4 Reti combinatorie

Contenuto

Il Capitolo 2 ha illustrato il ruolo dell'algebra di commutazione nella realizzazione di reti logiche a partire dalla specifica di una determinata funzionalità. L'operazione è sufficientemente immediata quando si tratta di trovare una funzione (tra le infinite) che implementi il comportamento desiderato; il compito si fa più arduo se ci si pone l'obiettivo non solo di realizzare le funzioni desiderate, ma anche di farlo in modo efficiente — in particolare minimizzando i costi della rete logica risultante. Questo capitolo presenta due metodi per realizzare reti logiche efficienti su due livelli, utilizzando sia un approccio esatto, sia un approccio euristico adatto alla progettazione mediante opportuni strumenti software. Per completare il discorso della sintesi di reti logiche combinatorie viene inoltre presentato un metodo euristico per la realizzazione di reti su più livelli, il cui obiettivo è ridurre ulteriormente il costo complessivo del circuito.

4.1 Introduzione

Progettare una rete combinatoria comporta una sequenza di passi successivi: generalmente si parte da una descrizione del problema fornita in linguaggio naturale quale l'italiano o l'inglese e si vuole arrivare a un circuito composto dagli elementi digitali di base, ovvero le porte logiche. Si è visto che esiste una corrispondenza diretta tra la rappresentazione circuitale e la rappresentazione mediante una o più espressioni booleane. Per questo si può quindi affermare che l'obiettivo della progettazione è quello di elaborare una specifica *semi-formale* come quella in linguaggio naturale fino a trasformarla in una descrizione *formale* costituita da una o più espressioni booleane. Questo processo si articola in alcuni passi:

Comprensione ed analisi della specifica semi-formale. Spesso la descrizione iniziale di un problema non è completa ma richiede una parziale riscrittura mirata ad eliminare ogni ambiguità. Casi tipici riguardano la scelta di una particolare codifica per le informazioni che la rete deve elaborare, la decisione di come gestire condizioni limite o di errore, l'analisi dell'ambiente in cui la rete è destinata ad operare, ecc. Terminata questa fase, si ottiene ancora una specifica semi-formale ma, a questo punto, *completa*.

Formalizzazione della specifica. È un passaggio essenziale ed ha lo scopo di tradurre la specifica in un linguaggio più adatto alla successiva manipolazione. La *funzionalità* della rete è già univocamente definita, quindi si tratta semplicemente di cambiare formalismo. Nel caso delle reti combinatorie si utilizzano le *tabelle della verità*: esse esprimono la relazione che esiste tra gli ingressi e le uscite della rete in esame. Dal punto di vista algebrico, una tabella della verità definisce completamente una funzione di commutazione.

Sintesi. Consiste nel passare dalle tabelle della verità ad *una* espressione algebrica per la funzione che esse rappresentano. Come già detto, una funzione può essere descritta da un numero infinito di espressioni, tutte equivalenti. Tali espressioni, tuttavia, possono differire rispetto a un dato criterio di valutazione, quale, per esempio, il numero di letterali. La sintesi ha quindi il solo scopo di arrivare ad una formulazione algebrica del problema iniziale, trascurando qualsiasi considerazione relativa alla qualità.

Ottimizzazione. Fissato un criterio di qualità, le tecniche di ottimizzazione hanno lo scopo di manipolare l'espressione di partenza risultante dalla sintesi in modo da ottenerne una che sia funzionalmente equivalente ma con caratteristiche migliori. Alcuni metodi presentati in questo capitolo uniscono la fase di sintesi e quella di ottimizzazione.

Il primo dei passi appena descritti è quello che richiede una maggiore attenzione e che permette al progettista di sfruttare appieno la propria esperienza al fine di produrre una buona specifica semi-formale. Il passaggio da tale specifica ad un insieme di tabelle della verità è una operazione semplice ma difficilmente automatizzabile mediante strumenti software in quanto richiede la capacità di comprendere il linguaggio naturale. Le fasi di sintesi e di ottimizzazione, invece, possono essere descritte completamente in forma algoritmica e pertanto possono essere svolte grazie ad opportuni software.

4.2 Formalizzazione della specifica

Poiché si tratta di una attività prettamente umana, il processo di formalizzazione di una specifica non è riconducibile ad un insieme o ad una sequenza di regole e di elaborazioni univoca e ben determinata. È utile considerare un primo esempio:

> *Si realizzi una rete combinatoria in grado di discriminare i numeri pari dai numeri dispari.*

Una prima considerazione riguarda il termine *numeri*: dalla specifica si deduce che la rete in questione tratta soltanto numeri *interi*, informazione presente nella specifica in forma implicita e che deve pertanto essere esplicitata. Continuando l'analisi, emerge che nulla si può dire a proposito dell'intervallo di variazione dei numeri che la rete deve essere in grado di trattare: questo aspetto comporta una scelta da parte del progettista. Spesso scelte di questo tipo sono guidate da considerazioni relative all'ambito applicativo o alla natura dell'informazione stessa. Se i numeri in questione indicano l'anno di una data un intervallo ragionevole potrebbe essere $[0; 2100]$ mentre se indicano i giorni della settimana l'intervallo dovrebbe essere $[0; 6]$. Per fissare le idee e per semplicità, si consideri il secondo caso. Rimane a questo punto da decidere quale tipo di rappresentazione adottare per i numeri. In questo tipo di scelta il progettista ha generalmente un maggiore margine di libertà. Si supponga, a titolo di esempio, di adottare la codifica binaria naturale. Dato che l'intervallo scelto copre 7 valori, sono necessari $\lceil \log_2 7 \rceil = 3$ bit. A questo punto la specifica non presenta più alcuna indeterminazione per quanto concerne gli ingressi. Una prima riformulazione è quindi la seguente:

> *Si realizzi una rete combinatoria dotata di un ingresso a tre bit* $X = \{x_0, x_1, x_2\}$ *che rappresenta un numero intero nell'intervallo* $[0; 6]$ *in codifica binaria naturale. La rete deve essere in grado di discriminare i numeri pari dai numeri dispari.*

In modo analogo si procede alla formalizzazione della parte di specifica che riguarda le uscite. Il risultato che la rete deve produrre è binario (pari o dispari) ed un bit è quindi sufficiente. Fissando arbitrariamente una codifica, per esempio 0=pari e 1=dispari, si definisce completamente l'uscita. Rimangono a questo punto da analizzare gli eventuali casi limite o le potenziali condizioni di errore. Nella specifica in esame, considerando che gli ingressi rappresentano i giorni della settimana, e fissando arbitrariamente che l'intervallo di variazione è $[0; 6]$, gli ingressi possono essere rappresentati su 3 bit: il valore 7, benchè corretto secondo la codifica scelta, non è valido. Per trattare questo caso particolare il progettista può seguire diverse strade. Una possibilità è quella di introdurre una nuova uscita destinata a segnalare una condizione di errore e scegliere una codifica anche per questo nuovo segnale, per esempio 0=corretto 1=errore. La nuova specifica completa è quindi la seguente:

> *Realizzare una rete combinatoria dotata di un ingresso a tre bit* $X = \{x_0, x_1, x_2\}$ *che rappresenta un numero intero nell'intervallo* $[0; 6]$ *in codifica binaria naturale. La rete è dotata di due uscite* z *ed* e *di un bit. L'uscita* z *indica se il valore in ingresso è pari (*$z = 0$*) o dispari (*$z = 1$*), mentre l'uscita* e *indica un errore (*$e = 1$*) oppure il funzionamento corretto (*$e = 0$*).*

Il passo successivo consiste nella definizione della specifica formale mediante tabelle della verità. La specifica indica che $z = 1$ quando X vale 000, 010, 100 oppure 110, mentre $z = 0$ negli altri casi e che $e = 1$ quando $X = 111$ mentre $e = 0$ in tutti gli altri casi. Si possono dunque scrivere le tabelle della verità:

x_0	x_1	x_2	z
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	0

x_0	x_1	x_2	e
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Questo conclude il processo di formalizzazione della specifica. Le tabella della verità infatti definiscono in modo univoco le due funzioni $z = z(x_0, x_1, x_2)$ ed $e = e(x_0, x_1, x_2)$.
Si noti, come considerazione conclusiva, che il valore della funzione z in corrispondenza dell'ingresso 111 è in realtà una condizione di indifferenza in quanto non corrisponde ad ingressi validi.

4.3 Sintesi

Il processo di sintesi, come accennato in precedenza, è il passaggio da una descrizione di una funzione ad una sua rappresentazione in forma algebrica. Come noto, una stessa funzione è descritta da un numero infinito di espressioni equivalenti ma potenzialmente differenti rispetto ad una data metrica. Nel seguito si discuterà di come ottenere l'espressione di una funzione in una delle due forme standard note rispettivamente come prima e seconda forma canonica.

4.3.1 Prima forma canonica

Si consideri una semplice funzione descritta dalla seguente tabella della verità:

x	y	f
0	0	1
0	1	0
1	0	1
1	1	0

Il procedimento per arrivare all'espressione di una funzione in prima forma canonica (Paragrafo 2.4.2) passa attraverso il teorema di Shannon (Paragrafo 2.4.1). Per la funzione di due variabili $f(x,y)$, l'espansione assume la forma:

$$f(x,y) = f(0,0)\cdot x'y' + f(0,1)\cdot x'y + f(1,0)\cdot xy' + f(1,1)\cdot xy \tag{4.1}$$

Dalla tabella della verità si deducono le seguenti uguaglianze:

$$f(0,0) = 1; \quad f(0,1) = 0; \quad f(1,0) = 1; \quad f(1,1) = 0 \tag{4.2}$$

che costituiscono i discriminanti di $f(x,y)$. Sostituendo tali valori nell'espressione (4.1), si ottiene:

$$f(x,y) = 1\cdot x'y' + 0\cdot x'y + 1\cdot xy' + 0\cdot xy \tag{4.3}$$

e cioè, semplificando:

$$f(x,y) = x'y' + xy'. \tag{4.4}$$

Un approccio differente consiste nel vedere la tabella della verità in esame come somma logica di due funzioni fittizie f_0 ed f_1 cioè:

x	y	f_0
0	0	1
0	1	0
1	0	0
1	1	0

$+$

x	y	f_1
0	0	0
0	1	0
1	0	1
1	1	0

$=$

x	y	f
0	0	1
0	1	0
1	0	1
1	1	0

Da questa visione si deduce che $f(x,y) = f_0(x,y) + f_1(x,y)$. A questo punto f_0 e f_1 sono funzioni semplici (presentano un solo 1) ed è pertanto possibile ricavare la loro espressione algebrica in modo diretto, senza ricorrere esplicitamente al teorema di espansione. Per esempio, considerando la funzione $f_0(x, y)$, si nota che essa vale 1 solo quando $x = 0$ ed $y = 0$ ovvero quando $x' = 1$ e $y' = 1$ cioè quando $x'y' = 1$. Dato che $x'y'$ vale 0 in tutti gli altri casi, si può scrivere:

$$f_0(x, y) = x'y'$$

Procedendo in modo analogo per $f_1(x, y)$ si ottiene:

$$f_1(x, y) = xy'$$

e conseguentemente:

$$f(x, y) = f_0(x, y) + f_1(x, y) = x'y' + xy'$$

Dal punto di vista operativo, quindi, si può esprimere una funzione come somma dei mintermini per cui essa assume valore 1.

4.3.2 Seconda forma canonica

Il metodo appena descritto per la sintesi delle funzioni in prima forma canonica ammette, come ogni enunciato dell'algebra di commutazione, uno schema duale. Per descrivere il nuovo procedimento si consideri la funzione seguente:

x	y	f
0	0	1
0	1	0
1	0	1
1	1	1

Si nota che la $f(x, y)$ assume valore zero solo quando $x = 0$ e $y = 1$ ovvero quando $x = 0$ e $y' = 0$ cioè quando $x + y' = 0$. Dato che $x + y'$ vale 1 in tutti gli altri casi, si conclude che:

$$f = f(x, y) = x + y'$$

La funzione è quindi espressa come somma di tutte le variabili di ingresso corrispondenti alle combinazioni per cui la funzione vale 0: si tratta perciò di un maxtermine. In particolare, il maxtermine è composto dalle variabili che assumono valore 0 prese in forma naturale e da quelle che assumono valore 1 prese in forma complementata.
Anche secondo questo approccio è possibile vedere una funzione generica come combinazione di funzioni più semplici. Si consideri ancora la funzione $f(x, y) = x'y' + xy'$ e si scomponga la corrispondente tabella della verità come segue:

x	y	f_0
0	0	1
0	1	0
1	0	1
1	1	1

$\cdot$

x	y	f_1
0	0	1
0	1	1
1	0	1
1	1	0

$=$

x	y	f
0	0	1
0	1	0
1	0	1
1	1	0

La funzione $f(x, y)$ è espressa, in questo caso, come prodotto delle due funzioni $f_0(x, y) = x + y'$ e $f_1(x, y) = x' + y'$ cioè:

$$f(x, y) = f_0(x, y) \cdot f_1(x, y) = (x + y')(x' + y')$$

Il risultato ottenuto è una espressione in seconda forma canonica (Paragrafo 2.4.3) cui si può pervenire, anche in questo caso, ricorrendo al teorema di Shannon.

4.3.3 Funzioni non completamente specificate

Come già anticipato nel Paragrafo 2.3.4, in molte situazioni reali accade che una funzione logica $f(x_0, \dots, x_{n-1})$ non sia definita per tutte le 2^n possibili configurazioni degli ingressi. Ciò può dipendere da due differenti ragioni:

- Non tutte le 2^n configurazioni di ingresso sono ammesse. Nella codifica BCD (Paragrafo 3.6.1), per esempio, si utilizzano 4 bit per rappresentare le cifre da 0 (0000) a 9 (1001). Le combinazioni di ingresso 1010, 1011, 1100, 1101, 1110 e 1111 non sono pertanto ammesse.

- Il valore della funzione può essere ignorato in quanto non porta alcuna informazione oppure non influenza il comportamento delle reti a valle.

Quando il valore della funzione non è specificato si dice che la funzione presenta una *condizione di indifferenza* o *don't care* e si indica con il simbolo $\times$ o $-$. È importante notare che tali simboli non sono nuovi elementi della base B dell'algebra di commutazione. Si consideri a tale proposito la funzione non completamente specificata descritta dalla seguente tabella della verità:

x	y	f
0	0	1
0	1	0
1	0	–
1	1	0

Il simbolo di indifferenza $-$ indica che quando $x = 1$ e $y = 0$ la funzione f è libera di assumere un valore qualsiasi. Di fatto, quindi, la tabella della verità appena vista corrisponde a non una ma *due* funzioni: una in cui $f(1,0) = 0$ ed una in cui $f(1,0) = 1$ ovvero alle due tabelle:

x	y	f_0
0	0	1
0	1	0
1	*0*	*0*
1	1	0

x	y	f_1
0	0	1
0	1	0
1	*0*	*1*
1	1	0

Procedendo alla sintesi in prima forma canonica si ottengono le due funzioni:

$$f_0(x,y) = x'y' \tag{4.5}$$

$$f_1(x,y) = x'y' + xy' \tag{4.6}$$

Si pone ora il problema di come rappresentare algebricamente le condizioni di indifferenza e in particolare come descrivere le funzioni non completamente specificate in prima o in seconda forma canonica. Per fissare le idee, si faccia dapprima riferimento alla prima forma canonica. Una condizione di indifferenza, proprio in quanto può assumere un valore qualsiasi, può essere vista come un *parametro* della funzione. Indicando con p tale parametro si ha:

$$f(x,y) = f(0,0) \cdot x'y' + f(0,1) \cdot x'y + f(1,0) \cdot xy' + f(1,1) \cdot xy$$

si può esprimere la funzione in esame sostituendo il valore dei discriminanti $f(1,1) = 0$, $f(0,1) = 0$, $f(0,0) = 1$ e $f(1,0) = p$, ottenendo:

$$f(x,y) = 1 \cdot x'y' + p \cdot xy' = x'y' + pxy'$$

Si verifica facilmente che questa espressione è equivalente alle due funzioni (4.5) e (4.6) che infatti possono essere ottenute sostituendo a p i valori 1 e 0:

$$p = 0 \quad \rightarrow \quad f(x,y) = x'y' = f_0(x,y)$$
$$p = 1 \quad \rightarrow \quad f(x,y) = x'y' + xy' = f_1(x,y)$$

Per completezza è bene riportare anche il procedimento per la sintesi in seconda forma canonica. Dal teorema di Shannon in seconda forma canonica e dai discriminanti $f(0,0) = 1$, $f(0,1) = 0$, $f(1,0) = p$ e $f(1,1) = 0$ si ottiene:

$$f(x,y) = (x + y')(p + x' + y)(x' + y')$$

In generale una funzione non completamente specificata può presentare $k \geq 1$ condizioni di indifferenza: in questo caso i metodi visti continuano a valere ma è necessario introdurre un numero k di parametri $p_0, \ldots, p_{k-1}$. Ad ogni condizione di indifferenza corrisponde dunque un parametro aggiuntivo. Il numero di funzioni descritte da una specifica contenente k condizioni di indifferenza è 2^k poichè tanti sono i possibili assegnamenti di $\{0, 1\}$ a k parametri. Si noti che come caso particolare una funzione priva di condizioni di indifferenza non richiede parametri quindi $k = 0$ e $2^k = 1$: questo significa appunto che la funzione è unica o *completamente specificata*.

4.4 Minimizzazione esatta

I metodi di sintesi appena presentati permettono di ottenere in maniera sistematica espressioni strutturate secondo due forme ben precise, risolvendo così il problema del passaggio dalla specifica di una funzione ad una sua espressione. Questa sezione affronta il problema della *ottimizzazione* delle funzioni, ovvero della loro trasformazione, attraverso passi successivi, con lo scopo di ottenere nuove espressioni equivalenti ma *migliori* rispetto ad un dato *criterio* di valutazione.
Questo aspetto può essere precisato ulteriormente e formalizzato mediante il concetto di *metrica*. Una metrica è una *grandezza* che esprime in modo indiretto una caratteristica di un dato sistema. Nel caso in esame, il sistema è una rete combinatoria digitale costruita utilizzando pochi elementi di base: le *porte logiche* (Paragrafo 2.6). Le caratteristiche che si desidera valutare per stabilire la qualità di una tale rete sono generalmente l'*area* della realizzazione fisica della rete, il *tempo* necessario a produrre un dato risultato, la *potenza* o l'*energia* assorbita, ecc. Si parla pertanto di *metriche d'area*, *metriche di tempo* e così via.
Questa sezione presenta le tecniche di ottimizzazione dell'*area* di una rete e si farà pertanto riferimento ad alcune *metriche d'area* per valutare la qualità delle diverse soluzioni. Alcune metriche d'area comunemente utilizzate sono:

- numero di porte logiche generiche;
- numero di porte logiche a due ingressi;
- numero di implicanti o di implicati;
- numero di letterali.

I paragrafi seguenti presentano alcuni metodi, esatti ed euristici per la minimizzazione delle funzioni combinatorie. I metodi esatti considerati sono quello delle *mappe di Karnaugh* e quello di *Quine-McCluskey*, mentre tra i vari metodi euristici esistenti si farà riferimento alle a uno strumento automatico di minimizzazione molto conosciuto: *MIS*/*SIS* [MIS, 1987] [SIS, 1992].

4.4.1 Metodo delle mappe di Karnaugh

Il metodo delle mappe di Karnaugh permette di trasformare il problema della minimizzazione di una espressione dal piano algebrico al piano geometrico e topologico. Questa sezione introduce i concetti che stanno alla base di tale metodo e descrive operativamente la procedura da adottare per la soluzione di problemi di minimizzazione. Il metodo delle mappe consiste di due fasi ben distinte, indicate generalmente come *espansione* e *copertura*.

Espansione

La fase di espansione ha lo scopo di trasformare una espressione algebrica in modo da costruire termini prodotto costituiti dal minor numero possibile di letterali. Ciò può essere fatto mediante diverse leggi dell'algebra, applicate secondo opportune sequenze, sfortunatamente dipendenti dalla forma dell'espressione stessa. Non potendo ricorrere a regole fisse per l'uso ottimale delle regole dell'algebra, si limita l'attenzione alla seguente proprietà:

$$xP + x'P = (x + x')P = 1 \cdot P = P$$

in cui P indica un termine prodotto, costituito da k letterali. È semplice constatare che l'espressione iniziale è costituita da $2(k + 1)$ letterali mentre l'espressione finale solo da k letterali. Si nota inoltre che tale legge è applicabile ogni qualvolta si hanno due termini prodotto costituiti dalle stesse variabili e tali per cui una ed una sola variabile appare in un prodotto in forma naturale e nell'altro in forma negata (x, nell'esempio). Questa regola può chiaramente essere applicata ripetutamente. Si consideri un primo esempio:

$$\begin{aligned} f(x, y, z, w) &= xyzw' + xy'zw' + xy'zw + xyzw \\ &= xzw'(y + y') + xzw(y + y') \\ &= xzw' + xzw \\ &= xz(w' + w) = xz \end{aligned} \tag{4.7}$$

Si dice che dapprima si è effettuata una espansione rispetto alla variabile y e poi una espansione rispetto alla variabile z e si può esprimere l'entità della semplificazione dicendo che si è passati da 16 letterali nell'espressione iniziale a 2 letterali in quella finale. Dal punto di vista formale, il risultato di questo tipo di semplificazione prende il nome di *implicante*.

Definizione 4.1 *Sia $f(x_{n-1}, x_{n-2}, \ldots, x_0)$ una funzione di n variabili booleane. Un* implicante *per f è un termine prodotto $y_{k-1}y_{k-2}\cdots y_0$, in cui $y_i = x_j$ oppure $y_i = x'_j$, e per cui risulta $f(y_{k-1}\cdots y_0) = 1$. Quando $k = n$ tale prodotto coinvolge tutte le variabili ed è pertanto anche un mintermine.* □

Un implicante di una funzione è quindi un termine prodotto che coinvolge solo *alcune* variabili di f e tale per cui la funzione vale 1. In modo duale si introduce la definizione di *implicato*.

Definizione 4.2 *Sia $f(x_{n-1}, x_{n-2}, \ldots, x_0)$ una funzione di n variabili booleane. Un* implicato *per f è un termine somma $y_{k-1} + y_{k-2} + \cdots + y_0$, in cui $y_i = x_j$ oppure $y_i = x'_j$, e per cui risulta $f(y_{k-1} + \cdots + y_0) = 0$. Quando $k = n$ tale prodotto coinvolge tutte le variabili ed è pertanto anche un maxtermine.* □

Nell'esempio di espansione dell'equazione (4.7) non vi sono stati problemi e in particolare la scelta dell'ordine delle variabili rispetto cui espandere non avrebbe comportato alcuna differenza. Ma non è sempre così. Per chiarire questo aspetto, si consideri un nuovo esempio, descritto dall'espressione seguente:

$$f(x, y, z) = xyz + x'yz + xy'z \tag{4.8}$$

Applicando la proprietà in esame ai primi due termini si ottiene:

$$f(x, y, z) = (x + x')yz + xy'z = yz + xy'z$$

mentre applicandola al primo e al terzo si otterrebbe:

$$f(x, y, z) = x'yz + (y + y')xz = x'yz + xz$$

Per prima cosa si nota che le espressioni, ovviamente equivalenti, sono diverse nella forma; in secondo luogo si può constatare che a nessuna delle due espressioni ottenute è possibile applicare ulteriormente la stessa proprietà. In entrambi i casi si è passati da 9 a 5 letterali. Per migliorare ulteriormente la semplificazione si può fare ricorso ad una proprietà aggiuntiva: l'idempotenza (Paragrafo 2.1.1), utilizzata nella forma seguente:

$$P = P + P$$

in cui, ancora, P indica un generico termine prodotto. Si consideri nuovamente l'espressione (4.8) e si duplichi il primo termine per la proprietà d'idempotenza:

$$f(x, y, z) = xyz + x'yz + xy'z + xyz$$

Procedendo ora all'espansione secondo lo schema già visto si ottiene:

$$\begin{aligned} f(x, y, z) &= xyz + x'yz + xy'z + xyz \\ &= (x + x')yz + (y' + y)xz = yz + xz \end{aligned}$$

La nuova espressione ha solo 4 letterali, rispetto ai 5 delle due espressioni ottenute senza l'applicazione dell'idempotenza.

Dagli esempi riportati emerge implicitamente che l'espansione deve essere ripetuta fino a che è possibile. In termini più precisi e formali si dice che lo scopo dell'espansione è quello di individuare tutti e soli gli implicanti *primi*, cioè gli implicanti con il numero di letterali minore possibile.

Definizione 4.3 *Si dice primo un implicante o un implicato quando non è più suscettibile di espansione.* □

Distanza di Hamming

Le considerazioni di carattere algebrico svolte fino a questo punto sono il fondamento teorico del metodo delle mappe di Karnaugh. Dal punto di vista operativo, tuttavia, rimangono due problemi:

- quali coppie di termini scegliere per l'espansione;
- quali termini scegliere per la duplicazione.

Per quanto concerne il primo problema, si è visto che due termini prodotto sono semplificabili quando sono costituiti dalle stesse variabili e quando una sola di esse appare in un prodotto naturale e nell'altro negata. Queste condizioni possono essere espresse molto semplicemente grazie al concetto di *distanza di Hamming*. Siano P_0 e P_1 due termini prodotto costituiti dalle k variabili $x_0, \ldots, x_{k-1}$, scritte secondo questo ordine. Si scrivano ora i prodotti sottintendendo le variabili (i loro nomi) e riportando ordinatamente un 1 quando una variabile appare in forma naturale e uno 0 quando appare in forma negata. Secondo questo schema i prodotti $P_0 = xyz'w$ e $P_1 = xyz'w'$ sono esprimibili come $P_0 = 1101$ e $P_1 = 1100$, avendo fissato e sottinteso l'ordine delle variabili $\{x, y, z, w\}$. È semplice osservare che due termini prodotto così espressi sono semplificabili se e solo se differiscono in una ed una sola cifra binaria. Si nota inoltre che la variabile corrispondente alla cifra che differisce è quella eliminata nella semplificazione. Questo schema è riassunto in Figura 4.1.

$$\begin{array}{c} xyz'w \\ xyz'w' \\ \hline xyz' \end{array} \qquad\qquad \begin{array}{c} 1101 \\ 1100 \\ \hline 110- \end{array}$$

Figura 4.1 Schema di semplificazione dei termini prodotto

Il concetto di distanza di Hamming si applica appunto a coppie di sequenze di cifre binarie di uguale lunghezza e indica il numero di cifre diverse tra le due sequenze. Più in generale vale la seguente definizione.

Definizione 4.4 *Siano* $A = \{a_0, \ldots, a_{k-1}\}$ *e* $B = \{b_0, \ldots, b_{k-1}\}$ *due sequenze di* k *cifre binarie ciascuna; si dice* distanza di Hamming *tra* A *e* B *la grandezza indicata con* $d_H(A, B)$ *e definita dalla seguente relazione:*

$$d_H(A, B) = \sum_{i=0}^{k} a_i \oplus b_i$$

in cui il risultato dell'operazione di XOR è da intendersi come valore numerico e non logico. □

Per esempio, le due stringhe $A = 0010110$ e $B = 0110100$ hanno distanza di Hamming $d_H(A, B) = 2$, infatti:

a_i	0010110		
b_i	0110100		
$a_i \oplus b_i$	0100010	$\rightarrow$	$\sum_{i=0}^{6} a_i \oplus b_i$ = 2

È facile verificare alcune semplici proprietà della distanza di Hamming:

$$d_H(A, A) = 0$$
$$d_H(A, B) = d_H(B, A)$$
$$d_H(A, A') = k \qquad (k = |A|)$$

Grazie al concetto di distanza di Hamming ed alla rappresentazione dei prodotti mediante stringhe binarie si può infine riassumere la regola per la semplificazione dicendo che due termini prodotto sono semplificabili se e solo se la loro distanza di Hamming è uguale ad uno.

Costruzione delle mappe

Le mappe di Karnaugh, come descritto nel seguito, sono una *rappresentazione* di una funzione booleana studiata per mettere in evidenza le coppie di mintermini o di implicanti a distanza di Hamming unitaria. Tale rappresentazione ha come fondamento un'altra, meno intuitiva, rappresentazione di una funzione di commutazione: quella cartesiana in uno spazio con un opportuno numero di dimensioni. Si consideri dapprima la funzione $f(x) = x$ di una sola variabile. Ricorrendo alla rappresentazione cartesiana tradizionale, tale funzione può essere descritta in uno spazio bidimensionale, come mostra la Figura 4.2(a). Dato che le variabili dell'algebra di commutazione possono assumere solo i valori 0 ed 1, si può adottare per la stessa funzione una rappresentazione semplificata su una sola dimensione in cui il valore della funzione è indicato da un punto pieno se uguale ad 1 e da un punto vuoto se uguale a 0. Tale rappresentazione è mostrata in Figura 4.2(b).

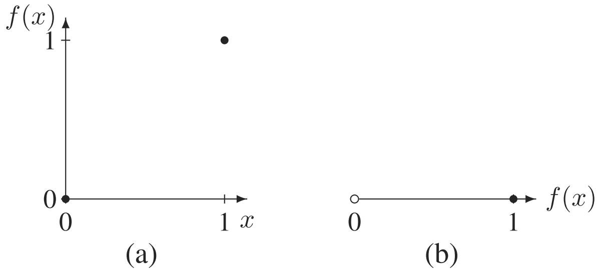

Figura 4.2 Rappresentazioni di $f(x)$: (a) tradizionale, (b) semplificata

Grazie a questo accorgimento quindi, una funzione di una sola variabile può essere rappresentata in uno spazio monodimensionale. È inoltre possibile generalizzare questo tipo di rappresentazione per funzioni di n variabili semplicemente ricorrendo ad uno spazio n-dimensionale. Inoltre, dato che sugli assi gli unici valori di possibili sono 0 ed 1, si usa ricorrere ad una rappresentazione ancora leggermente diversa, basata sui cosiddetti n-cubi, ovvero ipercubi n-dimensionali in cui ogni vertice è annotato con le sue coordinate, come mostrato in Figura 4.3.

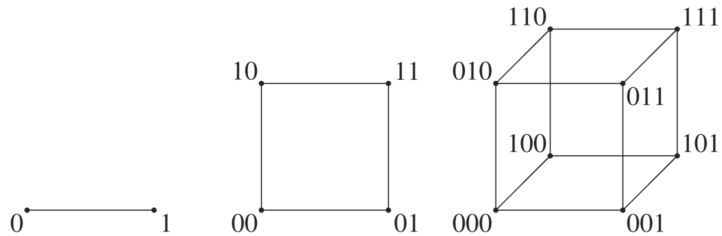

Figura 4.3 Rappresentazione di un 1-cubo, un 2-cubo e un 3-cubo

Si consideri ora la funzione in prima forma canonica $f(x, y, z) = xyz + x'yz + xyz' + x'yz'$, corrispondente ai mintermini con valore 111, 011, 110 e 010. Essa può essere rappresentata su un 3-cubo come mostra la Figura 4.4.

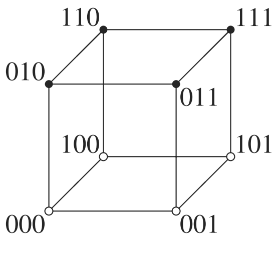

Figura 4.4 Rappresentazione di $f(x, y, z) = xyz + x'yz + xyz' + x'yz'$

Osservando tale figura si nota che i mintermini che possono essere semplificati mediante espansione, ovvero i mintermini a distanza di Hamming unitaria, sono rappresentati sul 3-cubo da punti pieni che risultano essere adiacenti, ossia connessi da un lato. Per esempio i mintermini $x'yz'$ (010) e $x'yz$ (011) giacciono in alto sulla faccia anteriore del 3-cubo e sono connessi da un lato. L'espansione porta al nuovo implicante $x'y$ che corrisponde ad un *lato* del 3-cubo. Tale lato può anche essere pensato come un sottocubo di dimensione 1. Allo stesso modo si possono semplificare i due mintermini xyz' (110) e xyz (111) dando origine all'implicante xy, ovvero ad un altro lato del 3-cubo. Infine, i due implicanti $x'y$ e xy possono ancora essere semplificati dando origine al nuovo implicante y. Sul 3-cubo tale implicante corrisponde alla *faccia* superiore, ovvero ad un sottocubo di dimensione 2.
La rappresentazione delle funzioni mediante n-cubi ha quindi l'interessante proprietà di evidenziare per costruzione i mintermini di una data funzione che risultano essere adiacenti. In particolare, tale rappresentazione non si limita ad evidenziare le *coppie* di mintermini adiacenti bensì, più in generale, *tutti* i raggruppamenti di mintermini semplificabili mediante espansioni successive.
Si consideri ora il caso generale di una funzione $f(x_0, \ldots, x_{n-1})$ di n variabili: essa può essere rappresentata su un n-cubo. Un insieme di mintermini $\{P_0, \ldots, P_{t-1}\}$ di tale funzione risulta semplificabile mediante espansioni successive se t è una potenza di 2, cioè se $t = 2^k$, e se i t mintermini formano un sottocubo di dimensione k. In tal caso si otterrà un implicante di $n-k$ letterali.
Da quanto discusso fino a questo punto risulta chiaro che la rappresentazione proposta risolve il problema della identificazione delle coppie candidate all'espansione. Rimane ora il problema della identificazione dei mintermini, o più in generale degli implicanti, da duplicare per poter iterare il processo di espansione.
Per chiarire questo punto si consideri ancora un esempio, cioè la funzione:

$$f(x, y, z) = xyz + xyz' + x'yz'$$

in cui si richiede la duplicazione del mintermine xyz' in modo da poter effettuare l'espansione con il primo e con il terzo mintermine. Come mostra la Figura 4.5, il vertice 110, cui corrisponde appunto il mintermine xyz', è condiviso dai due 1-cubi A e B. Questa circostanza indica che, a patto di accettare che un vertice di un n-cubo possa prendere parte a più raccoglimenti, la rappresentazione scelta risolve anche il problema della duplicazione.
Questo tipo di rappresentazione e tutti i concetti espressi in questo paragrafo stanno alla base del metodo delle mappe di Karnaugh. Una mappa di Karnaugh, infatti, altro non è che una trasposizione di un n-cubo su due dimensioni, in modo da essere più facilmente maneggiabile. È importante notare che tale trasposizione,

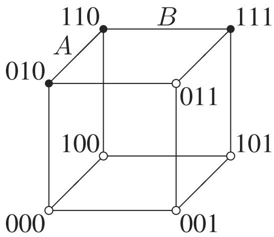

Figura 4.5 Rappresentazione di $f(x, y, z) = xyz + xyz' + x'y('z)$

o proiezione, deve mantenere le proprietà di adiacenza che caratterizzano i vertici degli n-cubi. Ciò è semplice per un 1-cubo e per un 2-cubo, come mostra la Figura 4.6. Il 3-cubo, invece è leggermente più complesso in quanto deve essere "srotolato" per poter essere portato su due dimensioni. Si noti inoltre che la casella in alto a sinistra (000) e la casella in alto a destra (010) corrispondono a due vertici originariamente adiacenti sul 3-cubo e devono pertanto essere considerate adiacenti anche sulla mappa. Lo stesso vale per le caselle in basso a destra ed in basso a sinistra.

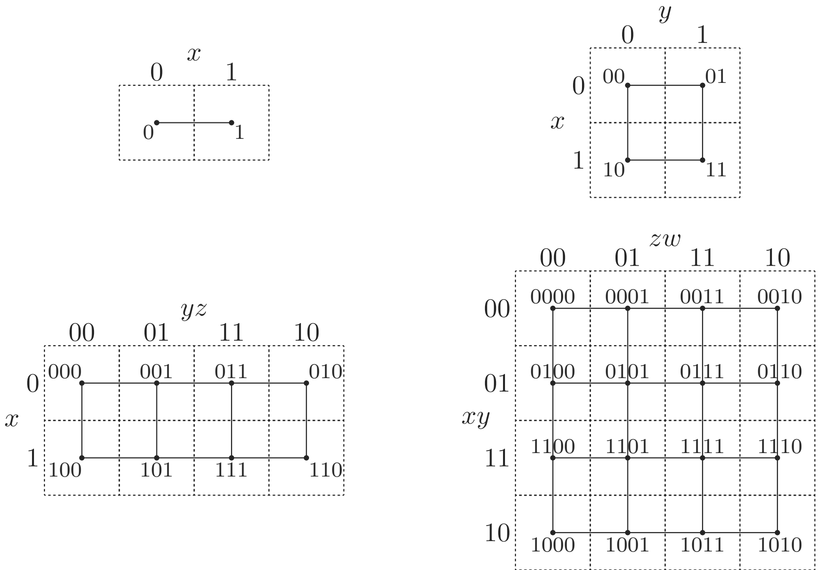

Figura 4.6 Costruzione delle mappe di Karnaugh

La costruzione appena presentata è estendibile con alcuni accorgimenti al caso di un 4-cubo. In questo caso le caselle giacenti su tutti i bordi della mappa sono adiacenti alle caselle corrispondenti, in orizzontale o in verticale, sul lato opposto.

Avendo rispettato le relazioni di adiacenza proprie degli n-cubi, le mappe di Karnaugh evidenziano i *mintermini semplificabili* come *celle adiacenti*. Per esempio, la funzione $f(x,y,z) = xyz + xyz' + x'yz'$ già considerata e mostrata su un 3-cubo in Figura 4.5, è rappresentata su una mappa di Karnaugh in Figura 4.7(a).

$x \backslash yz$	00	01	11	10
0	0	0	0	1
1	0	0	1	1

(a)

$x \backslash yz$	00	01	11	10
0	0	0	0	1
1	0	0	1	1

(b)

Figura 4.7 Mappa di Karnaugh della funzione $f(x,y,z) = xyz + xyz' + x'yz'$

Le relazioni di adiacenza tra i mintermini si traducono sulla mappa in relazioni di adiacenza tra celle e sono rappresentati dai raggruppamenti mostrati in Figura 4.7(b). Si consideri ora il raggruppamento verticale, costituito dai mintermini $x'yz'$ (010) e xyz' (110). L'espressione dell'implicante corrispondente si puó ricavare per ispezione della mappa, senza cioè ricorrere alle espressioni algebriche dei mintermini coinvolti. Infatti, la mappa mostra che le celle coinvolte nel raggruppamento sono contraddistinte da un valore costante delle variabili y e z mentre il valore della variabile x cambia. Si può indicare questo fatto scrivendo che $x = -$, $y = 1$ e $z = 0$ ovvero che l'implicante corrisponde alla stringa di bit -10 ovvero al termine prodotto yz'. In modo analogo, il raccoglimento orizzontale corrisponde alla stringa $11-$ (la variabile z cambia valore) ovvero al termine prodotto xy. Considerando entrambi i raccoglimenti si giunge quindi alla forma minima cercata della funzione.
Si consideri ora la funzione descritta dalla tabella della verità in Figura 4.8(a). La sua rappresentazione su una mappa di Karnaugh è mostrata in Figura 4.8(b).
I due raggruppamenti mostrati sulla mappa corrispondono alle stringhe $0--$ e -11 ovvero agli implicanti x' e yz rispettivamente. L'espressione minima per la funzione data è pertanto $f(x,y,z) = x' + yz$.
Come già visto, il valore di una funzione in corrispondenza di alcuni ingressi può non essere specificato: ciò fornisce un grado di libertà aggiuntivo nella fase di sintesi in quanto la condizione di indifferenza può essere considerata uno 0 o un 1 a seconda di cosa risulta più conveniente. Per capire come trattare le condizioni di indifferenza con il metodo delle mappe, si consideri l'esempio di Figura 4.9.
Ignorando la condizione di indifferenza (ovvero considerandola come uno 0) si otterrebbe la soluzione mostrata in Figura 4.10(a) cui corrisponde l'espressione

x	y	z	f
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

(a)

$x \backslash yz$	00	01	11	10
0	1	1	1	1
1	0	0	1	0

(b)

Figura 4.8 Tabella della verità e mappa di Karnaugh

$x \backslash yz$	00	01	11	10
0	0	1	–	0
1	0	0	1	0

Figura 4.9 Mappa di Karnaugh con condizioni di indifferenza

$x \backslash yz$	00	01	11	10
0	0	1	–	0
1	0	0	1	0

(a)

$x \backslash yz$	00	01	11	10
0	0	1	–	0
1	0	0	1	0

(b)

Figura 4.10 Espasione con condizioni di indifferenza

$f(x, y, x) = x'y'z + xyz$ avente costo pari a 6 letterali. Considerando invece la condizione di indifferenza come un 1 si possono espandere i mintermini come mostrato in Figura 4.10(b). A questa scelta corrisponde la funzione $f(x, y, x) = x'y' + yz$ con un costo di soli 4 letterali.

In generale una o più condizioni di indifferenza sono da considerarsi delle utili *risorse* che, introducendo alcuni gradi di libertà nella funzione, permettono potenzialmente di pervenire a n-cubi di dimensioni maggiori ovvero a termini prodotto con un numero minore di letterali.