



Sintesi Combinatoria

Sintesi di reti combinatorie a più livelli: Introduzione

Motivazioni e Introduzione
Modello per reti combinatorie a più livelli
Trasformazioni e Algoritmi



Sintesi di reti combinatorie a più livelli: Introduzione

- Obiettivo della sintesi logica: **ottimizzazione delle cifre di merito area e prestazioni**
 - Reti combinatorie a **due livelli**: area e ritardo sono ridotti contemporaneamente.
 - Reti combinatorie a **più livelli**: area e ritardo non procedono nella stessa direzione
- Le **reti a più livelli** portano in generale a **soluzioni più efficienti in termini di area/prestazioni** e consentono un utilizzo migliore delle librerie



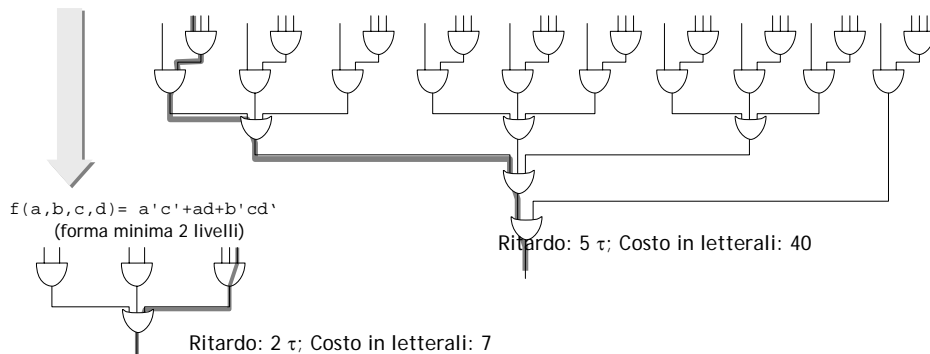
- 2 -



Sintesi di reti combinatorie a più livelli: Introduzione

- Esempio**—(Reti combinatorie a due livelli: Area e tempo sono ridotti contemporaneamente)
 - Ipotesi: porte con un massimo di 3 ingressi (ritardo uniforme: τ)

$$f(a,b,c,d) = a'b'c'd' + a'b'c'd + a'b'cd' + a'bc'd' + a'bc'd + ab'c'd + ab'cd' + abcd + abcd$$



- 3 -



Sintesi di reti combinatorie a più livelli: Introduzione

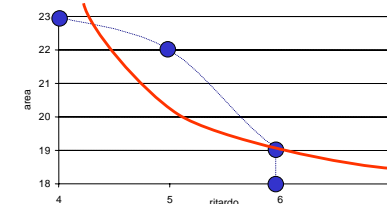
- Esempio**—(Reti combinatorie a più livelli: trade-off area/prestazioni)
 - Ipotesi: porte con un massimo di 3 ingressi (ritardo uniforme: τ)

$$f = 1' + c' * g * h' + a * b' * k' + g * k' + a' * b' * c' * d' * e' + a * d' * e' * f' + e' * g' * i' + e' * j'; \text{ Ritardo: } 4 \tau; \text{ Costo: } 23$$

$$f = 1' + c' * g * h' + k' * (a * b' + g) + a' * b' * c' * d' * e' + a * d' * e' * f' + e' * g' * i' + e' * j'; \text{ Ritardo: } 5 \tau; \text{ Costo: } 22$$

$$f = 1' + c' * g * h' + k' * (a * b' + g) + e' * (a' * b' * c' * d' + a * d' * f' + g' * i' + j'); \text{ Ritardo: } 6 \tau; \text{ Costo: } 19$$

$$f = 1' + c' * g * h' + k' * (a * b' + g) + e' * (d' * (a' * b' * c' + a * f') + g' * i' + j'); \text{ Ritardo: } 6 \tau; \text{ Costo: } 18$$



- 4 -



Sintesi di reti combinatorie a più livelli: Introduzione

- Nella realizzazione di **reti combinatorie multi-livello**, più che ricercare un ottimo, si cerca una **soluzione ragionevole in termini di area e prestazioni**.
- Sarebbe più corretto parlare di **sintesi** invece che di ottimizzazione. La sintesi può prevedere:
 - Minimizzazione dell'area (con vincolo sul ritardo)
 - Minimizzazione del ritardo (con vincolo sull'area)
- Le **operazioni e trasformazioni** definite per la sintesi multi-livello hanno come scopo base quello di manipolare l'espressione logica della rete combinatoria in modo da **individuare ed estrarre sotto-espressioni logiche comuni** nell'espressione di partenza
 - questo consente, in generale, di avere realizzazioni più efficienti (con riuso) in termini di porte utilizzate, rispetto all'ottimizzazione a due livelli, con tempi di propagazione peggiori

- 5 -



Sintesi di reti combinatorie a più livelli: Introduzione

- Ottimizzazione a più livelli:
 - Vantaggi:
 - Più efficiente in termini di area e prestazioni.
 - Permette di utilizzare elementi di libreria.
 - Svantaggi:
 - Maggiore complessità della ottimizzazione.
- Metodi di ottimizzazione:
 - Esatti
 - Complessità computazionale estremamente elevata: inaccettabili.
 - **Euristici**
 - Definizione di euristica: "*procedimento non rigoroso (approssimativo, intuitivo) che permette di conseguire un risultato la cui qualità è paragonabile a quella ottenuta con metodi rigorosi*"

- 6 -



Sintesi di reti combinatorie a più livelli: Introduzione

- **Euristica del problema di ottimizzazione - due passi:**
 - a) Si produce una soluzione ottimale ignorando i vincoli di realizzazione
 - fan_in, fan_out, elementi di libreria...

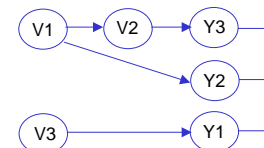
La soluzione è ottenuta tramite sequenze di **trasformazioni** applicate in **modo iterativo**. Le trasformazioni sono basate anche sulle **proprietà algebriche** delle espressioni Booleane. La rete è definita **ottima** rispetto ad un insieme di trasformazioni, quando una nuova l'applicazione di queste **non** può più **migliorare la funzione di costo**.
 - b) Si raffina il risultato considerando i vincoli strutturali
 - *library mapping* (o *library binding*).
 - Risultato dell'ottimizzazione è di inferiore qualità rispetto ad una ottimizzazione che considera contemporaneamente i punti a) e b) ma risulta computazionalmente più semplice.
- In questa sezione si analizza solo il punto relativo **all'identificazione della soluzione ottimale** (punto a).

- 7 -



Sintesi multilivello di reti combinatorie - *esempio (1)*

- Data una rete combinatoria con ingressi (a, b, c, d, e, f), uscite (Y1, Y2, Y3) e rappresentata dal seguente modello (**rete multilivello**)



Equazioni dei nodi

$$V_1 = \overline{abd} + ef$$

$$V_2 = a\overline{V}b + aV_1c + V_1cd + \overline{V}bd$$

$$V_3 = \overline{bde} + ab\overline{c}e + bde + \overline{ab}c\overline{e} + \overline{bdef}$$

$$Y_1 = V_3 + a\overline{f}$$

$$Y_2 = \overline{V}1ab + \overline{ab}c\overline{d}e + \overline{bc}d\overline{e} + \overline{V}1abf + \overline{ab}cde$$

$$Y_3 = \overline{ab}f + \overline{db}f + ae + a\overline{V}2 + de + d\overline{V}2$$

- 8 -



Sintesi multilivello di reti combinatorie - esempio (2)

1. E' possibile manipolare la rete con una sequenza di trasformazioni in modo da ridurre il costo dell'implementazione?
2. Quale cifra di merito si può utilizzare per valutare la bontà della soluzione?
3. Quali trasformazioni (di che tipo e a che scopo) si possono utilizzare (in modo tale che la sintesi possa essere effettuata in modo automatico)?

1. SI
2. Letterali
3. Manipolazione algebrica delle espressioni dei nodi con lo scopo di ridurre la complessità dei nodi, il costo della rete e individuare sottoespressioni comuni a più nodi

- 9 -



Sintesi multilivello di reti combinatorie - esempio (3)

Applicare in sequenza alla rete multi livello le trasformazioni sotto indicate

- a) **COSTO()**: Calcolo del numero di letterali. La funzione COST() calcola il costo in letterali indipendentemente dalla forma (SOP o multi livello) delle espressioni algebriche dei nodi.
- b) **SIMPLIFY(Y₂)**: Minimizzazione a due livelli di Y₂.
- c) **SIMPLIFY(V₃)**: Minimizzazione a due livelli di V₃.
- d) **ELIMINATE(V₃, +3)**: Eliminazione del nodo V₃: la trasformazione viene accettata solo se l'incremento di area, dovuto all'eliminazione è inferiore o uguale alla soglia data (+3).
- e) **FACTOR(V₂)**: Fattorizzazione del nodo V₂.
- f) **COSTO()**: Calcolo del numero di letterali.
- g) **[V₄] = EXTRACT(V₂, Y₃)**: Estrazione di un fattore comune a V₂ e Y₃. Il nodo V₄ derivato dall'estrazione può essere un nuovo nodo o un nodo già presente nella rete.
- h) **COSTO()**: Calcolo del numero di letterali.

- 10 -



Sintesi multilivello di reti combinatorie - esempio (4)

$$V_1 = \overline{abd} + ef$$

$$V_2 = a\overline{V}b + aV_1c + V_1cd + \overline{V}bd$$

$$V_3 = \overline{bde} + \overline{abce} + bde + \overline{abce} + \overline{bdef}$$

$$Y_1 = V_3 + a\overline{f}$$

$$Y_2 = \overline{V}ab + \overline{abcde} + \overline{bcde} + \overline{V}abf + \overline{abcde}$$

$$Y_3 = \overline{abf} + \overline{dbf} + ae + a\overline{V} + de + d\overline{V}$$

Costo dei nodi in letterali

$$C_{V1} = 5$$

$$C_{V2} = 12$$

$$C_{V3} = 18$$

$$C_{Y1} = 3$$

$$C_{Y2} = 21$$

$$C_{Y3} = 14$$

- 11 -



Sintesi multilivello di reti combinatorie - esempio (5)

COST(): 73

SIMPLIFY(Y₂): Minimizzazione a due livelli di Y₂. Tramite mappe di Karnaugh o manipolazione algebrica ottima, il risultato della minimizzazione è sotto riportato con $\Delta C_{Y2} = -7$ ($C_{Y2} = 14$)

$$Y_2 = \overline{V}ab + \overline{abcde} + \overline{bcde} + \overline{V}abf + \overline{abcde}$$

$$Y_2 = \overline{V}a(b + \overline{b}f) + \overline{abcd}(\overline{e} + e) + \overline{bcde}$$

$$Y_2 = \overline{V}a(b + f) + \overline{abcd} + \overline{bcde}$$

$$Y_2 = \overline{V}ab + \overline{V}af + \overline{abcd} + \overline{bcde}$$

SIMPLIFY(V₃): Minimizzazione a due livelli di V₃. Il risultato della minimizzazione è sotto riportato con $\Delta C_{V3} = -13$ ($C_{V3} = 5$)

$$V_3 = \overline{bde} + \overline{abce} + bde + \overline{abce} + \overline{bdef}$$

$$V_3 = \overline{bde} + \overline{ace}(b + \overline{b}) + bde + \overline{bdef}$$

$$V_3 = \overline{bde}(1 + f) + \overline{ace} + bde$$

$$V_3 = de(\overline{b} + b) + \overline{ace}$$

$$V_3 = de + \overline{ace}$$

- 12 -



Sintesi multilivello di reti combinatorie - esempio (6)

ELIMINATE($V_3, +3$): l'unico nodo che contiene V_3 è Y_1

$$Y_1 = V_3 + a\bar{f}$$

L'eliminazione di V_3 porta ad un incremento di area pari a -1, quindi la trasformazione viene accettata, V_3 eliminato (quindi $C_{V3}=0$) e Y_1 diventa come sotto (con $C_{Y1}=7$)

$$Y_1 = a\bar{c}e + de + a\bar{f}$$

FACTOR(V_2): Fattorizzazione del nodo V_2 .

$$V_2 = a\bar{V}b + aV_1c + V_1cd + \bar{V}bd$$

Il risultato è sotto riportato con $\Delta C_{V2}=-6$ ($C_{V2}=6$)

$$V_2 = (a+d)(\bar{V}b + V_1c)$$

Costo dei nodi in letterali dopo le trasformazioni effettuate

$$C_{V1}=5$$

$$C_{V2}=6$$

$$C_{V3}=0$$

$$C_{Y1}=7$$

$$C_{Y2}=14$$

$$C_{Y3}=14$$



Sintesi multilivello di reti combinatorie - esempio (7)

COST(): 46

[V_4] = EXTRACT(V_2, Y_3): Estrazione di un fattore comune a V_2 e Y_3 .

$$V_2 = (a+d)(\bar{V}b + V_1c)$$

$$Y_3 = a\bar{b}f + d\bar{b}f + ae + a\bar{V}2 + de + d\bar{V}2$$

$$Y_3 = \bar{b}f(a+d) + e(a+d) + \bar{V}2(a+d)$$

$$Y_3 = (a+d)(\bar{b}f + e + \bar{V}2)$$

E quindi

$$V_4 = (a+d)$$

$$V_2 = V_4(\bar{V}b + V_1c)$$

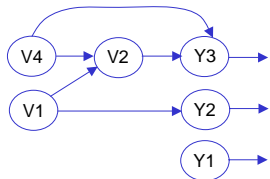
$$Y_3 = V_4(\bar{b}f + e + \bar{V}2)$$



Sintesi multilivello di reti combinatorie - esempio (8)

L'applicazione di tutte le trasformazioni ha portato alla rete seguente che ha un costo pari a 38 letterali

Le equazioni dei nodi sono:



$$V_1 = \bar{a}bd + ef$$

$$V_2 = V_4(\bar{V}b + V_1c)$$

$$V_4 = (a+d)$$

$$Y_1 = a\bar{c}e + de + a\bar{f}$$

$$Y_2 = \bar{V}1ab + \bar{V}1af + \bar{a}bcd + \bar{b}cde$$

$$Y_3 = V_4(\bar{b}f + e + \bar{V}2)$$



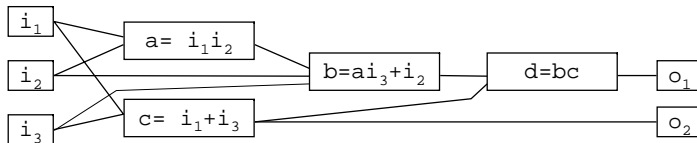
Sintesi di reti combinatorie a più livelli: Modello della rete (1)

- Nella sintesi multilivello, il **modello** utilizzato per rappresentare un circuito combinatorio è un **grafo orientato aciclico**
 - DAG - Direct Acyclic Graph
- **Grafo per reti combinatorie**
 - È un grafo orientato $G(V,E)$ aciclico
 - V : insieme dei nodi
 - E : insieme degli archi
- V è partizionato negli insiemi:
 - nodi di ingresso $\forall I$ (Primary Inputs - PI)
 - nodi di uscita $\forall O$ (Primary Outputs - PO)
 - nodi interni $\forall G$: Sono moduli della rete combinatoria a cui è associata una funzione combinatoria scalare (una sola uscita)



Sintesi di reti combinatorie a più livelli: Modello della rete (2)

- E' un modello comportamentale/strutturale
 - **Strutturale**: connessioni.
 - **Comportamentale**: ad ogni nodo è associata una funzione.
 - Nel modello considerato, ogni **funzione è a due livelli** con una sola uscita.
- Il modello è bipolare e non gerarchico
 - **Bipolare**: Ogni arco può assumere valore 0 o 1.

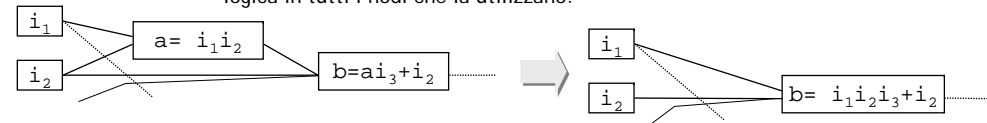


- 17 -



Sintesi di reti combinatorie a più livelli: Trasformazioni per reti logiche (1)

- Metodi euristici
 - Realizzano un miglioramento iterativo della rete logica mediante **trasformazioni** logiche che **conservano il comportamento di I/O** del grafo
- Rispetto al grafo che rappresenta la rete combinatoria, sono possibili **due tipi di trasformazioni**:
 - **Locali**: modificano localmente (la funzione di) un nodo non toccando la struttura della rete.
 - Esempio: la fattorizzazione di un nodo
 - **Globali**: modificano anche la struttura della rete
 - Esempio: l'eliminazione di un nodo nella rete sostituendo la sua espressione logica in tutti i nodi che la utilizzano.



- 18 -



Sintesi di reti combinatorie a più livelli: Trasformazioni per reti logiche (2)

- Le trasformazioni logiche modificano sia l'**area** sia le **prestazioni** poiché agiscono:
 - Sulle funzioni locali;
 - sul numero dei letterali (**area**);
 - Sulle connessioni
 - variazione del n° di nodi (**area**) e del n° nodi del cammino critico (**prestazioni**: n° nodi attraversati, usato come stima per il ritardo di propagazione)
- Sono usate cifre di merito per valutare le trasformazioni
 - Trasformazioni non convenienti sono rifiutate.
- Le trasformazioni sono applicate in modo iterativo.
- La rete è considerata ottimale quando, rispetto ad un insieme di operatori, nessuno di questi la migliora.

- 19 -



Sintesi di reti combinatorie a più livelli: Approcci alla ottimizzazione multi-livello

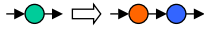
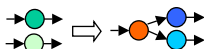
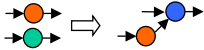
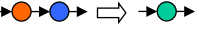
- L'**approccio** tipicamente utilizzato è quello **algoritmico**
 - Ogni trasformazione è associata ad un algoritmo
 - L'algoritmo:
 - determina dove può essere applicata la trasformazione;
 - applica la trasformazione (e la mantiene se porta benefici);
 - termina quando nessuna trasformazione di quel tipo è ulteriormente applicabile.
 - Il maggior vantaggio dell'approccio algoritmico è che trasformazioni di un dato tipo sono sistematicamente applicate alla rete.
 - Algoritmi legati a differenti trasformazioni sono applicati in sequenza.
 - Sfortunatamente **differenti sequenze** possono portare a **soluzioni diverse**.
 - Soluzione: uso di sequenze derivate da **sperimentazioni**.

- 20 -



Sintesi di reti combinatorie a più livelli: Trasformazioni base

□ Le trasformazioni base per manipolare le espressioni logiche sono:

- **Decomposizione di una espressione**
 - aumenta la probabilità di poter sostituire i termini ottenuti con sotto-espressioni già esistenti (globale, aumenta il percorso di I/O)
- **Estrazione di una sotto-espressione da più espressioni**
 - vedi sopra
- **Sostituzione**
 - sostituisce una sotto-espressione in un nodo (diminuisce il n° di letterali nel nodo di partenza) (globale, aumenta il percorso di I/O)
- **Semplificazione di una espressione e fattorizzazione**
 - diminuisce il n° di letterali in una espressione (locale)
- **Eliminazione**
 - inversa della sostituzione, aumenta le prestazioni temporali (globale, diminuisce il percorso di I/O)

- 21 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: eliminazione

□ **Eliminazione:** globale, riduce la lunghezza del percorso I/O



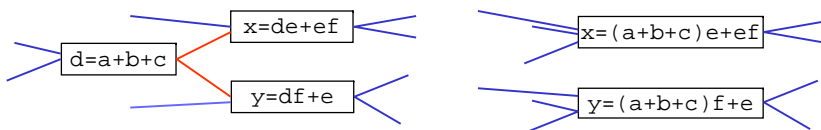
- La lunghezza è calcolata in numero di nodi attraversati.
- **Eliminazione** nella rete di tutti i **vertici con un solo ingresso** e di quelli relativi a funzioni costanti (**Sweep**)
- **Riduzione vincolata (Eliminate** opzione **Val-Intero**) - ad es. eliminate 5
 - L'eliminazione di un vertice è accettata se incrementa l'area di una quantità inferiore a Val-Intero.
 - Ad esempio, l'incremento di area può venire calcolato come $n*1-n-1$, dove 1 è numero di letterali del nodo eliminato mentre n è il numero di nodi che lo assorbono
- **Riduzione non vincolata**
 - tutti i nodi vengono ridotti ad un solo nodo; si ottiene una rete a due livelli.

- 22 -



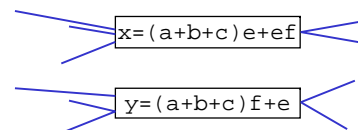
Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: eliminazione

□ Esempio di eliminate 2:



Costo: 3+4+3= 10

incremento di costo: $2*3-2-3 = 1$ (accettato)



Costo: 6+5= 11

□ eliminate -1

	n						
	1	2	3	4	5	6	7
1	-1	-1	-1	-1	-1	-1	-1
2	-1	0	1	2	3	4	5
3	-1	1	3	5	7	9	11
4	-1	2	5	8	11	14	17
5	-1	3	7	11	15	19	23
6	-1	4	9	14	19	24	29
7	-1	5	11	17	23	29	35
8	-1	6	13	20	27	34	41
9	-1	7	15	23	31	39	47

Osservano i dati riportati in tabella, relativi al calcolo di $n*1-n-1$ al variare di n e 1 , si può constatare che l'effetto di eliminate -1 (con $l=1$) è quello di eliminare tutti i nodi composti da un solo letterale.

- 23 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: semplificazione

□ **Semplificazione:** trasformazione locale

- **Semplificazione a due livelli** di ogni nodo (**Simplify**)
 - Metodo esatto (Quine-McClusky) o euristico.
- **Fattorizzazione** di un nodo (**Factor**)
 - All'interno di un nodo, raccoglie a fattore comune alcuni termini.
 - Esempio: (ipotesi: porte a 3 ingressi)

$$f = 1 + c' * g * h + a * b' * k + g * k + a' * b' * c' * d' * e + a * d' * e' * f + e' * g' * i + e' * j'$$

Ritardo: 4 τ ; Costo: 23



$$f = 1 + c' * g * h + k' * (a * b' + g) + e' * (d' * (a' * b' * c' + a * f') + g' * i + j')$$

Ritardo: 7 τ ; Costo: 18

- 24 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: fattorizzazione

o Fattorizzazione

- L'espressione logica fattorizzata può essere ottenuta utilizzando una euristica.
 - **Politica della euristica:** si **pesano i letterali dell'espressione di partenza** con ordinamento lessico-grafico a parità di peso
 - Elemento più a destra per primo
- L'**insieme dei termini prodotto viene ricorsivamente partizionato (blocco della partizione e blocco residuo)** utilizzando come termine di riferimento il letterale che compare con più frequenza.
 - **Ottimizzazione:** tutti i **letterali** che hanno la **stessa cardinalità della partizione** vengono **raccolti contemporaneamente**
- Ad ogni passo della ricorsione le **partizioni** sono **in OR** fra loro mentre i **termini a fattor comune** sono **in AND**.

- 25 -



Sintesi di reti combinatorie a più livelli: Fattorizzazione - esempi

o Esempio: $f = a*c*d + a'*b*c' + a'*b*d' + b'*c*d$ Ritardo: 3τ costo: 12

	a	a'	b	b'	c	c'	d	d'
$a*c*d$	1	0	0	0	1	0	1	0
$a'*b*c'$	0	1	1	0	0	1	0	0
$a'*b*d'$	0	1	1	0	0	0	0	1
$b'*c*d$	0	0	0	1	1	0	1	0
	1	2	2	1	2	1	2	1

Blocco della partizione indotta dal fattore comune dc

	a	a'	b	b'
a	1	0	0	0
b'	0	0	0	1
	1	0	0	1

Blocco residuo della partizione

	a	a'	b	b'	c	c'	d	d'
$a'*b*c'$	0	1	1	0	0	1	0	0
$a'*b*d'$	0	1	1	0	0	0	0	1
	0	2	2	0	0	1	0	1

Fattore comune $a'b$

	c	c'	d	d'
c'	0	1	0	0
d'	0	0	0	1
	1	0	0	1

Ritardo: 3τ costo: 8

$f = d*c*(a+b') + a'*b*(c'+d')$

- 26 -



Sintesi di reti combinatorie a più livelli: Fattorizzazione - esempi

o Esempio1: (forma 2 livelli non ottimizzata)

Ritardo: 4τ costo: 12

$f = a*b*c*d + a'*b!*c*d + !a!*b*c*d + !a!*b!*c*d$

	a	!a	b	!b	c	!c	d	!d
$a*b*c*d$	1	0	1	0	1	0	1	0
$a'*b!*c*d$	1	0	0	1	0	1	1	0
$!a!*b*c*d$	0	1	0	1	1	0	1	0
$!a!*b!*c*d$	0	1	0	1	0	1	1	0
	2	2	1	3	2	2	4	0

Fattore comune d

	a	!a	b	!b	c	!c
$a*b*c$	1	0	1	0	1	0
$a'*b!*c$	1	0	0	1	0	1
$!a!*b*c$	0	1	0	1	1	0
$!a!*b!*c$	0	1	0	1	0	1
	2	2	1	3	2	2

Blocco della partizione indotta dal fattore comune $!b$

	a	!a	c	!c
$a!*c$	1	0	0	1
$!a*c$	0	1	1	0
$!a!*c$	0	1	0	1
	1	2	1	2

Blocco residuo della partizione

$a*b*c$

Blocco della partizione indotta dal fattore comune $!c$

	a	!a
a	1	0
$!a$	0	1
	1	1

Ritardo: 5τ costo: 10

$f = d*(a*b*c + !b*(!a*c + !c*(a + !a)))$

- 27 -



Sintesi di reti combinatorie a più livelli: Fattorizzazione - esempi

o Esempio2: $f = a*b!*d + !a*b*d + !a!*b*d + !a*c*d + !b*c!*d$ Ritardo: 3τ costo: 15

	a	!a	b	!b	c	!c	d	!d
$!a*b*d$	0	1	1	0	0	0	1	0
$!a*c*d$	0	1	0	0	1	0	1	0
$a*b!*d$	1	0	1	0	0	0	0	1
$!a!*b*d$	0	1	0	1	0	0	0	1
$!b*c!*d$	0	0	0	1	1	0	0	1
	1	3	2	2	2	0	2	3

Blocco della partizione indotta dal fattore comune $!d$

	a	!a	b	!b	c	!c
$a*b$	1	0	1	0	0	0
$!a!*b$	0	1	0	1	0	0
$!b*c$	0	0	0	1	1	0
	1	1	1	2	1	0

Blocco residuo della partizione

	a	!a	b	!b	c	!c	d	!d
$!a*b*d$	0	1	1	0	0	0	1	0
$!a*c*d$	0	1	0	0	1	0	1	0
	0	2	1	0	1	0	2	0

Fattore comune $!a*d$

	b	!b	c	!c
b	1	0	0	0
c	0	0	1	0
	1	0	1	0

Ritardo: 5τ costo: 10

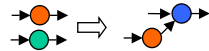
$f = !d*(a*b + !b*(!a*c) + !a*d*(b+c))$

- 28 -



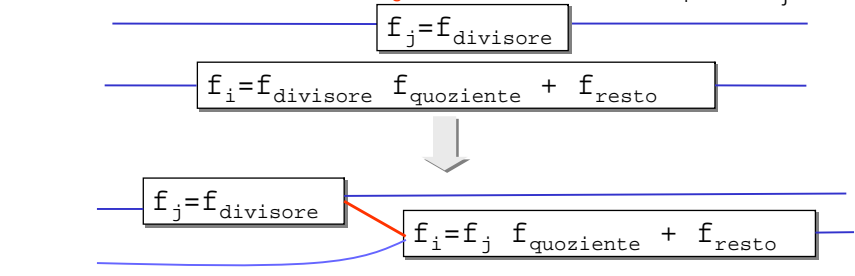
Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: sostituzione

- **Sostituzione (Substitute)**: globale, aumenta la lunghezza del percorso I/O



- Sostituzione di una sotto-espressione mediante una variabile (nodo) già presente nella rete. In generale, ogni sostituzione è accettata se produce guadagno nel numero di letterali.

- Fa uso della **divisione algebrica**; si cerca di ridurre f_i usando f_j

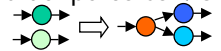


- 29 -

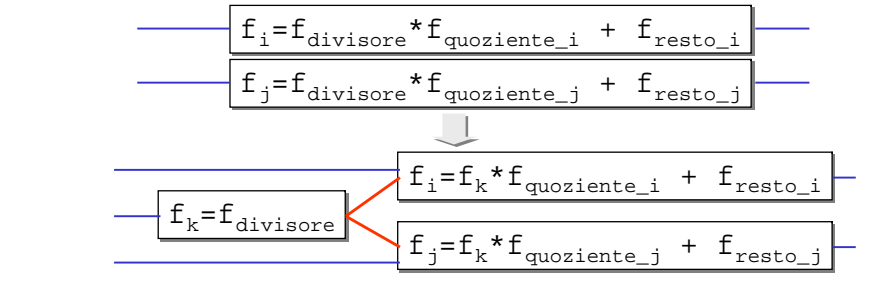


Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: estrazione

- **Estrazione (Extract)** globale, aumenta la lunghezza del percorso I/O



- Estrae una espressione da gruppi di nodi. L'estrazione viene fatta fino a che è possibile.
 - Identificazione un divisore comune a due o più espressioni.
 - Il divisore costituisce un nuovo nodo della rete ed ha per successori i nodi da cui è stato estratto.

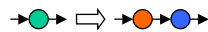


- 30 -

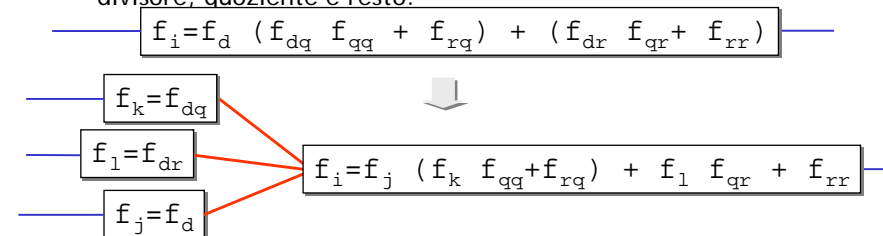


Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: decomposizione algebrica

- **Decomposizione algebrica**: globale, aumenta la lunghezza del percorso I/O



- Riduce le dimensioni di una espressione per:
 - Rendere più semplice l'operazione di *library mapping*.
 - Aumentare la probabilità di successo della sostituzione
- La decomposizione può essere applicata ricorsivamente al divisore, quoziente e resto.



- 31 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: divisori

- **Decomposizione algebrica, estrazione e sostituzione**: come si trovano i divisori?

- **Modello algebrico**: le espressioni Booleane vengono viste come **espressioni algebriche**, cioè come **polinomi di primo grado**, nelle **variabili naturali e complementate**, con **coefficienti unitari**
- Lavorando con il modello algebrico valgono le **proprietà algebriche** mentre quelle dell'algebra booleana non sono valide
- E' definita la **divisione algebrica**: f_{divisore} è un divisore algebrico di $f_{\text{dividendo}}$ se
 - $f_{\text{dividendo}} = f_{\text{divisore}} * f_{\text{quoziente}} + f_{\text{resto}}$ e
 - $f_{\text{quoziente}} * f_{\text{divisore}} \neq 0$ e
 - il supporto di f_{divisore} e di $f_{\text{quoziente}}$ è disgiunto
- Esistono algoritmi diversi per calcolare i divisori di una espressione algebrica

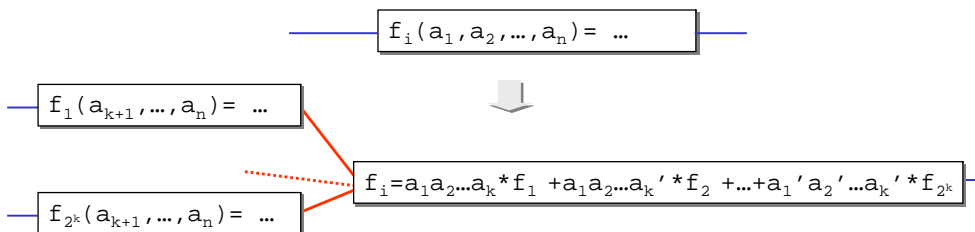
- 32 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: decomposizione disgiuntiva

□ **Decomposizione disgiuntiva semplice (Decompose)** globale, aumenta la lunghezza del percorso I/O

- Riduce le dimensioni di una espressione (v. decomposizione algebrica)
- La decomposizione disgiuntiva semplice può essere applicata ricorsivamente.



- 33 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi: decomposizione disgiuntiva

□ **Decomposizione disgiuntiva (cont.)**

- Deriva dalla applicazione del teorema di *espansione di Shannon*:

$$f(a_1, a_2, \dots, a_n) = a_1 * f_{a_1} + a_1' * f_{\bar{a}_1}$$
- Il risultato, in termini di costo, dipende fortemente dalla decomposizione che viene effettuata sulle variabili di supporto della funzione.
 - Con n variabili il numero di possibili scomposizioni è 2^{n-2}

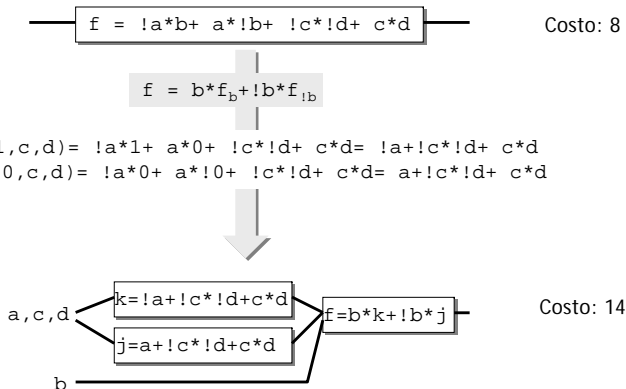
- 34 -



Sintesi di reti combinatorie a più livelli: Decomposizione disgiuntiva - esempi

□ **Esempio1:**

- Esempio: scomposizione disgiuntiva di f rispetto a b



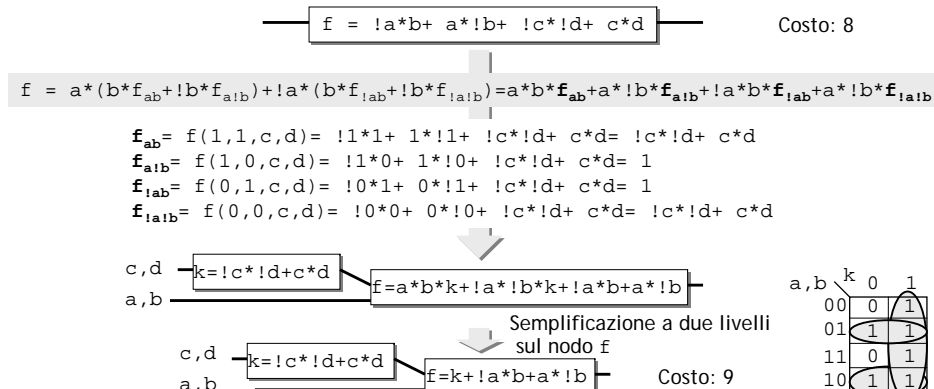
- 35 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi

□ **Esempio1:**

- Esempio: scomposizione disgiuntiva di f rispetto ad ab



- 36 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi

□ Esempio2 (xor):

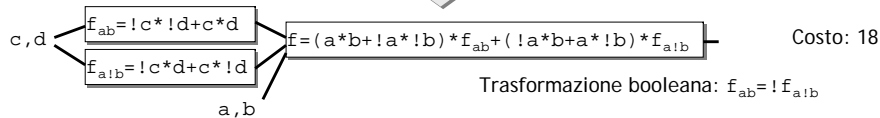
- scomposizione disgiuntiva di f rispetto ad ab

$$f = !a*!b*c*d+!a*!b*c*d+!a*b*c*d+!a*b*c*d+!a*b*c*d+!a*b*c*d+!a*b*c*d+!a*b*c*d$$

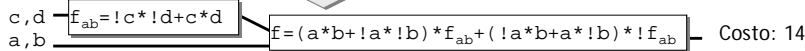
Costo: 32

$$\begin{aligned} f_{ab} &= !c*d+c*d \\ f_{a!b} &= !c*d+c*d \\ f_{!ab} &= !c*d+c*d \\ f_{!a!b} &= !c*d+c*d \end{aligned}$$

$$\begin{aligned} f_{ab} &= f_{!a!b} = !c*d+c*d \\ f_{a!b} &= f_{!ab} = !c*d+c*d \end{aligned}$$



Trasformazione booleana: $f_{ab} = !f_{a!b}$



- 37 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi

□ Esempio3:

$$f = a*!b*c+!a*!b*d+!a*c*d+!c*d$$

Costo: 11

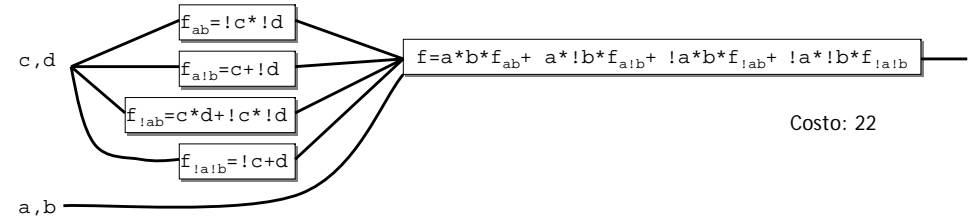
scomposizione disgiuntiva di f rispetto ad ab

$$f_{ab} = 1*0*c+0*0*d+0*c*d+!c*d = !c*d$$

$$f_{a!b} = 1*1*c+0*1*d+0*c*d+!c*d = c+!c*d \Rightarrow c+!d$$

$$f_{!ab} = 0*0*c+1*0*d+1*c*d+!c*d = c*d+!c*d$$

$$f_{!a!b} = 0*1*c+1*1*d+1*c*d+!c*d = d+c*d+!c*d \Rightarrow !c+d$$



- 38 -



Sintesi di reti combinatorie a più livelli: Trasformazioni e algoritmi

□ Esempio3 (cont.):

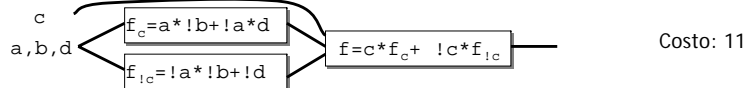
$$f = a*!b*c+!a*!b*d+!a*c*d+!c*d$$

Costo: 11

scomposizione disgiuntiva di f rispetto ad c

$$f_c = a*!b*1+!a*!b*d+!a*1*d+0*d = a*!b+!a*!b*d+!a*d \Rightarrow a*!b+!a*d$$

$$f_{!c} = a*!b*0+!a*!b*d+!a*0*d+1*d = !a*!b*d+!d \Rightarrow !a*!b+d$$



scomposizione disgiuntiva di f rispetto ad a

Costo: 14

$$f_a = 1*!b*c+0*!b*d+0*c*d+!c*d = !b*c+!c*d$$

$$f_{!a} = 0*!b*c+1*!b*d+1*c*d+!c*d = !b*d+c*d+!c*d$$

scomposizione disgiuntiva di f rispetto ad b

Costo: 15

$$f_b = a*0*c+!a*0*d+!a*c*d+!c*d = !a*c*d+!c*d$$

$$f_{!b} = a*1*c+!a*1*d+!a*c*d+!c*d = a*c+!a*d+!a*c*d+!c*d \Rightarrow a*c+!a*d+!c*d$$

scomposizione disgiuntiva di f rispetto ad d

Costo: 13

$$f_d = a*!b*c+!a*!b*1+!a*c*1+!c*0 = a*!b*c+!a*!b+!a*c \Rightarrow a*!b+!a*!b+!a*c$$

$$f_{!d} = a*!b*c+!a*!b*0+!a*c*0+!c*1 = a*!b*c+!c \Rightarrow a*!b+!c$$

- 39 -



Sintesi di reti combinatorie a più livelli: Esercizi

□ Esercizi & Soluzioni di fattorizzazione:

$$f = abcd'+ ab'c'+ a'bc'+ b'cd = c(abd'+ b'd)+ c'(ab'+ a'b)$$

$$f = abcd'+ abc'd + ab'c'd'+ a'bc'd'+ a'b'd + a'cd + b'cd = d'(abc + c'(ab'+ a'b)) + d(abc'+ c(b'+a))+ a'b'$$

$$f = ac'd+ a'b'cd + a'c'd'+ b'c'd = a'b'cd + c'(d(b'+a)+a'd')$$

$$f = abc'+ abd'+ ab'cd+ ac'd'+ a'b'cd+ bc'd' = a(b'cd+ c'd')+ b(a'cd+ d'(c'+a)) + a'c'$$

$$f = ab'cd+ a'b'cd+ a'b'c'+ a'b'd'+ b'c'd' = a'b'cd+ b'(acd+ d'(c'+ a))+ a'c'$$

- 40 -