An abstract graphic on the left side of the slide, featuring a complex network of yellow lines resembling a circuit board. These lines are interspersed with small black and white dots, creating a dense, branching pattern that extends from the bottom left towards the top left.

# **Tecnologia da informação e comunicação**

## **ÁLGEBRA BOOLEANA**

**FELIPE G. TORRES**

# REVISÃO – ÁLGEBRA DE BOOLE



## GEORGE BOOLE

(2 de novembro de 1815 — 8 de dezembro de 1864)

- Foi um dos matemáticos fundamentais para a criação da computação moderna.
- Desenvolveu o modelo matemático lógico, conhecido como Álgebra Booleana.
- Quase todas as linguagens de programação possuem variáveis booleanas em sua homenagem.

### Álgebra “Tradicional”:

- Variáveis representam números reais;
- Operadores são aplicados as variáveis e o resultado é um número real.

### Álgebra Booleana:

- Variáveis representam apenas 0 ou 1;
- Operadores retornam apenas 0 ou 1.

# FUNÇÕES E VARIÁVEIS LÓGICAS

Seguem algumas definições importantes :

- **Variável booleana:** é uma quantidade que pode ser, em diferentes momentos, igual a 0 ou 1.
- **Função booleana:** associa a cada  $n$  variáveis de entrada uma única saída.

Podemos descrever uma função booleana utilizando:

- tabela verdade
- portas lógicas
- equações
- formas de onda

Diferente da álgebra comum, a álgebra booleana possui somente três operações básicas : OR, AND e NOT, conhecidas como operações lógicas.

Por exemplo, supondo:

$x = \text{jovem};$

$y = \text{faz Sistemas de Informação};$

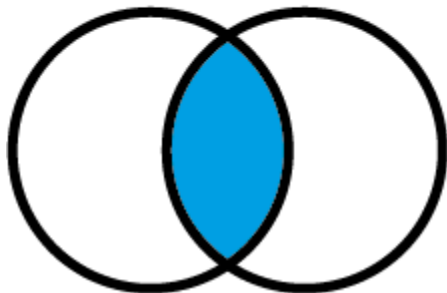
- $(1-x)$  iria então representar a operação de selecionar todas as coisas no mundo exceto jovens, isto é, todas as coisas que não são jovens;
- $(xy)$  representaria o conjunto dos jovens que fazem Sistemas de informação;
- $(1 - x) (1 - y)$  seriam todas as coisas que não são jovens nem fazem Sistemas de informação;
- $(x + y)$  seria o conjunto das coisas que são jovens ou que fazem Sistemas de informação.

# ÁLGEBRA DE BOOLE

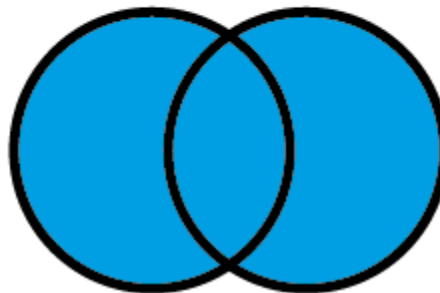
- A álgebra de boole possui diversos componentes que possibilitam a aplicação de lógica na matemática.

Alguns dos principais componentes desse modelo são:

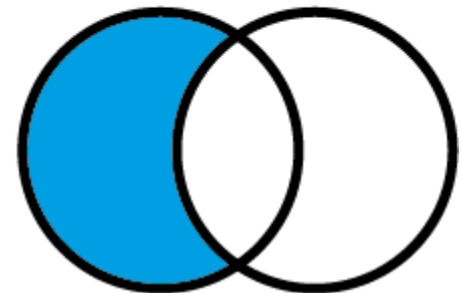
**AND**



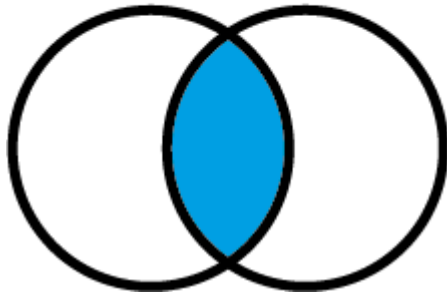
**OR**



**NOT**



# AND



Este operador entre duas ou mais variáveis somente apresenta o resultado 1 se todas as variáveis estiverem no estado lógico 1.

Também chamado de conjunção, esse operador pode ser representado por:

**AND**

**&&**

**\***

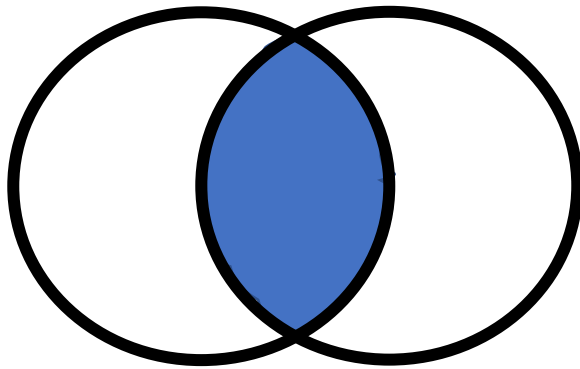
**.**

**^**

# ÁLGEBRA DE BOOLE

Ex:.

Chocólatras      Diabéticos



Uma parcela da população de chocólatras são diabéticos.

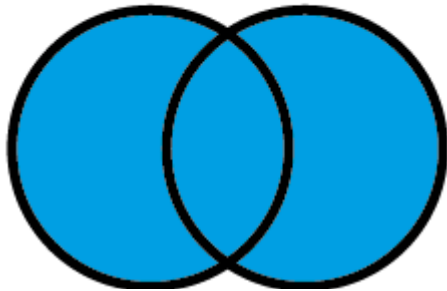
*Chocólatra AND Diabético*

**Tabela verdade:**

Chocólatra	Diabético	Chocólatra AND Diabético
S / 1	S / 1	S / 1
S / 1	N / 0	N / 0
N / 0	N / 0	N / 0
N / 0	S / 1	N / 0



# OR



Este operador entre duas ou mais variáveis somente apresenta o resultado 1 se pelo menos uma das variáveis estiverem no estado lógico 1.

Também chamado de disjunção, esse operador pode ser representado por:

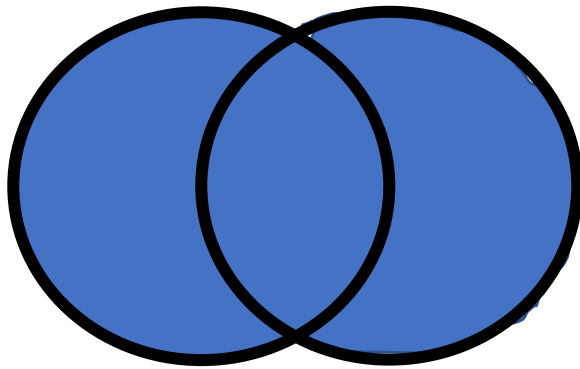
OR    ||    +    v

# ÁLGEBRA DE BOOLE

Ex:.

Média < 7

Faltas > 30%



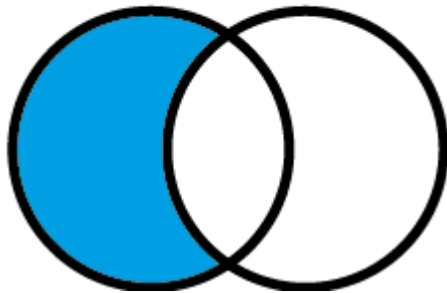
Se o aluno tem a média menor que 7 ou possui mais de 30% de faltas, ele está reprovado.

*Média > 7 AND Faltas > 30%*

**Tabela verdade:**

Média < 7	Faltas > 30%	Média < 7 OR Faltas > 30%
S / 1	S / 1	S / 1
S / 1	N / 0	S / 1
N / 0	N / 0	N / 0
N / 0	S / 1	S / 1

# NOT



Este operador de complementação ou inversão de uma variável é implementada através da inversão do valor lógico da variável referida.

Também chamado de negação, esse operador pode ser representado por:

**NOT**

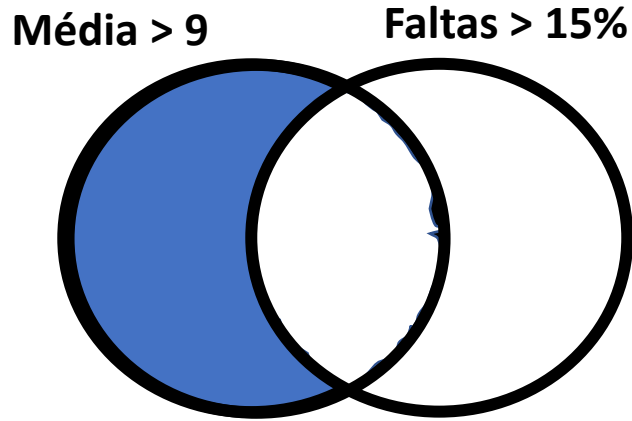
**!**

**¬**

**~**

# ÁLGEBRA DE BOOLE

Ex:.



Somente os alunos com faltas  $\leq 15\%$  serão selecionados para o concurso de programação.

**NOT** *Faltas > 15%*

**Tabela verdade:**

Faltas > 15%	NOT Faltas > 15%
S / 1	N / 0
N / 0	S / 1

Ex:.

“Eu irei almoçar se Maria ou João forem e se Célia não for.”

- Avaliação formal (para  $m = 1$ ,  $j = 0$ ,  $c = 1$ ):

$$F = (m \text{ OR } j) \text{ AND NOT}(c)$$

$$F = (m + j) \neg c$$

$$F = (1 + 0) \cdot \neg 1 = 1 \cdot 0 = 0$$

# ÁLGEBRA DE BOOLE

Ex.: Simplificar  $(a + b)(a + \neg b + \neg c)$ :

$$\begin{aligned}
 & (a+b)(a + \neg b + \neg c)= \\
 & = a(a + b) + \neg b(a + b) + \neg c(a + b)= \\
 & = aa + ab + \neg ba + \neg bb + \neg ca + \neg cb= \\
 & = a + ab + \neg ba + 0 + \neg ca + \neg cb= \\
 & = a + ab + \neg ba + \neg ca + \neg cb= \\
 & = a + \neg ba + \neg ca + \neg cb= \\
 & = a + a(\neg b + \neg c) + \neg cb= \\
 & = a + a \neg(bc) + \neg cb= \\
 & = a + a \neg(bc) + \neg cb= \\
 & = a + \neg cb= \\
 & = a + \neg cb
 \end{aligned}$$

Seja uma função  $f(A_1, \dots, A_n)$  com  $n$  entradas. A tabela verdade expressa o estado da saída para todas as combinações possíveis dos estados de entrada  $\{A_1, \dots, A_n\}$ . Segue um exemplo para duas entradas.

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	1
0	1	1
1	0	1
1	1	0

Além de 0s e 1s a função  $f(\cdot)$  pode ser igual ao caracter  $x$ , chamado de *don't care*. Este caracter serve para indicar que para uma dada combinação de entradas,  $x$  pode ser tanto 0 como 1.

# OPERAÇÕES E PORTAS LÓGICAS

- **Operação NOT:** Para qualquer entrada  $A$ , ela é definida como:

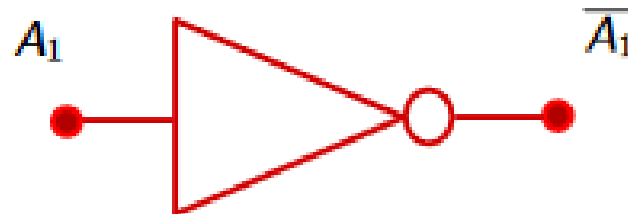
$$f(A) = \bar{A}$$

ou seja, é a entrada negada (barrada). Para uma entrada  $A_1$ , por exemplo temos:

Tabela verdade

$A_1$	$f(A_1)$
0	1
1	0

Porta lógica





# OPERAÇÕES E PORTAS LÓGICAS

- Operação OR:** Para qualquer entrada  $\{A_1, \dots, A_n\}$ , ela é definida como:

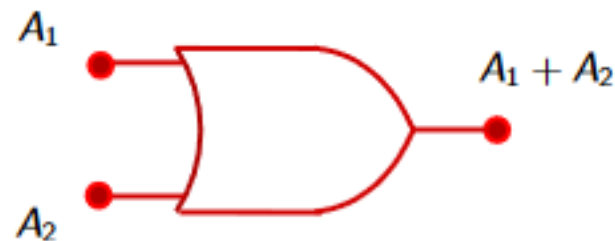
$$f(A_1, \dots, A_n) = \sum_{i=1}^n A_i$$

E vale 1 se qualquer uma das entradas for igual a 1. Para duas entradas temos:

Tabela verdade

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	0
0	1	1
1	0	1
1	1	1

Porta lógica



# OPERAÇÕES E PORTAS LÓGICAS

- **Operação AND:** Para qualquer entrada  $\{A_1, \dots, A_n\}$ , ela é definida como:

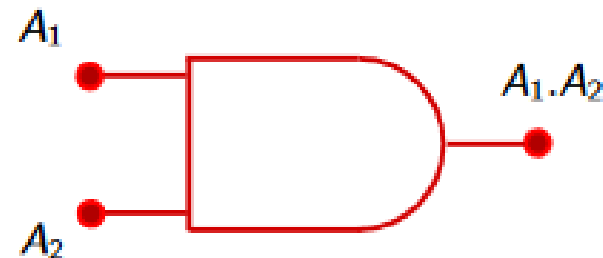
$$f(A_1, \dots, A_n) = \prod_{i=1}^n A_i$$

E vale 1 apenas se todas as entradas forem iguais a 1. Para duas entradas temos:

Tabela verdade

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	0
0	1	0
1	0	0
1	1	1

Porta lógica



# OPERAÇÕES E PORTAS LÓGICAS

- **Operação NOR:** É a operação OR negada. Para duas entradas  $\{A_1, A_2\}$ , ela é definida como:

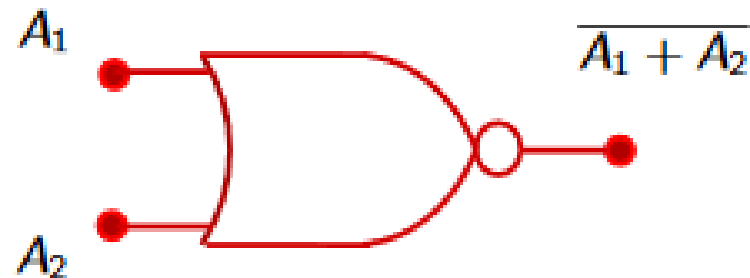
$$f(A_1, A_2) = \overline{A_1 + A_2}$$

E vale 1 apenas se todas as entradas forem iguais a 0. Para duas entradas temos:

Tabela verdade

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	1
0	1	0
1	0	0
1	1	0

Porta lógica



# OPERAÇÕES E PORTAS LÓGICAS

- Sobre a **porta NOR**, podemos realizar os seguintes comentários:
  - Utilizando a tabela verdade podemos verificar que:

$$\overline{A_1 + A_2} = \overline{A_1} \cdot \overline{A_2}$$

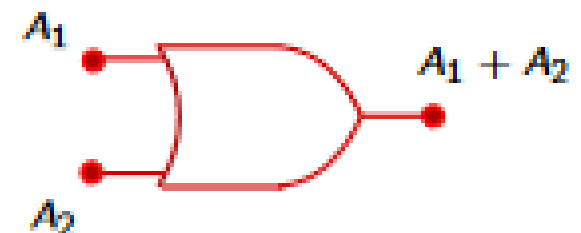
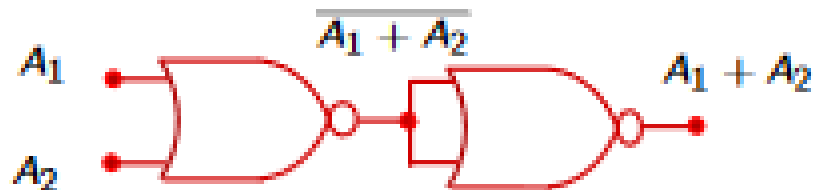
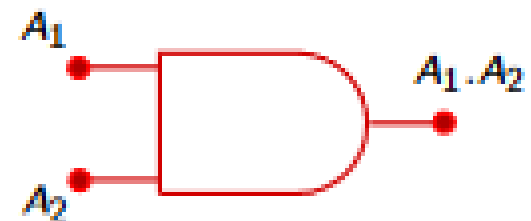
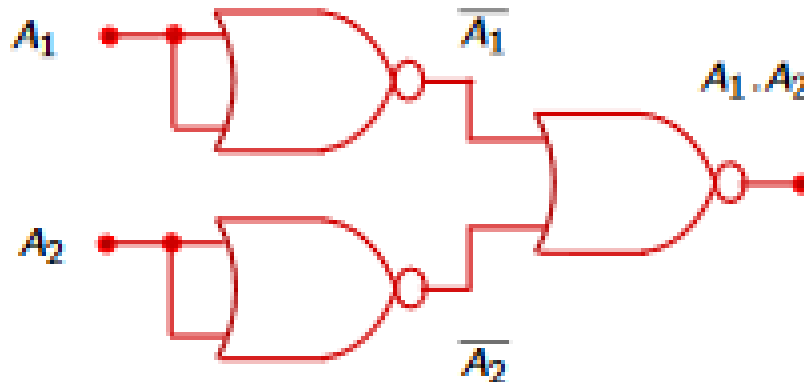
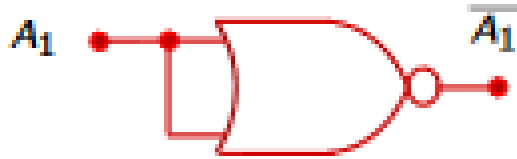
Que é um dos resultados do teorema de Morgan que veremos a seguir...

Tabela verdade

$A_1$	$A_2$	$\overline{A_1 + A_2}$	$\overline{A_1} \cdot \overline{A_2}$
0	0	1	1
0	1	0	0
1	0	0	0
1	1	0	0

# OPERAÇÕES E PORTAS LÓGICAS

- Utilizando apenas a **porta NOR**, podemos obter as outras três portas básicas:



# OPERAÇÕES E PORTAS LÓGICAS

- Operação NAND:** É a operação AND negada. Para duas entradas  $\{A_1, A_2\}$ , ela é definida como:

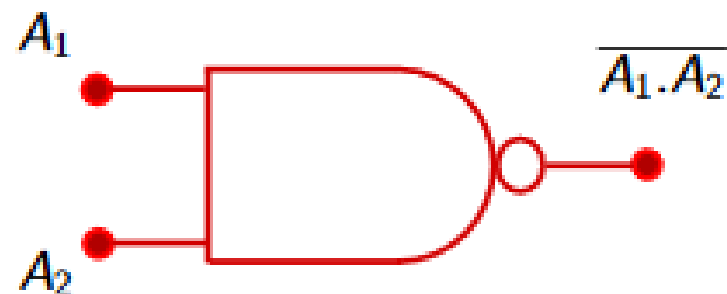
$$f(A_1, A_2) = \overline{A_1 \cdot A_2}$$

E vale 0 apenas se todas as entradas forem iguais a 1. Para duas entradas temos:

Tabela verdade

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	1
0	1	1
1	0	1
1	1	0

Porta lógica



# OPERAÇÕES E PORTAS LÓGICAS

- Sobre a **porta NAND**, podemos realizar os seguintes comentários:
  - Utilizando a tabela verdade podemos verificar que:

$$\overline{A_1} \cdot \overline{A_2} = \overline{A_1 + A_2}$$

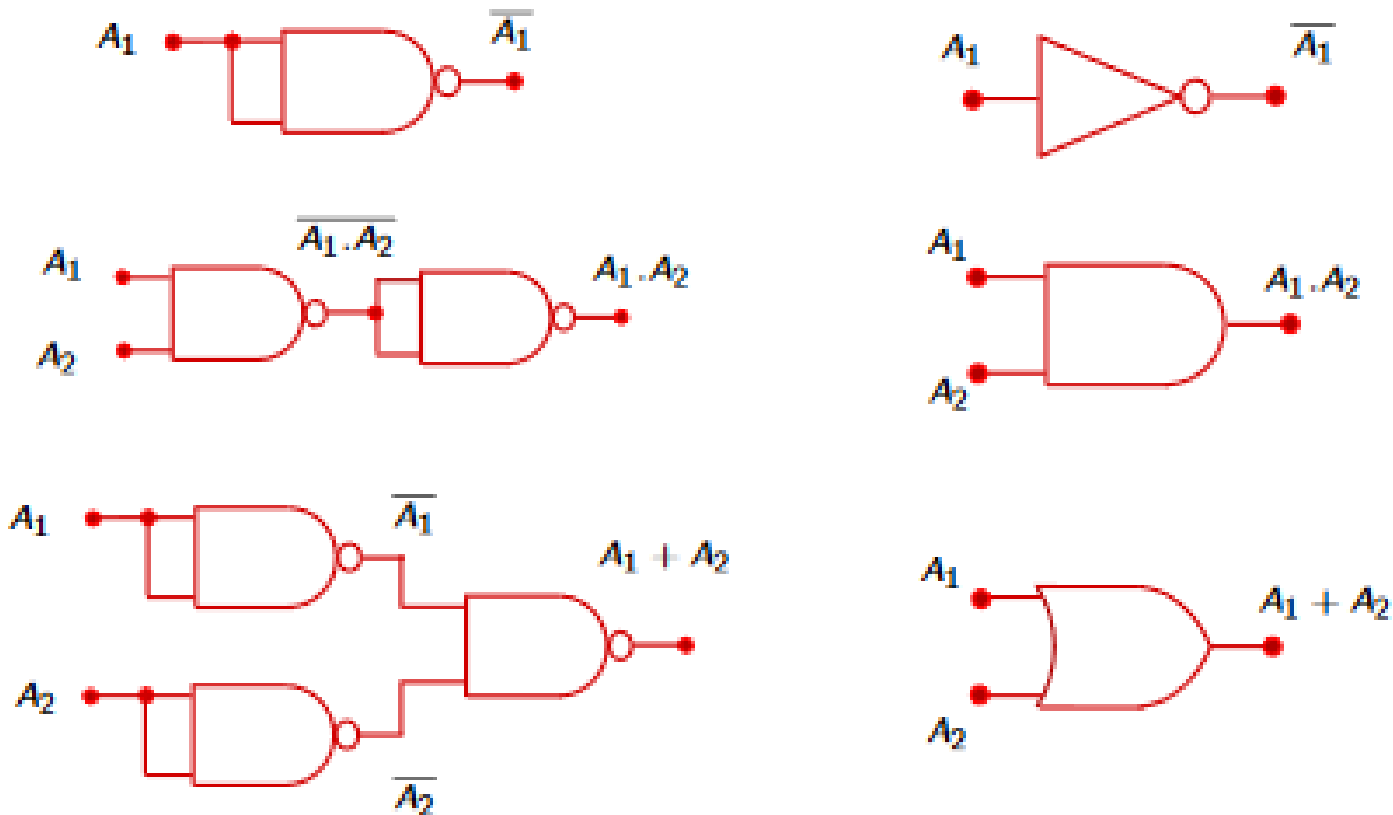
Que é um dos resultados do teorema de Morgan que veremos a seguir...

Tabela verdade

$A_1$	$A_2$	$\overline{A_1} \cdot \overline{A_2}$	$\overline{A_1 + A_2}$
0	0	1	1
0	1	1	1
1	0	1	1
1	1	0	0

# OPERAÇÕES E PORTAS LÓGICAS

- Utilizando apenas a **porta NAND**, podemos obter as outras três portas básicas:





# OPERAÇÕES E PORTAS LÓGICAS

- **Operação XOR (ou exclusivo):** Definida para duas entradas  $\{A_1, A_2\}$ , ela é definida como:

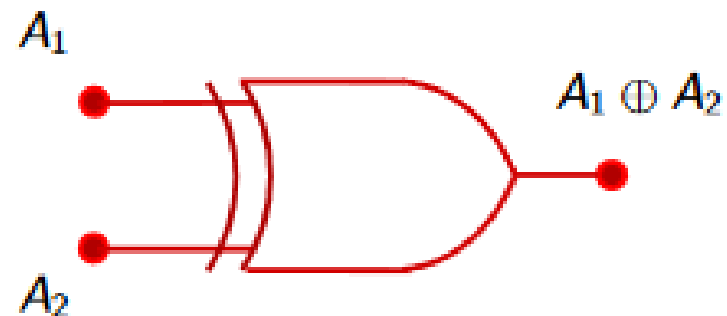
$$f(A_1, A_2) = \overline{A_1} \cdot A_2 + \overline{A_2} \cdot A_1 = A_1 \oplus A_2$$

E vale 1 apenas se as entradas forem diferentes. Para duas entradas temos:

Tabela verdade

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	0
0	1	1
1	0	1
1	1	0

Porta lógica



# OPERAÇÕES E PORTAS LÓGICAS

- **Operação XNOR (coincidência):** Definida para duas entradas  $\{A_1, A_2\}$ , ela é definida como:

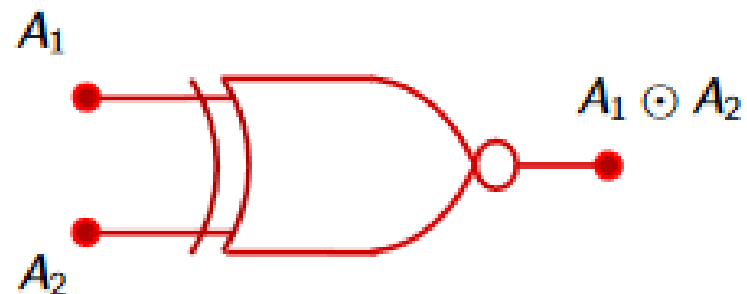
$$f(A_1, A_2) = \overline{A_1} \cdot \overline{A_2} + A_1 \cdot A_2 = A_1 \odot A_2$$

E vale 1 apenas se as entradas forem iguais. Para duas entradas temos:

Tabela verdade

$A_1$	$A_2$	$f(A_1, A_2)$
0	0	1
0	1	0
1	0	0
1	1	1

Porta lógica



- As regras operacionais de minimização utilizando a álgebra de Boole decorrem dos postulados e propriedades a seguir:

- Postulados da complementação

$$\bar{\bar{A}} = A$$

- Postulados da adição

$$A + 0 = A, A + 1 = 1, A + A = A, A + \bar{A} = 1$$

- Postulados da multiplicação

$$A \cdot 0 = 0, A \cdot 1 = A, A \cdot A = A, A \cdot \bar{A} = 0$$

- **Propriedades:** Comutativa, associativa e distributiva são válidas para a **adição** e a **multiplicação**.

# TEOREMA DE DE MORGAN

- O seguinte teorema é importante pois permite simplificar expressões booleanas  $\Rightarrow$  minimização

## Teorema de De Morgan

As seguintes igualdades são verdadeiras :

- $\overline{A \cdot B \cdot C \cdot \dots \cdot N} = \bar{A} + \bar{B} + \dots + \bar{N}$
- $\overline{A + B + C + \dots + N} = \bar{A} \cdot \bar{B} \cdot \dots \cdot \bar{N}$

- Exemplo 1 :** Minimize a expressão sem utilizar o teorema.

$$\begin{aligned}
 \bar{A}\bar{B} + \bar{A}B + A\bar{B} &= \bar{A}(B + \bar{B}) + A\bar{B} \\
 &= \bar{A}(1 + \bar{B}) + A\bar{B} \\
 &= \bar{A} + (A + \bar{A})\bar{B} \\
 &= \bar{A} + \bar{B}
 \end{aligned}$$

**TEOREMA DE DE MORGAN**

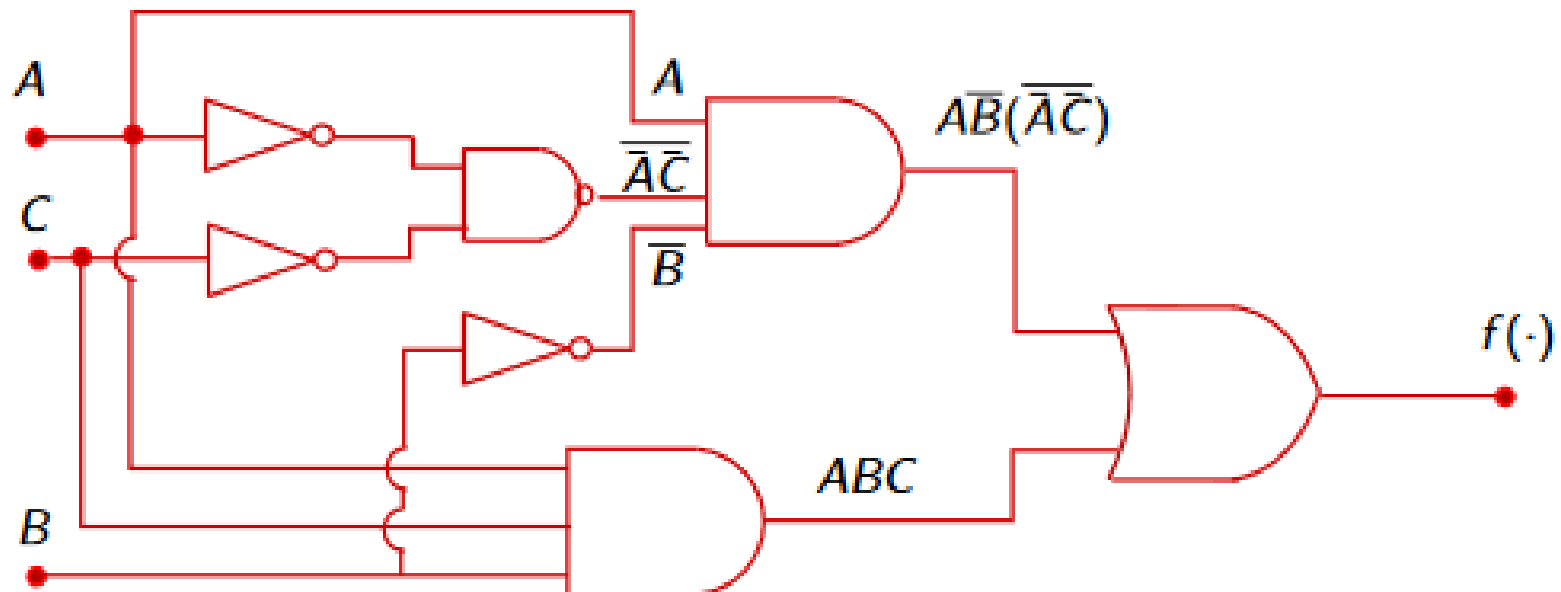
- **Exemplo 2 :** Minimize a mesma expressão utilizando o teorema de De Morgan.

$$\begin{aligned}
 \bar{A}\bar{B} + \bar{A}B + A\bar{B} &= \bar{A}(B + \bar{B}) + A\bar{B} \\
 &= \overline{\bar{A} + A\bar{B}} \\
 &= \overline{A.(\bar{A} + B)} \\
 &= \overline{AB} \\
 &= \bar{A} + \bar{B}
 \end{aligned}$$

- **Exemplo 3 :** Minimize a seguinte expressão

$$\begin{aligned}
 ABC + A\bar{B} + A\bar{C} &= A(BC + \bar{B} + \bar{C}) \\
 &= A(BC + \overline{\bar{B} + \bar{C}}) \\
 &= A(BC + \overline{BC}) \\
 &= A
 \end{aligned}$$

- Descreva a expressão lógica que representa o circuito a seguir



- A expressão lógica é dada por

$$f(A, B, C) = A \cdot B \cdot C + A \cdot \overline{B} \cdot (\overline{A} \cdot \overline{C})$$

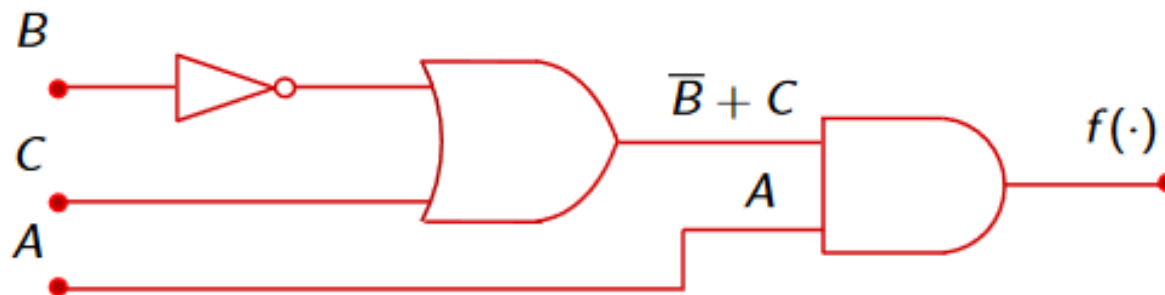
**Minimizar** a expressão de um circuito lógico, significa obter uma outra equivalente com menos termos e operações. Isto implica em **menos portas lógicas e conexões**.

- Como vimos, podemos usar a **álgebra de Boole** para realizar a minimização.
- Neste caso, a **simplificação nem sempre é óbvia**.
- Geralmente, podemos seguir **dois passos essenciais** :
  - colocar a expressão na forma de soma de produtos
  - identificar fatores comuns e realizar a fatoração
- Algumas vezes devemos contar com **habilidade e experiência** para obter uma boa simplificação.

- Utilizando a **álgebra de Boole**, podemos minimizar a expressão da função do exercício anterior

$$\begin{aligned}f(A, B, C) &= A \cdot B \cdot C + A \cdot \overline{B} \cdot (\overline{\overline{A} \cdot \overline{C}}) \\&= A \cdot B \cdot C + A \cdot \overline{B} \cdot (A + C) \\&= A \cdot C \cdot (B + \overline{B}) + A \cdot \overline{B} \\&= A \cdot (C + \overline{B})\end{aligned}$$

- O circuito lógico simplificado é dado por.



A quantidade de portas lógicas foi reduzida de 7 para 3!!!



- A partir do circuito apresentado anteriormente, obtenha a sua tabela verdade e, a partir dela, obtenha a expressão lógica.

<i>A</i>	<i>B</i>	<i>C</i>	<i>f(A, B, C)</i>
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

- Utilizando a tabela, sua expressão lógica é dada por

$$f(A, B, C) = A \cdot \bar{B} \cdot \bar{C} + A \cdot \bar{B} \cdot C + A \cdot B \cdot C$$

embora seja equivalente à função obtida através do circuito, ela possui um número maior de termos.

MEIRELLES, Fernando de Souza. **INFORMÁTICA: NOVAS APLICAÇÕES COM MICROCOMPUTADORES.**, Makron Books. 2005

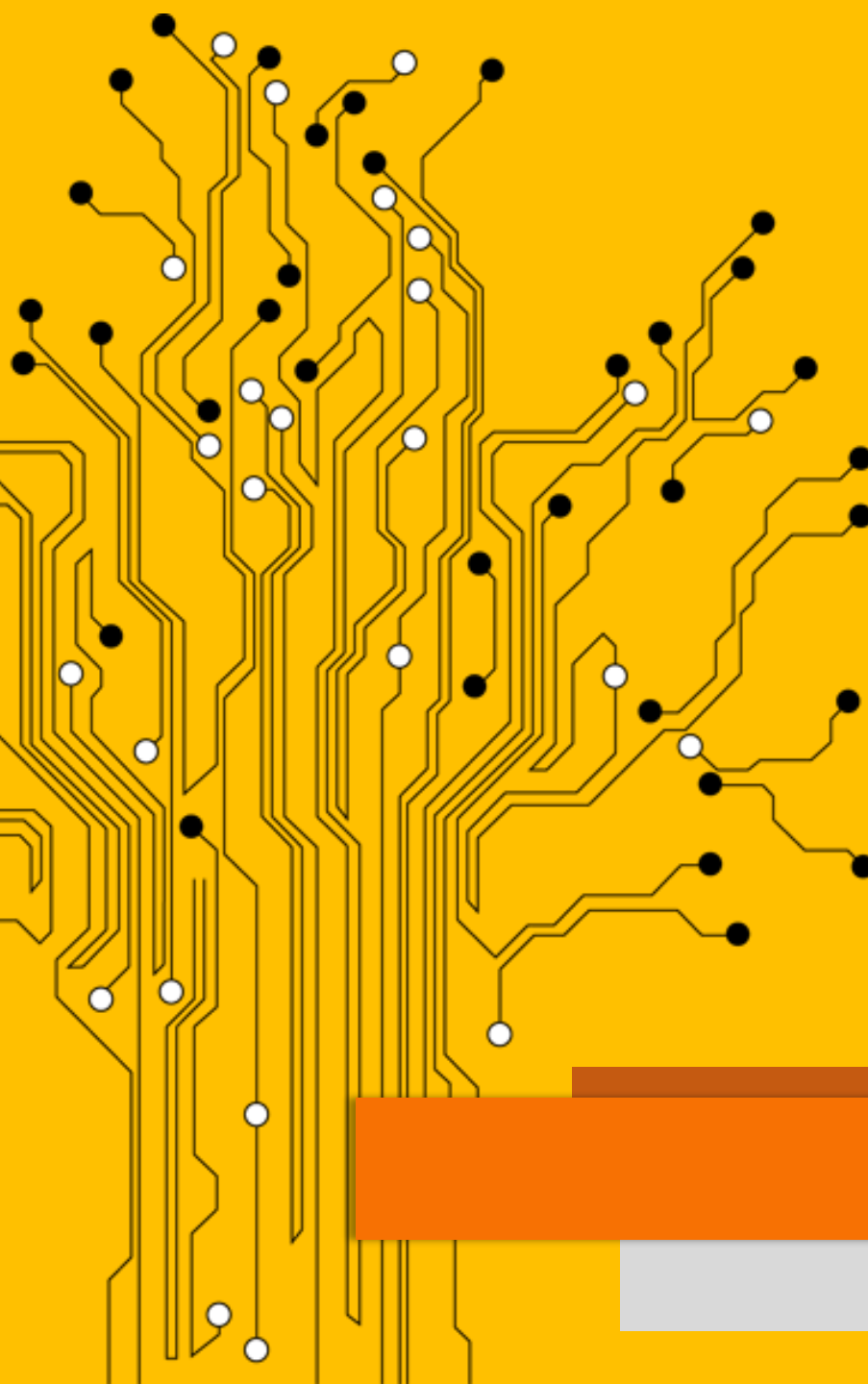
CAPUANO E IDOETA. **Elementos de eletrônica Digital.** Ed Erica

TORRES, Gabriel. **Hardware: curso completo** . 4. ed. Rio de Janeiro: Axcel Books, 2005

BROOKSHEAR, J. Glenn. **Ciência da Computação – Uma Visão Abrangente.** Porto Alegre: Bookman. 2009

CAPRON, Harriet L. **Introdução a Informatica.** Pearson Brasil

PERES, Fernando Eduardo; FEDELI, Ricardo Daniel; POLLONI, Enrico G. F. **Introdução À Ciência da Computação – 2. ed.** Cengage Learning, 2010

An abstract graphic on the left side of the slide, featuring a complex network of yellow lines resembling a circuit board. These lines are interspersed with small black and white dots, creating a dense, branching pattern that extends from the bottom left towards the top left.

# **Tecnologia da informação e comunicação**

## **ÁLGEBRA BOOLEANA**

**FELIPE G. TORRES**