1 Lezione del 10-10-24

1.1 Sintesi di reti in forma SP a costo minimo

Esistono due criteri di costo per le reti:

- **A porte:** ogni **porta** conta per un'unità di costo;
- **A diodi:** ogni **ingresso** conta per un'unità di costo.

Presentiamo un metodo, applicabile a reti con'un uscita, che produce reti in forma SP a 2 livelli di logica in quanto, per una legge combinatoria F, si ha::

$$\text{Sintesi di F a 2 L.L. in forma SP} \subset \text{Sintesi di F a 2 L.L.} \subset \text{Sintesi di F}$$

1.1.1 Espansione di Shannon

Si può dimostrare il seguente risultato:

Teorema 1.1: Espansione di Shannon

Si può sempre scrivere qualunque legge combinatoria f come somma di prodotti degli ingressi (diretti o negati).

Questo significa che, se ho una legge combinatoria $z = f(x_{N-1}, ..., X_0)$, posso dire:

$$\begin{aligned}
z = f(0, ..., 0, 0) \cdot \overline{x_{N-1}} \cdot \overline{x_{N-2}} \cdot ... \cdot \overline{X_1} \cdot \overline{X_0} + \\
f(0, ..., 0, 1) \cdot \overline{x_{N-1}} \cdot \overline{x_{N-2}} \cdot ... \cdot \overline{X_1} \cdot X_0 + \\
... + \\
f(1, ..., 1, 0) \cdot x_{N-1} \cdot x_{N-2} \cdot ... \cdot X_1 \cdot \overline{X_0} + \\
f(1, ..., 1, 1) \cdot x_{N-1} \cdot x_{N-2} \cdot ... \cdot X_1 \cdot X_0
\end{aligned}$$

che equivale a quanto avevamo visto con la sintesi di reti combinatorie a N ingressi con multiplexer a N variabili di comando.

A questo punto possiamo ottenere la cosiddetta **forma canonica SP**, applicando le proprietà:

$$\begin{cases} 1 \cdot \alpha = \alpha \\ 0 \cdot \alpha = 0 \\ 0 + \beta = \beta \end{cases}$$

all'espansione di Shannon (sostanzialmente rimuoviamo tutti i termini a cui corrispondono uscite negate). Della forma canonica SP possiamo dire che è:

- **SP:** è fatta da somme e prodotti;
- **Canonica:** ogni prodotto ha come fattori tutti gli ingressi, diretti o negati;
- Ciascuno dei termini della somma si chiama **mintermine**;
- Ogni mintermine corrisponde ad uno stato riconosciuto dalla rete.

L'insieme dei termini (mintermini) sommati fra di loro che otteniamo dall'espansione di Shannon prende il nome di **lista di mintermini**.

1.2 Semplificazione della forma canonica SP

Definiamo quindi un metodo per la semplificazione della lista dei mintermini. Divideremo quest'operazione in due passaggi principali:

- **Identificazione degli implicanti principali**: si ricava una lista di termini ricavati da quelli di partenza, e di dimensioni più piccole, che rappresentano la stessa legge combinatoria;
- **Eliminazione delle ridondanze**: si rimuovono gli implicanti che non portano informazioni utili alla legge combinatoria.

1.2.1 Metodo di Quine-McCluskey

Si presenta il metodo di Quine-McCluskey per l'identificazione degli implicanti principali. Questo metodo prevede di:

- **Fondere i mintermini** applicando **esaustivamente** la regola:

  $$\alpha x + \alpha \bar{x} = \alpha$$

  che possiamo dimostrare come:

  $$\alpha x + \alpha \bar{x} = \alpha(x + \bar{x}) = \alpha, \quad x + \bar{x} = 1$$

  alla lista dei mintermini.

  Ripetiamo questo passaggio $N - 1$ volte per la dimensione N dei termini, riducendo ogni volta la dimensione degli implicanti di 1. Si ricava una forma SP, detta **lista di implicanti**.

- **Rimuovere i duplicati** dalla lista dei duplicanti, applicando l'altra regola:

  $$\alpha x + \alpha = \alpha$$

  sugli implicanti che hanno elementi in comune.

Troviamo quindi quella che è detta **lista degli implicanti principali**. Questa lista contiene meno elementi della forma canonica SP, ma non è ancora di costo minimo: potrebbe contenere ridondanze, cioè implicanti non necessari alla corretta modelizzazione della legge combinatoria.

1.2.2 Liste di copertura non ridondanti

Una **lista di copertura** è una lista di implicanti, la cui somma è una forma SP per la funzione f. La **lista di copertura non ridondante** è la lista che smette di essere una lista di copertura appena si toglie un elemento.

Un punto importante è che la lista dei mintermini è una lista non ridondante, mentre la lista degli implicanti principali può esserlo. Si introduce quindi uno strumento per la visualizzazione degli implicanti e le loro ridondanze.

1.3 Mappe di Karnaugh

Per una rete a N ingressi la corrispondente **mappa di Karnaugh** è una matrice di 2^N celle, dove le coordinate rappresentano gli ingressi, e gli elementi della matrice le uscite. Sono diagrammi che tornano utili per rappresentare graficamente gli implicanti, ed eliminarne le ridonanze. Vediamo, ad esempio, mappe con $N = 2$, 3 e 4:

$X_1 \backslash X_0$	0	1
0	0	1
1	1	0

$X_2 \backslash X_1X_0$	00	01	11	10
0	0	0	1	-
1	1	-	0	0

$X_3X_2 \backslash X_1X_0$	00	01	11	10
00	0	0	0	0
01	0	0	0	0
11	0	0	0	0
10	0	0	0	0

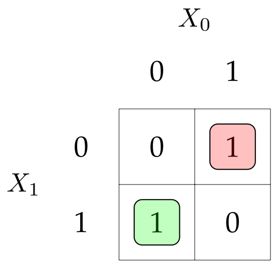
In una mappa di Karnaugh, celle **contigue** hanno coordinate **adiacenti**, e viceversa. Oltre le 4 coordinate, per le mappe non possiamo più rappresentare queste mappe senza la terza dimensione.

1.3.1 Ricerca dei sottocubi principali

Definiamo innanzitutto:

- **Sottocubo di ordine 1:** una casella che contiene un 1, corrispondente quindi ad uno stato di ingresso riconosciuto dalla rete, si indica come SO1;
- **Coordinate** di un SO1: stato di ingresso corrispondente al sottocubo;
- **Adiacenza** fra SO1: due SO1 sono adiacenti se differiscono fra loro di una sola coordinata.

Vediamo, ad esempio, una mappa di Karnaugh con $N = 2$, una serie di sottocubi di ordine 1 con la tabella associata:

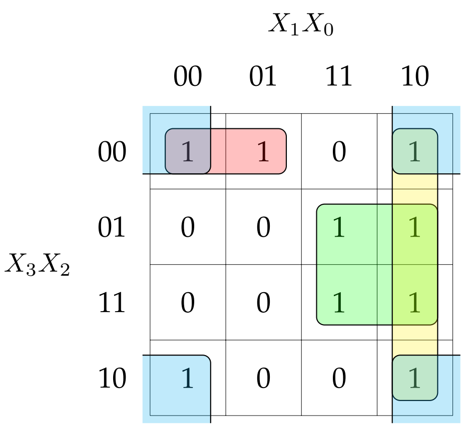


	X_1	X_0
A	0	1
B	1	0

Notiamo come A corrisponde all'implicante $\overline{X_1}X_0$, e B all'implicante $X_1\overline{X_0}$.
Possiamo continuare:

- **Sottocubo di ordine 2:** costituito da SO1 adiacenti, e si dice che **copre** i SO1 che lo formano. Si indica come SO2;
- **Sottocubo di ordine 4:** costituito da SO2 adiacenti, e si dice che **copre** i SO2 che lo formano. Si indica come SO4;
- **Sottocubo di ordine 8:** costituito da SO4 adiacenti, e si dice che **copre** i SO4 che lo formano. Si indica come SO8;

Vediamo un'ultimo esempio, con $N = 4$:



	X_3	X_2	X_1	X_0
A	0	0	0	-
B	-	1	1	-
C	-	-	1	0
D	-	0	-	0

Notiamo dall'esempio che le mappe di Karnaugh rispettano quello che potremmo chiamare *effetto pacman*: lo stesso implicante può esistere su lati opposti della mappa. Il bisogno di rappresentare le adiacenze dà origine a questa particolarità, come determina l'ordine particolare delle attivazioni degli ingressi. Inoltre, notiamo come i trattini nelle tabelle delle coordinate denotano che la variabile non influenza l'implicante, cioè rappresentano, in inglese, un *don't care*.

Si dice che un sottocubo è **principale** quando non esiste nessun sottocubo più grande che lo copre completamente. Si ha quindi che sottocubi e implicanti sono correlati: un sottocubo principale di ordine p rappresenta un implicante principale di $N - \log_2(p)$ variabili.

Si presenta quindi l'algoritmo per la ricerca dei sottocubi principali:

Algoritmo 1 per la ricerca dei sottocubi principali

Input: una mappa
Output: i sottocubi principali della mappa
ciclo:
Considera tutti i sottocubi di ordine p non interamente contenuti in sottocubi di ordine più grande, e segnali tutti: questi sono sicuramente principali
if l'insieme trovato finora basta a coprire tutta a mappa **then**
  L'algoritmo è terminato
else
  Poni $p \leftarrow \frac{p}{2}$ e vai a ciclo
end if

1.3.2 Ricerca delle liste di copertura non ridondanti

Una **lista di copertura** è l'insieme (qualunque) di sottocubi che coprono tutti gli SO1 della mappa. La lista dei sottocubi principali è una lista di copertura. Una **lista di copertura non ridondante** è una lista di copertura che smette di essere tale quando si toglie un sottocubo.

Come avevamo visto algebricamente, la lista degli SO1 (che corrisponderebbe ai mintermini) non è ridondante, ma la lista dei sottocubi principali (che corrisponderebbe agli implicanti) può esserlo.

Possiamo classificare i sottocubi principali come segue:

- Alcuni sottocubi sono gli unici a coprire un dato sottocubo di ordine 1. In questo caso, si chiamano sottocubi **essenziali**, e costituiscono il **cuore** (*core*) della mappa.
- Alcuni insiemi di sottocubi possono essere disposti, per cui rimuovere uno a caso fra i sottocubi compresi non risulta in variazioni della copertura. Questi si dicono **semplicemente eliminabili**.
- I sottocubi interamente contenuti all'interno di sottocubi essenziali sono sempre ridondanti, e si dicono **assolutamente eliminabili**.

Rimuovendo i sottocubi assolutamente eliminabili, e mantenendo solo un sottoinsieme dei sottocubi semplicemente eliminabili, possiamo generare una qualsiasi lista non ridondante a partire dalla lista di copertura data. Vogliamo però selezionare la lista non ridondante che dà costo minimo.

1.3.3 Ricerca delle liste di copertura di costo minimo

Per trovare una **lista di copertura di costo minimo**, applichiamo un determinato criterio quando eliminiamo i sottocubi semplicemente eliminabili. In generale avremo che, per qualsiasi sottocubo semplicemente eliminabile, potremo considerarlo come:

- **Essenziale**, in questo caso resta nella lista di copertura. e qualche altro sottocubo diventa assolutamente eliminabile;
- **Assolutamente eliminabile**, in questo caso si toglie dalla lista di copertura, e qualche altro sottocubo diventa essenziale.

Il criterio adottato è quello di conservare il minor numero di sottocubi possibile. Prendiamo in esempio la mappa:

X_3X_2 \ X_1X_0	00	01	11	10
00	0	0	0	1
01	1	0	0	1
11	1	1	0	0
10	1	1	0	1

	X_3	X_2	X_1	X_0
A	1	-	0	-
B	0	-	1	0
C	-	1	0	0
D	-	0	1	0
E	0	1	-	0
F	1	0	-	0

Si ha che il sottocubo A è essenziale: altrimenti non potremmo mai coprire 1101 e 1001. Tutti gli altri cubi sono semplicemente eliminabili. Partiamo quindi dalla configurazione {A}, e decidiamo quali elementi includere successivamente. La prima zona che osserviamo è quella di E: i sottocubi semplicemente eliminabili sono {A, B, C, E}. Eliminando C e B, resta E, e viceversa: chiaramente è meglio lasciare solo E. Possiamo rendere questo graficamente come:

{A}

{A, E} {A, B, C}

La prossima zona di interesse è rappresentata da B, D, E e F. Si può continuare con la struttura ad albero:

{A}

{A, E} {A, B, C}

{A, E, D} {A, E, B, F} {A, C, B, D} {A, C, B, F}

Scegliamo quindi {A, E, D}.