 + w_1 + w_2 < 0$
$x_1 = 1 \quad x_2 = 0$	$\longrightarrow$	$w_0 + w_1 \geq 0$	$\longrightarrow$	$2w_0 + w_1 + w_2 \geq 0$
$x_1 = 0 \quad x_2 = 1$	$\longrightarrow$	$w_0 + w_2 \geq 0$	$\longrightarrow$	$2w_0 + w_1 + w_2 \geq 0$
$x_1 = 1 \quad x_2 = 1$	$\longrightarrow$	$w_0 + w_1 + w_2 < 0$	$\longrightarrow$	$2w_0 + w_1 + w_2 < 0$

Contradiction

Symmetric Functions

AND, *OR* and *XOR* are symmetric Boolean functions

Permuting the inputs does
not change the output



$$XOR(x_1, x_2, x_3) = XOR(x_2, x_1, x_3) = XOR(x_2, x_3, x_1)$$

Definition: A Boolean function f is symmetric if

$$f(X) = f(\pi(X))$$

for an arbitrary permutation π

Questions on Symmetric Functions

AND, OR and XOR are symmetric Boolean functions

Q1: How many symmetric Boolean functions of n variables?

Q2: What are the symmetric functions that can be realized by a single LT gate?

Marshall Stone
1903-1989



Boolean representation
theorem, **1936**

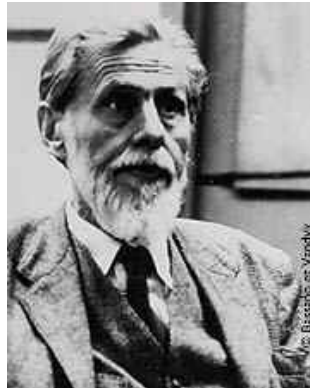
History on LT

Claude Shannon
1916-2001



Boolean algebra to
electrical circuits
1938

Warren McCulloch
1899 - 1969



Walter Pitts
1923 - 1969

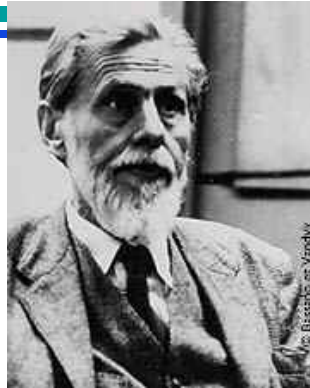


Computing with LT (neural) gates: connection
between circuits and neural networks, **1943**

History on LT

Warren McCulloch
1899 - 1969

Neurophysiologist, MD



Walter Pitts
1923 - 1969

Logician, Autodidact



Computing with LT (neural) gates: connection between circuits and neural networks, **1943**

Warren McCulloch arrived in early 1942 to the University of Chicago, invited Pitts, who was still homeless, to live with his family.

In the evenings McCulloch and Pitts collaborated. Pitts was familiar with the work of Gottfried Leibniz on computing and they considered the question of whether the nervous system could be considered a kind of universal computing device as described by Leibniz.

This led to their 1943 seminal neural networks paper:
A Logical Calculus of Ideas Immanent in Nervous Activity.