Dispensa su

Funzioni Booleane

Jianyi Lin
Università degli Studi di Milano
jianyi.lin@unimi.it

18 novembre 2011

1 Operazioni booleane

In questa sezione introduciamo il concetto di funzione booleana e accenniamo la sintesi di questo tipo di funzione.
L'algebra booleana è stata introdotta dal logico inglese George Boole nel XIX secolo per studiare le operazioni su logiche a due valori di verità, rispettivamente: *vero* e *falso*. Questi corrispondono ai valori di verità che può assumere una proposizione, secondo il principio di *bivalenza*[1].

Sia $B = \{0, 1\}$. Il numero 0 rappresenta il valore di verità *falso*, mentre 1 rappresenta il valore di verità *vero*. Nel seguito indicheremo con le lettere a, b, c, oppure x, y, z, oppure $x_1, x_2, x_3, ...$ le *variabili booleane*, cioè quelle che assumono valori in B.

Sull'insieme B si possono definire alcune operazioni, dette *operazioni booleane*. Ricordiamo che una operazione è una legge di composizione interna, cioè una legge che ad ogni n-upla di elementi di un insieme associa un unico elemento dell'insieme; ad es. l'operazione $+$ associa alla coppia $(2, 3)$ il numero 5.

Le operazioni booleane più importanti sono elencante in Tabella 1. L'o-

Tabella 1: I 3 operatori booleani di base

Nome	Termine informatico	Simboli	Esempi
Congiunzione	AND	$\wedge$	$a \wedge b$
Disgiunzione	OR	$\vee$	$a \vee b$
Negazione	NOT	$\neg$,	$\neg a$, $\overline{a}$

perazione AND restituisce 1 se e solo se entrambi gli operandi valgono 1, altrimenti restituisce 0:

$$a \wedge b = \begin{cases} 1 & \text{se } a = b = 1; \\ 0 & \text{altrimenti.} \end{cases}$$

L'operazione OR restituisce 0 se almeno uno dei due operandi vale 1, altrimenti restituisce 0:

$$a \vee b = \begin{cases} 1 & \text{se } a = 1 \text{ oppure } b = 1; \\ 0 & \text{altrimenti.} \end{cases}$$

L'operazione NOT restituisce il valore di verità opposto:

$$\overline{a} = \begin{cases} 1 & \text{se } a = 0 \\ 0 & \text{altrimenti.} \end{cases}$$

L'algebra indotta da queste operazioni sull'insieme B è detta algebra *di Boole* o *booleana* su B.

[1]Il principio di bivalenza equivale sostanzialmente al principio del *terzo escluso* introdotto nella Metafisica di Aristotele, secondo il quale una proposizione, ossia semplicemente una frase, o è vera oppure è falsa: non può essere altrimenti.

Osservazione 1. Notiamo che $a \wedge b = a \cdot b$ (indicando con $\cdot$ la usuale moltiplicazione) e $\overline{a} = 1 - a$ (indicando con $-$ la usuale sottrazione). A cosa corrisponderebbe $a \vee b$ in termini di somma e divisione intera?

Altre operazioni utilizzate in informatica sono: *i*) *disgiunzione esclusiva*[2] (XOR, $\oplus$) restituisce 1 se uno solo dei due operandi vale 1, altrimenti 0; *ii*) NAND è la negazione di AND, quindi restituisce 0 se entrambi gli operandi valgono 1, altrimenti 1; *iii*) NOR è la negazione di OR, quindi restituisce 1 se entrambi gli operandi valgono 0, altrimenti 0.

Esercizio 1. Scrivere l'operazione $a \oplus b$ usando le operazioni $\wedge$, $\vee$, $\bar{}$.

Le operazioni di congiunzione, disgiunzione e negazione godono di importanti proprietà (o leggi), alcune delle quali comuni alle ben note operazioni dell'aritmetica.

Proposizione 1. Valgono le seguenti proprietà:

i) *associatività* di $\wedge$ e $\vee$:
$x \wedge (y \wedge z) = (x \wedge y) \wedge z$
$x \vee (y \vee z) = (x \vee y) \vee z$

ii) *commutatività* di $\wedge$ e $\vee$:
$x \wedge y = y \wedge x$
$x \vee y = y \vee x$

iii) *assorbimento*:
$x \vee (x \wedge y) = x$
$x \wedge (x \vee y) = x$

iv) *distributività* di $\vee$ rispetto a $\wedge$ e di $\wedge$ rispetto a $\vee$:
$(x \wedge y) \vee z = (x \vee z) \wedge (y \vee z)$
$(x \vee y) \wedge z = (x \wedge z) \vee (y \wedge z)$

v) *complementi*:
$x \wedge \overline{x} = 0$
$x \vee \overline{x} = 1$

vi) *Leggi di De Morgan*:
$\overline{(x \wedge y)} = (\overline{x} \vee \overline{y})$
$\overline{(x \vee y)} = (\overline{x} \wedge \overline{y})$

Esercizio 2. Dimostrare una legge di De Morgan verificando che valga per ogni assegnamento possibile alle variabili x e y.

[2] disgiunzione e disgiunzione esclusiva corrispondono a *vel* e *aut* in latino.

2 Formule

Chiameremo *formule booleane* o semplicemente *formule* le espressioni (scritture) ben formate in cui appaiono costanti, variabili e operatori booleani. Intuitivamente, per ben formato si intende che abbiano un senso; ad es. $a \wedge b \wedge (0 \vee \overline{b})$ è ben formato, mentre $a \vee (\wedge b) \vee (1c \wedge \wedge)$ non è ben formato e quindi non ha senso.

Una variabile è chiamata *letterale*, e anche la sua negazione è chiamata *letterale*. Ad es.: se x, y sono variabili $x, \overline{x}, y, \overline{y}$ sono letterali; le espressioni $\overline{0}, \overline{x \vee y}$ non sono letterali. Un *monomio* è una congiunzione di letterali (eventualmente un solo letterale); esempi: $\overline{x}, x \wedge \overline{y}, x \wedge y \wedge \overline{x}$; non è un monomio $\overline{x \wedge y}$. Una formula è in *forma normale disgiuntiva* (DNF) se è una disgiunzione di monomi (eventualmente un solo monomio). Sono esempi di formule in DNF: $x \wedge y \wedge \overline{x}$, $(x \wedge y) \vee \overline{y}$, $(x \wedge x) \vee (y \wedge z)$, $x \vee y$. Non sono formule in DNF: $\overline{x \wedge y}$, $(x \vee y) \vee z$, $x \vee (\overline{x \vee y})$. In sintesi, una formula con m monomi ove il k-esimo monomio è congiunzione di n_k letterali corrisponde a:

$$\bigvee_{k=1}^{m} (\bigwedge_{j=1}^{n_k} \ell_{kj})$$

dove ℓ_{kj} è il j-esimo letterale che appare nel k-esimo monomio.

Osservazione 2. Due formule, non necessariamente in DNF, sono equivalenti quando le stesse assegnazioni di valori alle variabili danno lo stesso valore di verità. Abbiamo indicato questa equivalenza con il simbolo di uguaglianza $=$.

Osservazione 3. Quindi due formule possono essere diverse, cioè con differente scrittura, ma equivalenti. Esempi evidenti di questo fatto sono le Leggi di De Morgan e le altre proprietà.

3 Funzioni booleane

Una funzione booleana è una funzione di variabili booleane a valori booleani. Più precisamente:

Definizione 1. Una *funzione booleana* f è una funzione

$$f : B^n \to B$$

dove $B = \{0, 1\}$ e $B^n = B \times ... \times B$ è il prodotto cartesiano ripetuto n volte. L'intero n viene detto *arietà* di f, ed f viene detta funzione *n-aria*.

Una funzione booleana può essere specificata da una formula booleana oppure dalla sua *tabella di verità*.

Esempio 1. Si pensi alla somma di due numeri in cifre binarie, con il metodo elementare. Fissiamo l'attenzione su una colonna. La funzione f specificata da

$$f(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_3$$

permette di calcolare la cifra da riportare nella riga del risultato della somma di due cifre binarie (x_1 e x_2) in presenza di un eventuale riporto indicato in x_3.

La tabella di verità della funzione n-aria f esplicita per ogni possibile assegnamento di valori (0 e 1) alle variabili $x_1, ..., x_n$ il corrispondente valore (0 o 1).

Esempio 2. La tabella di verità della funzione f dell'esempio precedente è facilmente calcolabile a partire dalla formula booleana: $f(0,0,0) = 0 \oplus 0 \oplus 0 = 0$, $f(0,0,1) = 0 \oplus 0 \oplus 1 = 1$, $f(0,1,0) = ... = 1$ e così via, ottenendo la Tabella 2.

Tabella 2: Tabella di verità di $f(x_1, x_2, x_3) = x_1 \oplus x_2 \oplus x_3$

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
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

Abbiamo visto quindi che avendo una funzione specificata da una formula è immediato ottenere la sua tabella di verità. Questa procedura potrebbe essere chiamata *analisi*. Non è così immediato il procedimento inverso.

4 Sintesi

Problema 1 (*sintesi*)**.** Data la specificazione di una funzione (booleana) f tramite la sua tabella di verità, ottenere una formula che specifica f.

Nella progettazione di circuiti binari un problema centrale è la sintesi minima, volta cioè a minimizzare la dimensione della formula di f; questo si traduce sul piano tecnologico nel minimizzare lo spazio occupato dai circuiti.

Nelle seguenti pagine vediamo tuttavia solo il problema di sintesi, in cui è sufficiente ottenere almeno una formula che specifichi la funzione. Il metodo che illustriamo permetterà di ottenere una formula booleana in DNF a partire dalla tabella di verità. Quando necessario, indicheremo con lettere greche le formule: φ (fi), ψ (psi), σ (sigma), ...

Osservazione 4. Consideriamo le formule $\varphi(x_1, x_2) = (x_1 \wedge \overline{x}_2)$ e $\psi(x_1, x_2) = (x_1 \wedge x_2)$. Chiaramente φ vale 1 se e solo se alle variabili (x_1, x_2) si assegnano i valori $(1, 0)$, cioè $x_1 = 1, x_2 = 0$, mentre ψ vale 1 se e solo se assegnamo $(x_1, x_2) = (1, 1)$. Consideriamo la formula $\varphi \vee \psi$; essa vale 1 se e solo se φ vale 1 oppure ψ vale 1. In altre parole $\varphi \vee \psi$ vale 1 se applichiamo gli assegnamenti $(1, 0)$ e $(1, 1)$ alle variabili (x_1, x_2), altrimenti vale 0.

Ne risulta quindi che la formula $\varphi \vee \psi = (x_1 \wedge \overline{x}_2) \vee (x_1 \wedge x_2)$ specifica la funzione descritta dalla Tabella 3. Osserviamo che abbiamo scritto le

Tabella 3: Tabella di verità di $f(x_1, x_2) = \varphi(x_1, x_2) \vee \psi(x_1, x_2) = (x_1 \wedge \overline{x}_2) \vee (x_1 \wedge x_2)$

x_2	x_3	$f(x_1, x_2)$
0	0	0
0	1	0
1	0	1
1	1	1

formule φ, ψ in corrispondenza delle righe della tabella in cui il risultato è 1, ed inoltre le formule φ, ψ sono monomi.

E' facile allora, estendendo il ragionamento, ottenere il seguente algoritmo di sintesi. Data una tabella di verità con n variabili $x_1, ..., x_n$:

1. identificare le righe dove il risultato (ultima colonna) è 1;
2. per ognuna di queste righe, considerare il corrispondente assegnamento alle variabili (prime n colonne), e costruire il monomio

$$\varphi(x_1, ..., x_n) = \ell_1 \wedge ... \wedge \ell_n$$

   dove il letterale $\ell_i = x_i$ se nella riga c'è l'assegnamento $x_i = 1$, altrimenti $\ell_i = \overline{x}_i$, per ogni $i = 1, ..., n$. Rimarchiamo che per ogni riga con risultato 1 si costruisce un monomio φ diverso;
3. la formula σ che specifica la funzione descritta dalla tabella di verità sarà allora la disgiunzione di tutti i siffatti monomi φ.

Infatti, in una generica riga con risultato 1 si consideri il corrispondente assegnamento. Come era già evidente nell'Oss. 4, tale assegnamento rende vero (valore 1) un solo monomio che appare nella formula σ. Quindi, con tale assegnamento, σ varrebbe 1 grazie a questo monomio (poiché è disgiunzione di monomi).

Esempio 3. Consideriamo la Tabella 2. Le righe con risultato 1 sono le righe $2, 3, 5$ e 8, e i corrispodenti assegnamenti di valori a (x_1, x_2, x_3) sono

$(0,0,1),(0,1,0),(1,0,0),(1,1,1)$. Costruiamo i rispettivi monomi secondo il passo 2 dell'algoritmo:

$$\begin{aligned}
\varphi_2(x_1,x_2,x_3) &= \overline{x}_1 \wedge \overline{x}_2 \wedge x_3 \\
\varphi_3(x_1,x_2,x_3) &= \overline{x}_1 \wedge x_2 \wedge \overline{x}_3 \\
\varphi_5(x_1,x_2,x_3) &= x_1 \wedge \overline{x}_2 \wedge \overline{x}_3 \\
\varphi_8(x_1,x_2,x_3) &= \overline{x}_1 \wedge \overline{x}_2 \wedge \overline{x}_3
\end{aligned}$$

Una formula σ che specifica la funzione f della Tabella 2 è allora

$$\sigma = \varphi_2 \vee \varphi_3 \vee \varphi_5 \vee \varphi_8$$

ovvero

$$f(x_1,x_2,x_3) = (\overline{x}_1 \wedge \overline{x}_2 \wedge x_3) \vee (\overline{x}_1 \wedge x_2 \wedge \overline{x}_3) \vee (x_1 \wedge \overline{x}_2 \wedge \overline{x}_3) \vee (\overline{x}_1 \wedge \overline{x}_2 \wedge \overline{x}_3)$$

Notiamo che, come preannunciato, la formula ottenuta per f dall'algoritmo di sintesi è in DNF. Non è certo una formula breve! Riuscireste ad ottenere una formula più breve?

Osservazione 5. La specificazione originale $f(x_1,x_2,x_3) = x_1 \oplus x_2 \oplus x_3$ è diversa dalla formula ottenuta, ma ne è comunque equivalente (e quindi possiamo legare con $=$ le due formule). Il fatto che abbiamo ottenuto una formula booleana diversa ma equivalente, non deve sorprendere dal momento che come già noto al lettore due espressioni matematiche differenti (nella scrittura) possono essere equivalenti.

Esercizio 3. Costruire la tabella di verità per la funzione $f(x_1,x_2,x_3) = x_1 \wedge (\overline{x_2 \vee \overline{x}_3})$. Dalla tabella di verità ottenere una formula in DNF per f applicando l'algoritmo di sintesi.