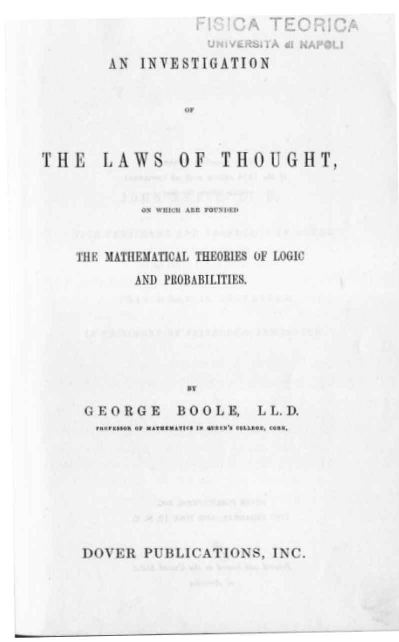
L'*algebra* di Boole

Esiste una definizione generale di ***algebra*** in cui, oggi, viene fatta rientrare la geniale intuizione del matematico inglese **George Boole (1815 - 1864)** che, intorno al **1850** si rese conto, molto prima che l'Elettronica muovesse i primi passi, delle potenzialità di una struttura matematica basata su un sistema minimale (2 soli elementi rappresentativi).
La riscoperta e l'utilizzazione in Elettronica dell'algebra di Boole si deve a **von Neumann**, colui che propose il modello strutturale su cui si basano i moderni computer.

L'algebra binaria di Boole, come richiesto dalla definizione di algebra, prevede un **insieme di supporto, A** e due operatori binari che si chiamano, **"somma" (simbolo ⊕)** e **"prodotto" (simbolo ⊗)** che associano a coppie di elementi di A un elemento dello stesso insieme.

La scelta di questi simboli è fatta per sottolineare, in questa fase, che **questi sono altra cosa rispetto agli omonimi operatori aritmetici**.

Il libro di Boole



AN INVESTIGATION

OF

THE LAWS OF THOUGHT,

ON WHICH ARE FOUNDED

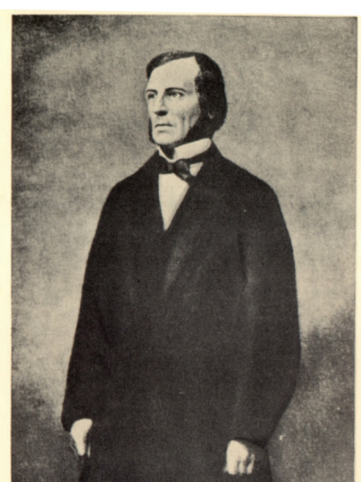
THE MATHEMATICAL THEORIES OF LOGIC AND PROBABILITIES.

BY

GEORGE BOOLE, LL.D.

PROFESSOR OF MATHEMATICS IN QUEEN'S COLLEGE, CORK.

DOVER PUBLICATIONS, INC.



BOOLE

Proprietà degli operatori booleani (assiomi)

- **Le operazioni di somma e prodotto sono commutative**.
  Per ogni coppia di elementi x ed y appartenenti all'insieme A, si ha:

$$(x \oplus y) = (y \oplus x); \quad (x \otimes y) = (y \otimes x)$$

- **La somma è distributiva rispetto al prodotto e questo è distributivo rispetto alla somma.**
  Per ogni coppia di elementi *x* ed *y* appartenenti all'insieme *A*, si ha:

$$(x \oplus (y \otimes z)) = (x \oplus y) \otimes (x \oplus z);$$
$$(x \otimes (y \oplus z)) = (x \otimes y) \oplus (x \otimes z);$$

Proprietà degli operatori booleani (assiomi) (2)

- **O è l'elemento neutro per la somma ed I è l'elemento neutro per il prodotto.**
  Esistono due elementi **O, I** tali che: per ogni elemento *x* appartenente all'insieme A,

$$x \oplus O = x; \quad x \otimes I = x$$

- **Ogni elemento x dell'insieme A ammette un complemento x' che è unico e si indica con ~ x**.
  Per ogni elemento *x* appartenente all'insieme A, esiste un elemento x' tale che:

$$x \oplus x' = I; \quad x \otimes x' = 0$$

  Simboli usati per il complemento sono: **x' , ~ x,** $\overline{x}$

Un modello: algebra delle classi

S è un **insieme finito**.

2^S è l'insieme delle **parti di S** (l'insieme di tutti i possibili sottoinsiemi costituiti da elementi di S),
Esistono **$2^{|S|}$ sottoinsiemi distinti** (vanno dall'insieme vuoto $\varnothing$ ad S), avendo indicato con $|S|$ la ***cardinalità*** di S.

Si può verificare che nell'algebra delle classi sono verificati tutti gli assiomi dell'algebra di Boole

A	$\leftrightarrow$	2^S
$\oplus$	$\leftrightarrow$	$\cup$
$\otimes$	$\leftrightarrow$	$\cap$
$\sim$	$\leftrightarrow$	$\approx$
0	$\leftrightarrow$	$\varnothing$
I	$\leftrightarrow$	S

Teorema di Stone

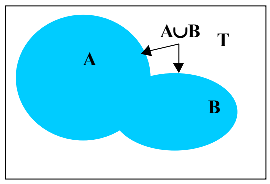
L'importanza di questo modello è legato al teorema di Stone che afferma:

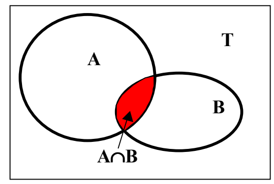
Ogni algebra di Boole su supporto finito è isomorfa ad una algebra delle classi basata su un insieme anch'esso finito. In particolare la cardinalità di A deve essere una potenza del due.

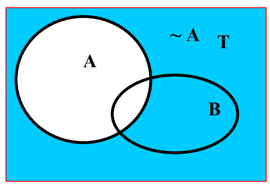
In parole semplici qualunque relazione valida nell'algebra delle classi autorizza l'uso dell'equivalente relazione in un'algebra booleana su supporto finito, come sono in pratica tutte quelle che sono suscettibili di applicazioni e come si vedrà in seguito.

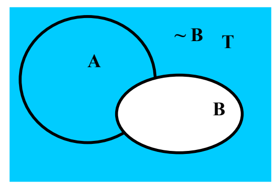
Diagrammi di Venn

Visualizzazione grafica delle operazioni nel modello di Algebra di Boole consistente nell'algebra degli insiemi. T è l'universo della nostra struttura di insiemi, cioè è l'insieme contenente tutti gli elementi possibili. A e B sono due insiemi di punti

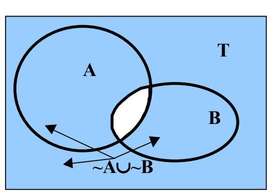




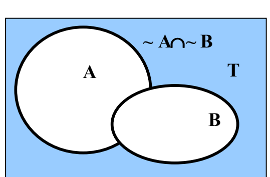




Verifica grafica di teoremi (1)

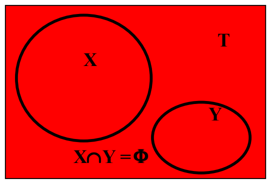


dal confronto con la figura relativa a **A∩B** si ricava
~A∪~B= ~ (A∩B)



dal confronto con la figura relativa a **A∪B** si ricava
~A∩~B= ~ (A∪B)

Verifica grafica di teoremi (2)



Graficamente poi sono ovvie le relazioni

$\sim\Phi = T$
$\sim T = \Phi$

$\mathbf{X \cup \sim X = T}$
$\mathbf{X \cap \sim X = \Phi}$
$\mathbf{X \cup T = T}$
$\mathbf{X \cap T = X}$

Il nostro modello di Algebra di Boole

L'algebra di Boole con cui lavoriamo ha un supporto A costituito da due soli elementi 0 ed 1: A = {0,1}.
Le operazioni interne di questa algebra sono definite dalle seguenti tabelle delle operazioni che corrispondono alle operazioni logiche di OR ed AND e NOT

a	b	a⊕b
0	0	0
0	1	1
1	0	1
1	1	1

a	b	a⊗b
0	0	0
0	1	0
1	0	0
1	1	1

a	ā
0	1
1	0

Esse soddisfano i 4 postulati dell'algebra di Boole, come si può verificare per induzione perfetta cioè semplicemente verificando per tutti i possibili valori delle variabili, operazione macchinosa ma semplice concettualmente.

Ad es. la commutatività di ⊕: a⊕b = b⊕a
va verificata nei 4 casi possibili a=b=0, a=0 b=1, a=1 b=0 , a=b=1.
Per a=b=0 si ha a⊕b = 0 b⊕a=0 e quindi a⊕b = b⊕a
Per a=b=1 si ha a⊕b = 1 b⊕a=1 e quindi a⊕b = b⊕a
Per a=1 b=0 si ha a⊕b = 1 b⊕a=1 e quindi a⊕b = b⊕a e così via.

L'algebra delle porte ed il teorema di Stone

E' facile verificare che questa algebra è isomorfa all'algebra delle classi sui sottoinsiemi di un insieme costituito da un solo elemento, diciamolo a: S= {a} di cardinalità |S| = 1.

L'insieme delle parti di S è : $\mathcal{P}(S)= \{\emptyset, \{a\}\}$ di cardinalità $2^{|S|} = 2$.

E' facile verificare che l'algebra $(\mathcal{P}(S), \cup,\cap)$ è un'algebra di Boole con elementi neutri $\emptyset$ e {a}=S per le operazioni $\cup$ e $\cap$ (ad esempio mediante induzione completa).

L'isomorfismo, di cui il teorema di Stone garantisce l'esistenza, può essere così costruito $\cup \longleftrightarrow \oplus$

$\emptyset \longleftrightarrow 0$

$\{a\} \longleftrightarrow 1$ $\cap \longleftrightarrow \otimes$

A titolo di esempio la distributività di $\oplus$ rispetto a $\otimes$:

$a \otimes (b \oplus c)= (a\otimes b) \oplus (a \otimes c)$ $\forall\, a,b,c \in \{0,1\}$ si traduce nella

$A \cap (B \cup C)= (A \cap B) \cup (A \cap C)$ $\forall\, A,B,C \in \mathcal{P}(S)=\{\emptyset, \{a\}\}$

ambedue vere nelle rispettive algebre.

Un esempio di algebra di Boole a 4 elementi

Un modello di Algebra di Boole con supporto finito e più di due elementi B = **{0,1,a,b}** è :

(B,+,•) dove le operazioni + e • sono definite dalle tabelle seguenti:

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

•	**0**	**a**	**b**	**1**
0	0	0	0	0
a	0	a	0	a
b	0	0	b	b
1	0	a	b	1

Evidentemente sarà $b= \bar{a}$.

Per rendersi conto del fatto che questa è un'algebra di Boole, basta verificare che le operazioni così definite soddisfano gli assiomi dell'algebra di Boole.

Algebra di Boole a 4 elementi (2)

Come verifica del teorema di Stone costruiamo l'algebra delle classi a partire da un insieme di cardinalità |S| = 2, S= {x,y} per verificare che essa è isomorfa all'algebra a 4 elementi costruita. Essa avrà come supporto $\mathcal{P}$(S)= {∅ ,{x},{y}, {x,y}=S} di cardinalità $2^{|S|} = 4$. Gli elementi neutri saranno ∅ ed S.

L'isomorfismo ←→ si può costruire così:

0 ←→∅
a ←→ {x}
b ←→ {y}
1 ←→ {x,y}

Creando poi questa associazione tra coppie di valori booleani ed insiemi, ⇒

(0,0)	∅
(0,1)	{x}
(1,0)	{y}
(1,1)	{x,y}

si può verificare che le operazioni nell'algebra ottenuta si possono definire a partire da operazioni sull'algebra di Boole a due elementi effettuate sui vettori (0,0),(0,1),(1,0),(1,1), componente per componente. Ad es.

a⊕b = (0,1)⊕(1,0) = (1,1) = 1 ←→ {x}∪{y} = {x,y} = S

a⊗b = (0,1) ⊗ (1,0) = (0,0) = 0 ←→ {x}∩ {y} = ∅

che verificano che b = ā cioè b ed a sono uno il complemento dell'altro.

Algebra di Boole ad 8 elementi

In modo analogo possiamo costruiamo l'algebra delle classi a partire da un insieme di cardinalità 3, S= {a,b,c}. Essa avrà come supporto $\mathcal{P}$(S)= {∅ ,{a},{b}, {a,b}, {a,c},{b,c}, {a,b,c} = S}, di cardinalità $2^3 = 8$, ed elementi neutri ∅ ed S. Anche qui le operazioni si possono definire a partire da operazioni sull'algebra di Boole a due elementi effettuate sui vettori della tabella, componente per componente. Ad es.

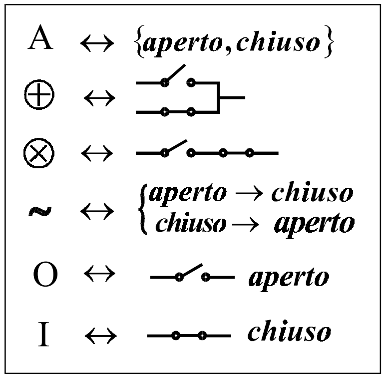
(0,1,1) ⊕ (1,0,1) = (1,1,1) ←→{a,b}∪{c,a}= {a,b,c}

(0,1,1) ⊗ (1,0,1) = (0,0,1) ←→{a,b}∩{c,a}= {a}

come risulta dal modello insiemistico che come si vede è isomorfo al modello di algebra definita mediante terne di valori booleani componente per componente. Generalizzando questa costruzione resta anche dimostrato il teorema di Stone per algebre di Boole di cardinalità qualsiasi(sempre potenze di 2).

c,b,a	
(0,0,0)	∅
(0,0,1)	{a}
(0,1,0)	{b}
(0,1,1)	{a,b}
(1,0,0)	{c}
(1,0,1)	{c,a}
(1,1,0)	{c,b}
(1,1,1)	{a,b,c}

L’algebra degli interruttori

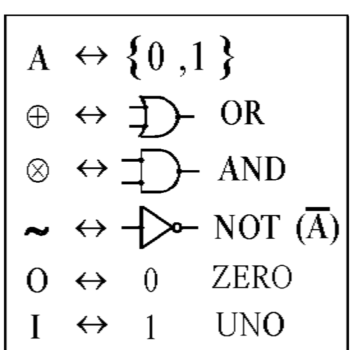


“**Somma**”:
Il collegamento tra ingressi ed uscita è stabilito quando **almeno uno** degli interruttori è “chiuso”.

“**Prodotto**”:
Il collegamento è stabilito quando entrambi gli interruttori sono “chiusi”

“**Negazione**”:
Un interruttore “chiuso” va nello stato “aperto”, uno “aperto” passa allo stato “chiuso”.

Algebra delle porte



OR:
Uscita ad “1” se almeno uno degli ingressi è ad “1”

AND:
Uscita ad “1” se entrambi gli ingressi sono ad “1”

NOT:
Uscita a “0” se l’ingresso è ad “1”, uscita ad “1” se l’ingresso è a “0”

Variabili e costanti booleane

Tutti i simboli matematici consueti, possono essere usati per indicare uno dei due valori booleani, (O, I); es. : A, B, C, oppure x, y, z, ..., oppure, ancora, x_0, x_1, x_2,.......

Se un simbolo è associato sempre allo stesso valore booleano, esso rappresenta una ***costante***. Se, invece esso può assumere volta per volta l'uno o l'altro dei due possibili valori, esso rappresenta una ***variabile*** booleana.

Espressioni booleane

Una ***espressione booleana*** è composta esclusivamente da **costanti** e **variabili booleane**, legate fra loro da **operatori booleani**.
Esempi: $(A \oplus B)$; $(X \oplus Y) \otimes (C \oplus D)$; $(\sim Z) \otimes (x_0 \oplus x_1)$
Le parentesi eliminano qualsiasi ambiguità sul significato degli operatori.
Poiché un'espressione booleana può essere indicata con un unico simbolo, si possono scrivere relazioni in cui gli operatori booleani si applicano, non solo a costanti o variabili, ma ad intere espressioni.

Un'operazione tra due o più espressioni booleane è ancora un'espressione booleana.
Si può dare una definizione precisa di una espressione booleana ***ben formata***, facendo ricorso ad un modello di definizione ricorsiva o induttiva che incontreremo spesso in Informatica.

Definizioni ricorsive o induttive (1)

Una definizione ***ricorsiva*** coinvolge i seguenti passi:

- Una o più regole base per definire gli oggetti semplici (elementari o non composti)
- Una o più regole induttive che serviranno a costruire o definire oggetti complessi

- Gli oggetti sono tutti e soli quelli che si costruiscono mediante i passi 1 e 2. Il secondo va eventualmente iterato.

Definizioni ricorsive (2)

<u>Definizione di fattoriale</u>

- $1! = 1$
- $(n+1)! = (n+1)\times n!$

<u>Definizione di espressione aritmetica sugli interi:</u>

1. espressioni aritmetiche elementari sono:
   - i nomi delle variabili
   - i numeri interi
   - le costanti
2. se e_1 ed e_2 sono espressioni aritmetiche ben formate lo sono anche le seguenti:
   $(- e_1)$ $(e_1 + e_2)$ $(e_1 - e_2)$ $(e_1 \times e_2)$ (e_1 / e_2)
3. espressioni aritmetiche ben formate sugli interi sono tutte e sole quelle che si costruiscono mediante i passi 1 e 2. Il secondo va eventualmente iterato.

Definizioni ricorsive (3)

Definizione di espressione logica o Booleana

1. espressioni booleane elementari sono:
   - i nomi di variabili a,b,c…x,y,z, a_1 ….
   - le costanti : 0 ed 1
2. se e_1 ed e_2 sono espressioni Booleane ben formate, lo sono anche le seguenti, scritte in due formalismi diversi ma equivalenti:

(e_1^*)	$(\sim e_1)$
$(e_1 \mid e_2)$	$(e_1 \oplus e_2)$
$(e_1 \& e_2)$	$(e_1 \otimes e_2)$

3. espressioni ben formate sono tutte e sole quelle che si costruiscono mediante i passi 1 e 2. Il secondo va eventualmente iterato.

Definizioni ricorsive (4)

Definizione di espressione booleana sugli insiemi

<u>Nel modello insiemistico</u>, con la relativa simbologia :

espressioni booleane sono:

1. i nomi delle variabili A,B,C….X,Y,Z…
2. le costanti : ∅ e T
3. se e_1 ed e_2 sono espressioni sono anche espressioni
   a) $(\sim e_1)$
   b) $(e_1 \cup e_2)$
   c) $(e_1 \cap e_2)$
4. espressioni booleane su insiemi sono tutte e solo quelle che si costruiscono mediante i passi 1 e 2. Il secondo va eventualmente iterato.

Semplificazione della scrittura delle espressioni

Una opportuna **priorità assegnata agli operatori** permette una prima semplificazione della scrittura delle espressioni booleane.

- La **massima priorità va all'operatore di negazione "~"**,
- quella **intermedia va all'operatore prodotto "⊗"**
- la **più bassa a quello di somma "⊕"**.

Queste regole consentono spesso di eliminare le parentesi senza introdurre ambiguità.
Si può ad esempio scrivere $A \oplus B \otimes C \otimes D \oplus \sim Z$ senza che possano nascere equivoci sulla successione di operazioni da effettuare.

Semplificazioni dei simboli

Ulteriore semplificazione della scrittura:

- **NOT** si indica, quando è possibile, con una **linea sul simbolo della variabile**:"$\bar{A}$" invece di "**~A**", altrimenti si scrive A*.
- **AND** si indica con "•" o si omette del tutto: **A ⊗ B** si scrive, **A•B** o **A B**
- **OR** si indica con "+": **A ⊕ B** si scrive più semplicemente, **A + B**.

In Elettronica "⊕" è associato ad una operazione diversa dall'OR.

Principio di “dualità”

Gli **assiomi fondamentali dell'algebra di Boole** sono validi, anche tra **espressioni booleane**, oltre che tra variabili.

Dall'esame degli assiomi (scrittura semplificata), si deduce che nell'algebra di Boole vale un "***principio di dualità***".
Dalla **proprietà commutativa di OR ed AND**:

$$x + y = y + x; \quad x\,y = y\,x$$

e quella **distributiva** di ciascun operatore rispetto all'altro:

$$x + y\,z = (x + y)\,(x + z); \quad x\,(y + z) = x\,y + x\,z;$$

si deduce che**: Si può passare dalla prima alla seconda relazione scambiando l'operatore OR con quello AND e viceversa.**

"0" è l'elemento neutro per l'OR: $x + 0 = x$
"1" è l'elemento neutro per l'AND: $x\ 1 = x.$
La **definizione di NOT** è: $x + x^* = 1; \ x\ x^* = 0;$
Per passare da una relazione all'altra basta sostituire l'operatore OR a quello AND e viceversa, invertendo nel contempo il valore delle costanti (0 con 1 ed 1 con 0).

Scheda riassuntiva

relazioni e teoremi dell'algebra di Boole

	postulati	duali	
P1	$x \oplus y = y \oplus x$	$x \otimes y = y \otimes x$	commutatività
P2	$(x \oplus y) \otimes z = (x \otimes z) \oplus (y \otimes z)$	$(x \otimes y) \oplus z = (x \oplus z) \otimes (y \oplus z)$	distributività
P3	$x \oplus 0 = x$	$x \otimes 1 = x$	elementi neutri
P4	$x \oplus \sim x = 1$	$x \otimes \sim x = 0$	complemento

teoremi	teoremi duali	
$x \oplus 1 = 1$	$x \otimes 0 = 0$	
$x \oplus x = x$	$x \otimes x = x$	idempotenza
$x \otimes y \oplus x \otimes z = x \otimes (y \oplus z)$	$(x \oplus y) \otimes (x \oplus z) = x \oplus y \otimes z$	associativa
$x \otimes y \oplus x \otimes \sim y = x$	$(x \oplus y) \otimes (x \oplus \sim y) = x$	
$x \oplus x \otimes y = x$	$x \otimes (x \oplus y) = x$	assorbimento
$(x \oplus y) \otimes (x \oplus \sim y) = x$	$(x \otimes y) \oplus (x \otimes \sim y) = x$	

Analogie tra operatori aritmetici e logici (1)

Nel seguito sono sottolineate e riportate in rosso le leggi che non valgono per entrambi.

Commutatività ed associatività degli operatori × + e AND OR ovvero ⊗ ⊕ o & | (completa analogia)

aritmetica elementare	algebra di Boole	algebra di Boole
$x \times y = y \times x$	$x \otimes y = y \otimes x$	$x \& y = y \& x$
$x+y = y+x$	$x \oplus y = y \oplus x$	$x \mid y = y \mid x$
$(x \times y) \times z = x \times (y \times z)$	$(x \otimes y) \otimes z = x \otimes (y \otimes z)$	$(x \& y) \& z = x \& (y \& z)$
$(x+y)+z = x+(y+z)=x+y+z$	$(x \oplus y) \oplus z = x \oplus (y \oplus z)$	$(x \mid y) \mid z = x \mid (y \mid z)$

Analogie tra operatori aritmetici e logici (2)

Proprietà distributiva di × rispetto a + (di AND rispetto ad OR) (di ⊗ rispetto a ⊕) (di & rispetto a |). Completa analogia

$x \times (y+z) = (x \times y)+(x \times z)$	$x \otimes (y \oplus z) = (x \otimes y) \oplus (x \otimes z)$	$x \& (y \mid z) = (x \& y) \mid (x \& z)$

<u>Proprietà distributiva di OR rispetto ad AND</u> (di ⊕ rispetto a ⊗) o (di | rispetto a &) non vale per le corrispondenti operazioni aritmetiche (+ rispetto a ×).

$x+(y \times z) \neq (x+y) \times (x+z)$	$x \oplus (y \otimes z) = (x \oplus y) \otimes (x \oplus z)$	$x \mid (y \& z) = (x \mid y) \& (x \mid z)$

Analogie tra operatori aritmetici e logici (3)

Elemento neutro per l'operatore × è 1. Per analogia il neutro per AND si designa con 1.
Elemento neutro per l'operatore + è 0 per OR è 0. Si ottiene cosi l'analogia completa

$x\times 1 = x$	$x\otimes 1 = x$
$x+0 = x$	$x\oplus 0 = x$

Gli elementi neutri sono uno il complemento dell'altro.

$-1\neq 0$	$1^{*} = 0$
$-0\neq 1$	$0^{*} = 1$

Analogie tra operatori aritmetici e logici (4)

Legge di annullamento L'analogia c'è solo per $\times,\otimes$ e non per $+,\oplus$

$x+1\neq 1$	$x\oplus 1 = 1$
$x\times 0 = 0$	$x\otimes 0 = 0$

Legge della doppia negazione (completa analogia)

$-(-x) = x$	$x^{**} = x$	$\sim\sim x = x$

La ***legge di idempotenza*** *(o assorbimento)* che caratterizza le algebre di Boole non ha invece l'analogo nell'aritmetica ordinaria

$x\times x \neq x$	$x\otimes x = x$	$x\&x = x$
$x+x \neq x$	$x\oplus x = x$	$x\vert x = x$

Analogie tra operatori aritmetici e logici (5)

Le seguenti formule, utili per la riduzione di espressioni nell'algebra di Boole, non valgono per l'aritmetica elementare:

aritmetica elementare	algebra di Boole	algebra di Boole
$x + (x \times y) \neq x$	$x \oplus (x \otimes y) = x$	$x \mid (x \& y) = x$
$x \times (x + y) \neq x$	$x \otimes (x \oplus y) = x$	$x \& (x \mid y) = x$
$x + (x \times y) \neq x$	$x \oplus (x^* \otimes y) = xy$	$x \mid (\sim x \& y) = x \mid y$
$x \times (x^* + y) \neq x$	$x \otimes (x^* \oplus y) = x \oplus y$	$x \& (\sim x \mid y) = x \& y$

Dimostrazioni dagli assiomi di Boole (1)

dimostrare che $\mathbf{x = x \otimes x}$

$x = x \otimes 1$	P3 Proprietà dell'elemento neutro per $\otimes$
$x = x \otimes \mathbf{(x \oplus \sim x)}$	P4 Proprietà dell'opposto $\mathbf{x \oplus \sim x = 1}$
$x = (x \otimes \mathbf{x}) \oplus (x \otimes \sim \mathbf{x})$	P2 Proprietà distributiva di $\otimes$ rispetto a $\oplus$
$x = (x \otimes \mathbf{x}) \oplus 0$	P4 Proprietà dell'opposto $x \otimes \sim \mathbf{x = 0}$
$x = (x \otimes \mathbf{x})$	P3 Proprietà del neutro rispetto a $\oplus$ **:** $\mathbf{z \oplus 0 = z}$

dimostrare che $\mathbf{x = x \oplus x}$

$x = x \oplus 0$	Proprietà dell'elemento neutro per $\oplus$
$x = x \oplus \mathbf{(x \otimes \sim x)}$	Proprietà dell'opposto $\mathbf{x \otimes \sim x = 0}$
$x = (x \oplus \mathbf{x}) \otimes (x \oplus \sim \mathbf{x})$	Proprietà distributiva di $\oplus$ rispetto a $\otimes$
$x = (x \oplus \mathbf{x}) \otimes 1$	Proprietà dell'opposto $x \oplus \sim \mathbf{x = 1}$
$x = (x \oplus \mathbf{x})$	Proprietà del neutro rispetto a $\otimes$ **:** $\mathbf{z \otimes 1 = z}$

Dimostrazioni dagli assiomi di Boole (2)

Dimostrare che, x ⊕ 1 = 1;

x ⊕ 1 = x ⊕ (x ⊕ ~x)	Proprietà dell'opposto x ⊕ ~x = 1
x ⊕ 1 = (x ⊕ x) ⊕ ~x	Proprietà associativa di ⊕
x ⊕ 1 = x ⊕ ~x	Teorema già dimostrato x = x ⊕ x
x ⊕ 1 = 1	Proprietà dell'opposto x ⊕ ~x = 1

Dimostrare che, x ⊗ 0 = 0;
si puo' dimostrare analogamente al precedente o mediante il principio di dualità

Teoremi di De Morgan

Il principio di dualità ci dice che la relazione:

$$\overline{x+y} = \bar{x} \cdot \bar{y}$$

che può essere facilmente verificata, **implica la relazione duale**:

$$\overline{x \cdot y} = \bar{x} + \bar{y}$$

Queste due relazioni sono note come **teoremi di De Morgan** e, negando entrambi i membri delle due eguaglianze si ottiene:

$$x+y = \overline{\bar{x} \cdot \bar{y}}$$

$$x \cdot y = \overline{\bar{x} + \bar{y}}$$

Queste due relazioni mettono in evidenza una caratteristica dell'algebra di Boole: **Uno dei due operatori fondamentali (OR e AND) non è indispensabile se esiste il NOT.**

- **L'OR può essere costruito con NOT ed AND.**
- **L'AND può essere ottenuto da NOT e OR.**

Teoremi di De Morgan (2)

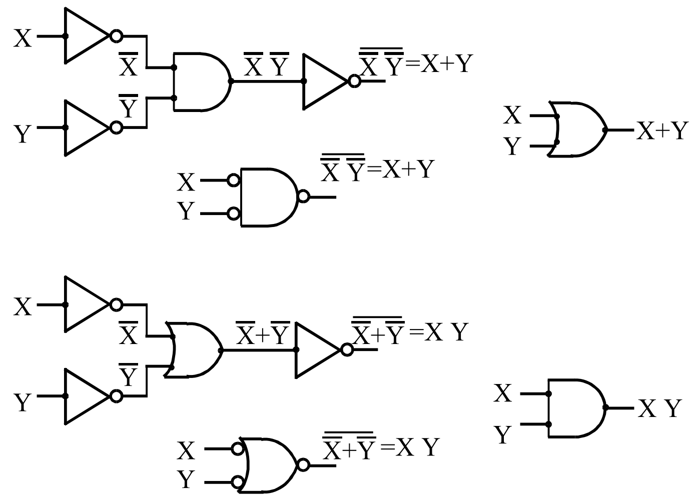


Tabella di verità

OR, AND e NOT, definite in maniera sintetica, possono essere messe in una forma esplicita con la tecnica delle "***tabelle di verità***". L'espressione deriva da un modello di algebra binaria equivalente a quella delle porte logiche in cui i due valori booleani sono "***vero***" ("1") e "***falso***" ("0").

AND

x	y	xy
0	0	0
0	1	0
1	0	0
1	1	1

OR

x	y	x+y
0	0	0
0	1	1
1	0	1
1	1	1

NOT

x	$\overline{x}$
0	1
1	0

E' importante rilevare che la definizione sintetica della due funzioni elementari corrisponde per l'AND all'ultimo rigo e per l'OR al primo delle rispettive tabelle di verità

Assegnazione di una funzione booleana

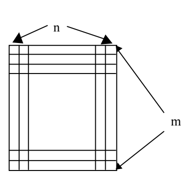
x_2	x_1	x_0	f	g
0	0	0	0	1
0	0	1	1	0
0	1	0	0	0
0	1	1	1	1
1	0	0	0	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	0

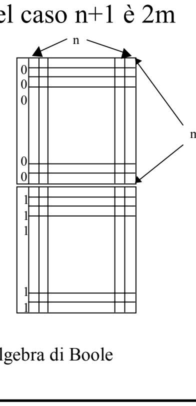
La tabella di verità è la **tecnica standard** per assegnare **una funzione booleana di *n* variabili**.
Poiché 2^n sono le disposizioni dei due valori booleani sugli n posti, se la funzione assegna un valore a tutte le combinazioni binarie, la tabella conterrà **2^n righe** e la funzione si dice, ***completa***.
Si possono definire contemporaneamente più funzioni di n variabili con **tante colonne di assegnazione quante sono le funzioni**.

Numero di righe di una tabella di verità

Il numero di righe di una tabella di verità evidentemente si ricava elencando tutte le possibili combinazioni delle variabili d'ingresso. Poichè i valori che le variabili possono assumere sono 0 o 1, occorre calcolare il numero di combinazioni di 0 ed 1 su n posti. Se ne può dare una dimostrazione induttiva che risulta anche costruttiva, cioè ci fornisce un algoritmo per costruire la tabella di verità completa, mediante raddoppio iterato della stessa.

- nel caso n = 1, il numero di combinazioni di 0 ed 1 su un posto è 2
- se il numero di combinazioni nel caso n è m, nel caso n+1 è 2m





Il numero combinazioni cercato è quindi

$\underbrace{2\cdot 2\cdot 2\cdot ....\ 2}_{\text{n volte}} = 2^n$

Numero di funzioni ad n variabili

Le possibili funzioni booleane distinte di n variabili sono: 2^{2^n}

Di questo risultato possiamo renderci conto mediante il seguente schema grafico, che è anche una dimostrazione *'costruttiva'*. Nel caso n = 2 graficamente si ha

$$2^m = 2^{2^n}$$ colonne

x y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0 0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0 1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1 0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1 1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

$m = 2^n$ elementi

Numero di funzioni ad n variabili

Le possibili funzioni booleane distinte di n variabili sono: 2^{2^n}

Di questo risultato possiamo renderci conto mediante il seguente schema grafico, che è anche una dimostrazione *'costruttiva'*. Nel caso n = 2 graficamente si ha:

colonne m 2^n

x y	f_0	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_8	f_9	f_{10}	f_{11}	f_{12}	f_{13}	f_{14}	f_{15}
0 0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0 1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1 0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1 1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

n elementi

La 16 funzioni di 2 variabili

a	b	f_0 = costante 0	$f_1 = a \cdot b$	$f_2 = p_2$	$f_3 = a$	$f_4 = p_1$	$f_5 = b$	$f_6 = a \oplus b$	$f_7 = a + b$	$f_8 = a \downarrow b$	$f_9 = a \equiv b$	$f_{10} = \bar{b}$	$f_{11} = s_1$	$f_{12} = \bar{a}$	$f_{13} = a \rightarrow b$	$f_{14} = a \mid b$	f_{15} = costante 1
0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
0	1	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
1	0	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1

n (variabili a, b); 2^n (righe)

Le 16 funzioni di 2 variabili (operatori binari) e loro proprietà

[0 0 0 0] 0 costante
[0 0 0 1] (a&b) **AND**
[0 0 1 0] (a&~b) variaz. and
[0 0 1 1] (a) identità
[0 1 0 0] (~a&b) variaz. and
[0 1 0 1] (b) identità
[0 1 1 0] (~a&b)|(a&~b) **XOR**
[1 1 1 0] (~a)|(~b) **NAND**
[0 1 1 1] (b)|(a) **OR**
[1 0 0 0] (~a&~b) **NOR**
[1 0 0 1] (~a&~b)|(a&b) **XNOR** ↔
[1 0 1 0] (~b) not identità
[1 0 1 1] (~b)|(a) Implicazione b→a
[1 1 0 0] (~a) not identità
[1 1 0 1] (~a)|(b) a→b
[1 1 1 1] 1 costante

Sono **commutativi** gli operatori per cui f(x,y)=f(y,x).
Sono **associativi** se f(f(x,y),z)=f(x,f(y,z)). Solo nel caso di associatività ha senso la definizione di porte a più ingressi, come per gli operatori and, or, xor, nand, or, xnor.

Numero di funzioni ad n variabili

Tra tutte le funzioni ad n variabili alcune **si possono ricavare l'una dall'altra mediante permutazione degli ingressi. Ad esempio**

f_{11} [1 0 1 1] (~b)|(a)

f_{13} [1 1 0 1] (~a)|(b)

Si ha: $f_{13}(a, b) = f_{11}(b, a)$.

Alcune funzioni differiscono **solo perche' alcune variabili sono complementate**, ad es. f = (a|b)&c e g = (~a|b)&c sono due funzioni diverse, ma possono essere ricavate l'una dall'altra; si ha difatti,

g(a,b,c) = f(~a,b,c).

Fatte queste riduzioni, anche combinate, il numero si riduce drasticamente:

n	N	N utili
1	4	3
2	16	6
3	256	22
4	65536	402

Teoremi e formule da dimostrare: manipolazioni algebriche (1)

teoremi	teoremi duali	
• **x + 1 = 1**	**x × 0 = 0**	
• **x + x = x**	**x × x = x**	**idempotenza**
• **x × y + x × z = x × (y + z)**	**(x + y) × (x + z)=x + y × z**	**distributiva**
• **x × y + x × ~y = x**	**(x + y) × (x + ~y) = x**	**consenso**
• **x + x × y = x**	**x × (x + y) = x**	**assorbimento**
• **(x + y*) × y = x × y**	**x × y* + y = x + y**	
• **(x + y)(x*+ z)(y + z) = (x + y)(x*+ z)**		
	x y + x*z + y z = x y +x*z	**consenso**

in formato Mat: (x&y)|(~x&z)|(y&z) = (~x&z)|(x&y);

• **x + x*y = x + y** e più in generale **(x + y)(x*+ z) = x z + x*y**

Teoremi e formule da dimostrare: manipolazioni algebriche (2)

Mediante manipolazioni algebriche si ottiene

- **x y + x y* + x* y + x* y* = 1 ;**
  in formato Mat: (x&y)|(x&~y)|(~x&y)|(~x&~y) =1
- **x*yz+xyz+x*y*z*+x*yz* = yz+x*z*;**
  in formato Mat:
  (~x&y&z)|(x&y&z)|(~x&~y&~z)|(~x&y&~z) = (y&z)|(~x&~z)
- **y z + x y + x*y z*= y;**
  in formato Mat: (y&z)|(x&y)|(~x&y&~z)=y
- **x y + x*z + y z = x*z + x y;**
  in formato Mat: (x&y)|(~x&z)|(y&z) = (~x&z)|(x&y);

Prodotti e somme di variabili nelle tabelle di verità

x	y	xy	
0	0	0	P_0
0	1	0	P_1
1	0	0	P_2
1	1	1	P_3

x	y	x+y	
0	0	0	S_0
0	1	1	S_1
1	0	1	S_2
1	1	1	S_3

Un modo per mettere la funzione da una forma tabellare in una forma algebrica è, in linea di principio, quella di interpretare la tabella in termini di operatori elementari.

- ***P_i*** sono i 2^n "**prodotti**" (si chiamano anche, ***clausole***) delle n variabili di un rigo, prese dirette o negate secondo la tabella
- ***S_i*** sono, invece, le "**somme**" delle variabili prese ciascuna complementata rispetto alla tabella.

Con riferimento alle rispettive tabelle si vede che:

- Per l'AND solo P_3 = x y = 1;
- per l'OR solo S_0 = x+y = 0

Generalizzando, tutta l'informazione è contenuta nei *P_i* che valgono "1" oppure negli *S_i* che valgono "0".

Mintermini e *maxtermini*

1. I "prodotti" (AND) P_i che corrispondono nella colonna delle assegnazioni ad un valore logico "1", si chiamano "***mintermini***".
2. Le "somme" (OR), S_i delle variabili, negate rispetto alle configurazioni che corrispondono ad uno "0" della tabella, si dicono "***maxtermini***".

I due tipi di componenti della tabella di verità esprimono ciascuno la condizione duale dell'altro.

- Un mintermine P_i vale 1 soltanto quando le tre variabili assumono la configurazione booleana che compare nell'AND;
- un maxtermine S_i vale 0 soltanto quando i valori booleani delle variabili sono tutti complementari a quelli che compaiono nell'OR.

L'insieme dei mintermini e quello dei maxtermini di una tabella di verità costituiscono il "positivo" ed il "negativo" (in senso fotografico) di una stessa realtà.

La forma algebrica delle funzioni elementari è costituita dall'unico ***mintermine*** dell'**AND** e dall'unico ***maxtermine*** dell'**OR**.

In una funzione completa di n variabili la somma del numero di mintermini e quello dei maxtermini deve essere 2^n

Forme canoniche SOP e POS

Si chiama ***forma canonica disgiuntiva*** della funzione l'OR dei ***mintermini*** (clausole che corrispondono agli "1" della tabella di verità). Con riferimento alla colonna ***f*** della tabella di esempio si ha:

$$f(x_0, x_1, x_2) = x_0\bar{x}_1\bar{x}_2 + x_0x_1\bar{x}_2 + x_0x_1x_2$$

In questo caso si dice anche che la funzione è espressa in forma **SOP** (**Sum of Product**), cioè come somma di prodotti.

Si chiama ***forma canonica congiuntiva*** della funzione **l'AND dei maxtermini** (OR delle configurazioni delle variabili, ad una ad una negate, che corrispondono agli "0" della tabella di verità).

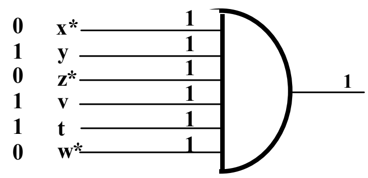
$$f(x_0,x_1,x_2)=(x_0+x_1+x_2)\cdot(x_0+\bar{x}_1+x_2)\cdot(x_0+x_1+\bar{x}_2)$$
$$(\bar{x}_0+x_1+\bar{x}_2)\cdot(x_0+\bar{x}_1+\bar{x}_2)$$

Questa forma si dice di tipo **POS**, cioè ***prodotto di somme***.

La forma canonica SOP realizza la funzione da cui è ricavata

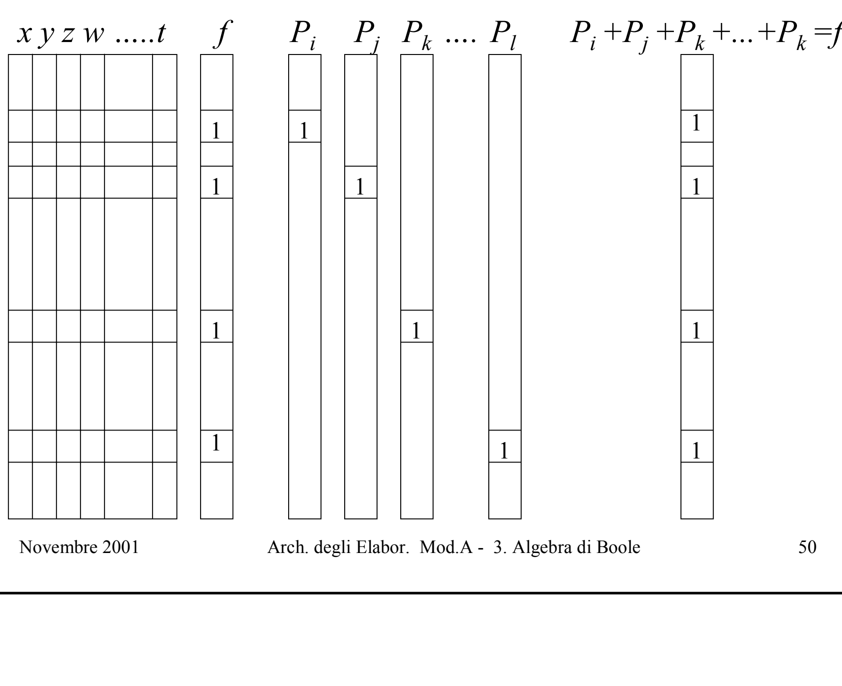
Ognuna delle clausole P_i costituisce una funzione logica che, per le proprietà dell'AND vale 1 se e solo se le variabili assumono la configurazione degli ingressi da cui è stato ricavato quel prodotto. Ad es. dalla riga 010110 della funzione con variabili x,y,z,t,v,w si ricava il prodotto:

$$P = x^* y z^* t v w^*$$

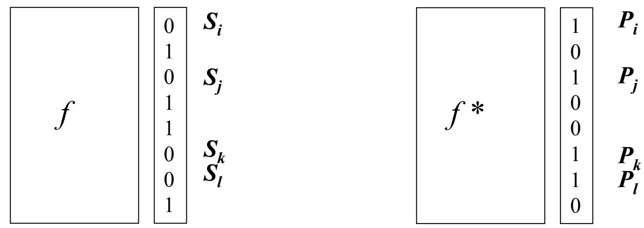


questo vale 1 se e solo se gli ingressi x,y,z,t,v,w valgono rispettivamente 010110.
La funzione logica ottenuta mediante OR di tutte le clausole P_i, per le proprietà dell'OR a più ingressi dunque varrà 1 per quei valori delle variabili e solo per questi, e dunque solo per le righe in cui la funzione da realizzare vale 1.

La forma canonica SOP realizza la funzione da cui è ricavata (2)



La forma canonica POS realizza la funzione da cui è ricavata



$f^{*}=P_i+P_j+P_k+...+P_l$ $\qquad$ $f=f^{**}=(P_i+P_j+P_k+...+P_l)^{*}$

$f=P_i^{*}\bullet P_j^{*}\bullet P_k^{*}\bullet ...\bullet P_l^{*}= S_i\bullet S_j\bullet S_k\bullet ...\bullet S_l$

dove se la riga i è 010…110 sarà $P_i = x^{*}yz^{*}...vtw^{*}$ e quindi per DeMorgan $S_j = P_i^{*} = x+y^{*}+z+…+v^{*}+t^{*}+w$

Per la riga	0	1	0	…	1	1	0	*sarà*
$S_j \rightarrow$	x	y^{*}	z	….	v^{*}	t^{*}	w	

Completezza funzionale di un set di operatori binari

Con lo sviluppo in forma POS o SOP risulta dimostrato che **AND - OR - NOT sono un set funzionalmente completo.**
Si può dimostrare che:
gli unici operatori che da soli risultano funzionalmente completi sono NAND e NOR.
Sistemi completi sono

AND - OR - NOT
NAND
NOR
AND - NOT
OR - NOT
AND - XOR
OR - XOR

Considerazioni conclusive

Le due forme canoniche sono equivalenti.

I teoremi di De Morgan consentono di passare dall'una all'altra.

Per motivi legati alle caratteristiche dei componenti elettronici su silicio, in pratica si usa esclusivamente la forma disgiuntiva.

Proprietà della porta XOR

Il connettivo XOR è commutativo ed associativo

$\mathbf{x \oplus y = x\, y^* + x^*y = (x + y)(x^*+ y^*)}$

$\mathbf{(x \oplus y)^* = x \oplus y^* = x^*\oplus y = x\, y \oplus x^*y^* = (x^*+ y)(x + y^*)}$

$\mathbf{x \oplus x = 0}$ $\mathbf{x \oplus x^* = 1}$

$\mathbf{x \oplus 1 = x^*}$ $\mathbf{x \oplus 0 = x}$

$\mathbf{x\,(y \oplus z) = x\, y \oplus x\, z}$

$\mathbf{x +y = x \oplus y \oplus x\, y = x \oplus x^*y}$

$\mathbf{x \oplus (x + y) = x^*y}$

$\mathbf{x \oplus x\, y = x\, y^*}$

Tecniche di minimizzazione

Minimizzare, ma che cosa?

- Il numero delle funzioni?
- Il costo?
- Il tempo di risposta della funzione?
- ......

La forma canonica di una funzione booleana può contenere apprezzabili ridondanze la cui eliminazione può dare notevoli economie nella progettazione hardware di una rete logica.

Tutti i procedimenti di minimizzazione non fanno altro che applicare le proprietà fondamentali dell'algebra booleana che in ultima analisi derivano dagli assiomi.

Mappe di Karnaugh

Queste mappe sono costruite in modo che le clausole (prodotti di variabili) **che differiscono per il valore diretto e negato di una sola variabile siano associate a caselle contigue**.
Se queste clausole sono **mintermini** (valore 1) nella tabella della funzione, nella mappa di Karnaugh corrispondente ci saranno due "1" in caselle contigue. Poichè tra i due mintermini si può mettere in evidenza il prodotto delle altre variabili da una situazione del tipo:

$$x_0 \overline{x_1} x_2 \overline{x_3} + x_0 \overline{x_1} x_2 x_3$$

si passa a:

$$x_0 \overline{x_1} x_2 (\overline{x_3} + x_3) = x_0 \overline{x_1} x_2$$

Due mintermini della forma canonica vengono sostituiti con una sola clausola che **non contiene la variabile che compariva una volta diretta ed una volta negata** nei due termini di partenza.

Funzione di 3 variabili

x_2	x_1	x_0	f	
0	0	0	0	P_0
0	0	1	1	P_1
0	1	0	0	P_2
0	1	1	1	P_3
1	0	0	0	P_4
1	0	1	1	P_5
1	1	0	0	P_6
1	1	1	1	P_7

x_2 \ $x_1\,x_0$				
	P_0	P_1	P_3	P_2
	P_4	P_5	P_7	P_6

x_2 \ $x_1\,x_0$				
	0	1	1	0
	0	1	1	0

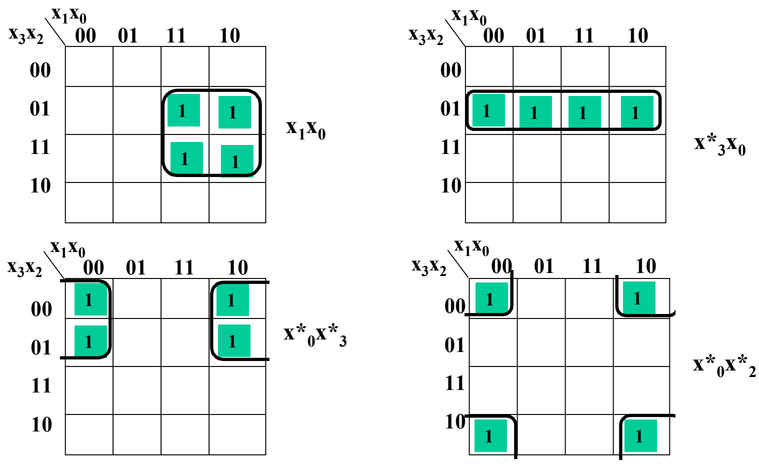
Funzione di 4 variabili

P	x_3	x_2	x_1	x_0	f
0	0	0	0	0	0
1	0	0	0	1	1
2	0	0	1	0	0
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	1
6	0	1	1	0	0
7	0	1	1	1	1
8	1	0	0	0	1
9	1	0	0	1	0
10	1	0	1	0	1
11	1	0	1	1	0
12	1	1	0	0	1
13	1	1	0	1	1
14	1	1	1	0	1
15	1	1	1	1	1

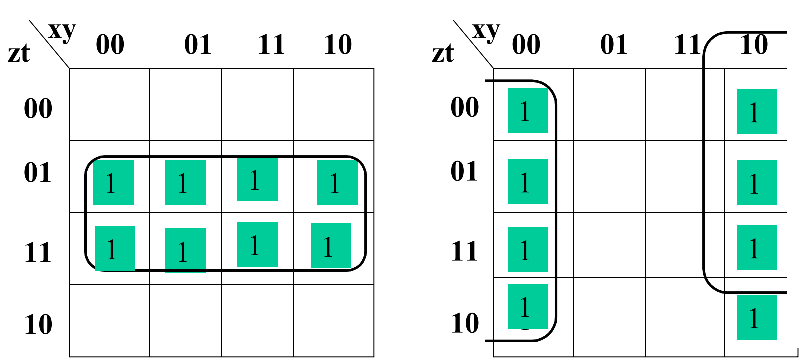
x_3x_2 \ $x_1\,x_0$	00	01	11	10
00	0_{0}	1_{1}	0_{3}	0_{2}
01	0_{4}	1_{5}	1_{7}	0_{6}
11	1_{12}	1_{13}	1_{15}	1_{14}
10	1_{8}	0_{9}	0_{11}	1_{10}

$$f = x_2x_0 + x_3\bar{x}_0 + \cancel{x_3x_2} + \bar{x}_3\bar{x}_1x_0$$

Mappe di Karnaugh a 4 variabili: prodotti a 2 termini: esempi



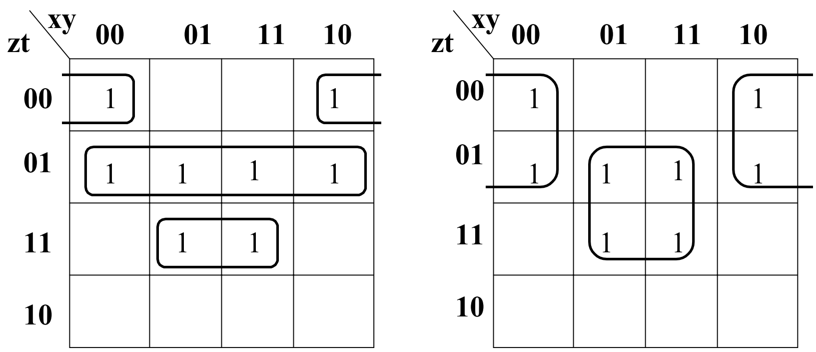
Mappe di Karnaugh a 4 variabili: prodotti a 1 termine



f = t

f = y*

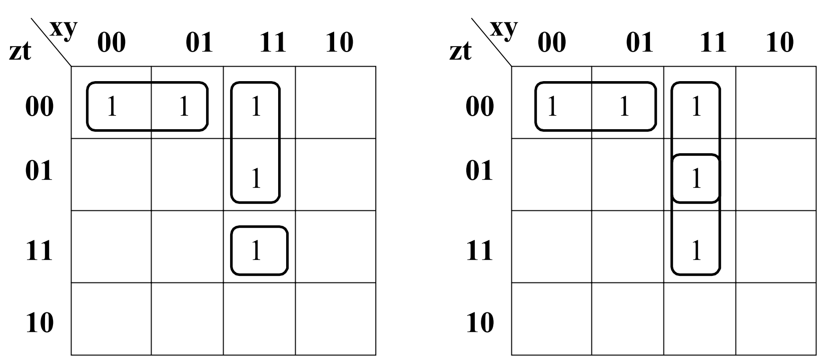
Mappe di Karnaugh: esempi (1)



z*t+yzt+y*z*t* *meglio* → **y*z*+yt**

Il numero di rettangoli che ricoprono gli 1 devono essere in numero minimo ed ognuno deve essere il più esteso possibile

Mappe di Karnaugh : esempi (2)



x*z*t*+xyz*+xyzt *meglio* → **x*z*t*+xyz*+xyt**

A parità di numero i rettangoli che ricoprono gli 1 devono essere delle massime dimensioni possibili

Mappe di Karnaugh : esempi (3)

z \ xy	00	01	11	10
0	1	1		1
1	1	1	1	1

z \ xy	00	01	11	10
0	1	1	0	1
1	1	1	1	1

x*+z+xy* *meglio* → **x*+y*+z**

Più semplicemente:

f* = xyz*

f = f= (xyz*)*= x*+y*+z**

Funzione incompleta

P	x_3	x_2	x_1	x_0	f
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	1
5	0	1	0	1	0
6	0	1	1	0	0
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0

x_3x_2 \ x_1x_0	00	01	11	10
00	1_0	0_1	0_3	1_2
01	1_4	0_5	0_7	0_6
11	x_{12}	x_{13}	x_{15}	x_{11}
10	1_8	0_9	x_{11}	x_{10}

Funzione incompleta (2)

$x_3x_2 \backslash x_1x_0$	00	01	11	10
00	1	0	0	1
01	1	0	0	0
11	x	x	x	x
10	1	0	x	x

$$f = \overline{x}_1\,\overline{x}_0 + \overline{x}_2\,\overline{x}_0$$

Si osservi che in questo come in altri casi è possibile arrivare a diverse forme minime della funzione perché gli "1" possono essere raggruppati in più modi.

Il metodo di Karnaugh diventa rapidamente molto laborioso al crescere delle variabili.
Già con 5 variabili bisogna passare ad una applicazione del metodo in 3 D, disegnando due mappe a 4 variabili, una relativa alla x_4 diretta e l'altra a $\mathbf{x_4}$*, immaginandole sovrapposte in verticale.

Metodo di Quine-Mc Cluskey

- E' un metodo che non è, in linea di principio, **limitato ad un numero ridotto di variabili** ed è, per questo utilizzato nei programmi di minimizzazione automatizzata.
- Anche questa tecnica può, però, richiedere **tempi di elaborazione esponenzialmente crescenti** in funzione del numero di variabili.
- Il metodo si basa sul concetto di "***implicante***" che può essere definito in maniera matematicamente ineccepibile e senza nessun riferimento alle mappe di Karnaugh. Il modo più semplice ed intuitivo è, però, quello di riferirsi alle mappe per dare concretezza ai concetti.

Un prodotto ***p*** di ***k*** variabili, corrisponde a $\boldsymbol{2^{n-k}}$ caselle adiacenti contenenti altrettanti "1". Queste, che appaiono nelle mappe come rettangoli, si chiamano anche ***sottocubi*** del ***cubo*** che è l'intera mappa. Le celle di un sottocubo corrispondono alle configurazioni delle variabili in cui ***k*** di esse hanno valore costante e le altre ***n-k*** assumono tutti i valori possibili.

Implicanti ed *implicanti primi*

Un prodotto *p* si dice *implicante* di una funzione *f* se questa assume il valore "1" in almeno tutti i vertici del sottocubo (caselle del rettangolo) corrispondente al prodotto.
Si dice che *p implica f* (p→f). Si usa anche dire che *f copre p*.

Se ***f*** è una funzione e $p_1, p_2, .. p_h$ sono tutti i suoi ***implicanti*** nel senso che non esistono altre caselle (si chiamano anche "vertici") in cui $f = 1$ si può scrivere: $f = p_1 + p_2 + \dots p_h$
Che rappresenta nient'altro che l'espressione della funzione attraverso i suoi **mintermini che sono evidentemente tutti implicanti della funzione**.

Si chiama "***implicante primo***" della funzione un implicante per il quale non esista un altro implicante che lo copra.
In termini di mappe un ***implicante primo*** **corrisponde ad un "1" isolato o ad un rettangolo di "1" che non sia contenuto in uno più grande**.

Il metodo di **Quine-Mc Cluskey** si applica in due tempi. In una prima fase si cercano gli "implicanti primi" della funzione.

Fase A del procedimento

Riferiamoci alla funzione SOP:

$$z = \Sigma_4\ (0, 2, 4, 6, 7, 9, 11, 15).$$

La funzione ha 4 variabili e quelli tra parentesi sono gli indici dei prodotti p_i che corrispondono ai mintermini (o implicanti) della tabella di verità.
La prima fase del metodo prevede che i prodotti vengano raggruppati in ***classi***, in funzione del numero di "1" nella configurazione delle variabili.
Per n variabili le classi distinte possono essere al massimo n+1.

Ciascuna delle configurazioni di una classe viene poi confrontata con ognuna di quelle della classe successiva alla ricerca di configurazioni che **differiscano per un solo bit** e che possono, pertanto, essere sostituite da una configurazione con tutti i bit uguali ed un trattino "-" al posto del bit che costituiva la differenza tra i due.
Il trattino indica che quella variabile non sarà presente nel prodotto risultante.

Fase A-1

x_3	x_2	x_1	x_0	f
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
1	0	1	1	1
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

i	x_3	x_2	x_1	x_0	
0	0	0	0	0	<
2	0	0	1	0	<
4	0	1	0	0	<
6	0	1	1	0	<
9	1	0	0	1	<
7	0	1	1	1	<
11	1	0	1	1	<
15	1	1	1	1	<

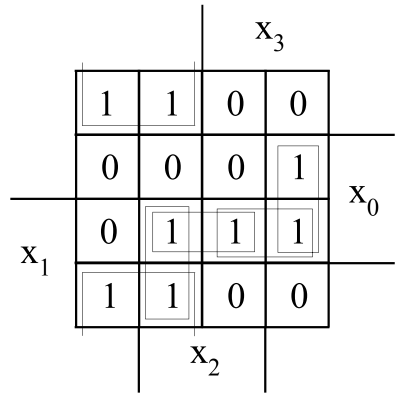
Fase A-2

i	x_3	x_2	x_1	x_0	
0	0	0	0	0	<
2	0	0	1	0	<
4	0	1	0	0	<
6	0	1	1	0	<
9	1	0	0	1	<
7	0	1	1	1	<
11	1	0	1	1	<
15	1	1	1	1	<

	x_3	x_2	x_1	x_0	
0/2	0	0	-	0	<
0/4	0	-	0	0	<
2/6	0	-	1	0	<
4/6	0	1	-	0	<
6/7	0	1	1	-	$\overline{x}_3 x_2 x_1$
9/11	1	0	-	1	$x_3 \overline{x}_2 x_0$
7/15	-	1	1	1	$x_2 x_1 x_0$
11/15	1	-	1	1	$x_3 x_1 x_0$

	x_3	x_2	x_1	x_0	
0/2/4/6	0	-	-	0	$\overline{x}_3 \overline{x}_0$

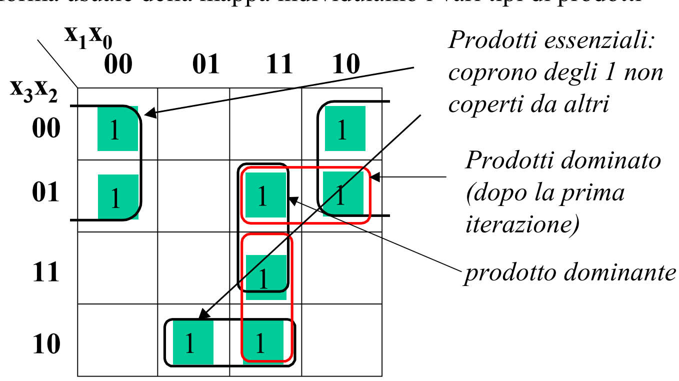
Implicanti primi con le mappe di Karnaugh



$$f = \bar{x}_3\bar{x}_0 + x_2x_1x_0 + \bar{x}_3x_2x_1 + x_3x_1x_0 + x_3\bar{x}_2x_0$$

Mappe di Karnaugh: implicanti primi

Tornando alla forma usuale della mappa individuiamo i vari tipi di prodotti



minimizzazione con Karnaugh

gli altri implicanti primi

Fase B
Tabella di "*copertura*"

	0	2	4	6	7	9	11	15
$\bar{x}_3\bar{x}_0$	X	X	X	X				
$x_3\bar{x}_2x_0$						X	X	
$\bar{x}_3x_2x_1$				X	X			
$x_3x_1x_0$							X	X
$x_2x_1x_0$					X			X

	7	9	11	15	
					$\bar{x}_3\bar{x}_0+$
$x_3\bar{x}_2x_0$		X	X		$+x_3\bar{x}_2x_0+$
$\boxed{\bar{x}_3x_2x_1}$	$\boxed{X}$				
$\boxed{x_3x_1x_0}$			X	$\boxed{X}$	
$x_2x_1x_0$	X			X	$+x_2x_1x_0$

Quine - McKluskey fase B: Tabella di "*copertura*"

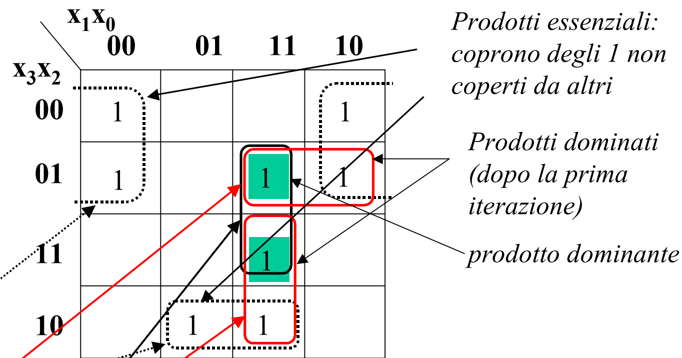
	P_0	P_2	P_4	P_6	P_7	P_9	P_{11}	P_{15}		
	X	X	X	X					essenziale	$\bar{x}_3\bar{x}_0$
						X	X		essenziale	$x_3\bar{x}_2x_0$
				X	X					
							X	X		
					X			X		

	P_7	P_{15}	
	X		dominata
		X	
	X	X	

	P_7	P_{15}		
		X	dominata	
	X	X	essenziale	

$$\bar{x}_3\bar{x}_0 + x_3\bar{x}_2x_0 + x_2x_1x_0$$

Mappe di Karnaugh: implicanti primi dopo l'eliminazione dei prodotti essenziali



$$\bar{x}_3\bar{x}_0 + x_3\bar{x}_2 x_0 + x_2 x_1 x_0$$

Prodotti essenziali e dominanti (minimizzazione)

$$\bar{x}_3 x_2 x_1 + x_3 x_1 x_0$$

gli altri implicanti primi (dominati)

Quine - McKluskey fase B: Tabella di "*copertura*" ciclica

	P_0	P_2	P_4	P_6	P_7	P_9	P_{11}	P_{15}		
	x	x	x	x				x		
		x				x	x			
				x	x	x				
							x	x		
	x		x		x					