Università degli studi di Parma

Dipartimento di Ingegneria dell'Informazione

Politecnico
di Milano

Sintesi di Reti Combinatorie

Ottimizzazione di Reti Combinatorie a più Livelli

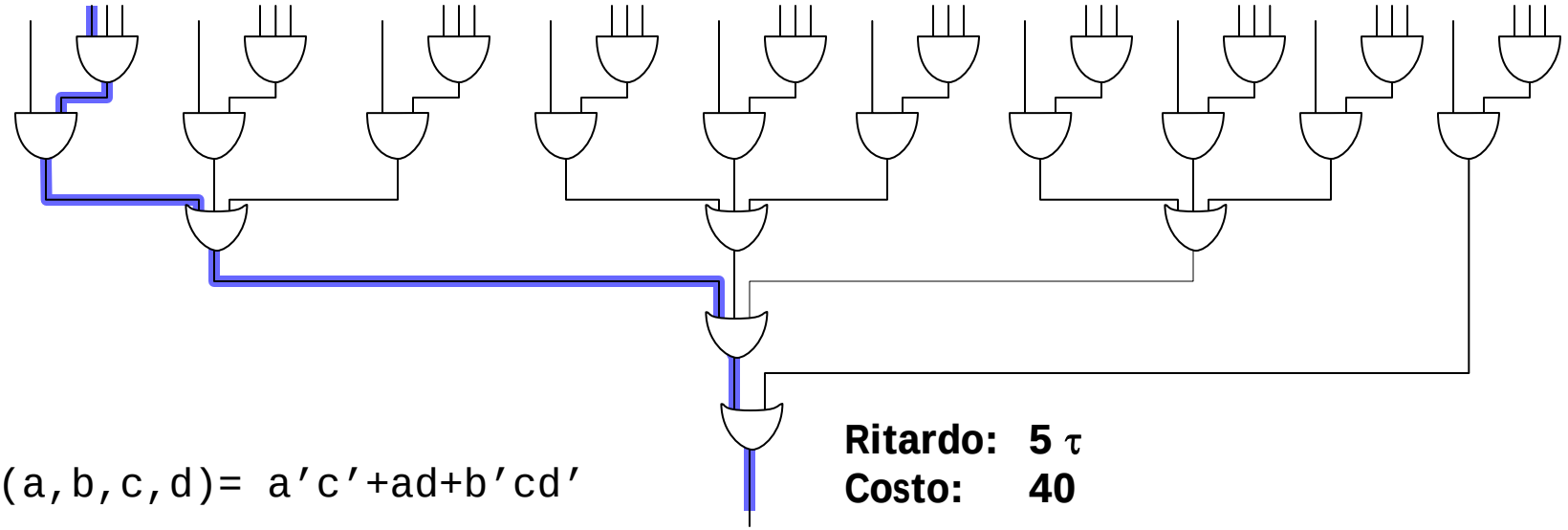
Modello

Trasformazioni

Euristiche

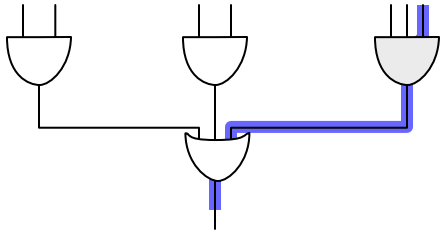
Introduzione

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' + ab'cd + abc'd + abcd



Ritardo: 5 τ
Costo: 40

f(a,b,c,d)= a'c'+ad+b'cd'



Ritardo: 2 τ
Costo: 7

Introduzione

- Problemi nella minimizzazione di reti multi-livello
  - Minimizzazione dell'area (con vincolo sul ritardo)
  - Minimizzazione del ritardo (con vincolo sull'area)
- 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
  - Euristici: non garantiscono di trovare la soluzione ottima

Introduzione

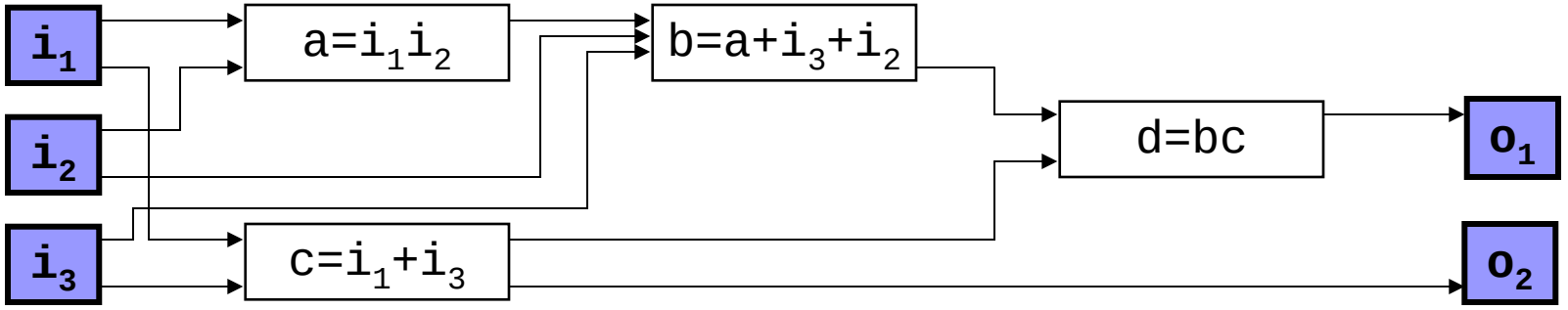
- Euristica del problema di ottimizzazione, in due passi
  - Si produce una soluzione ottimale ignorano i vincoli di realizzazione
    - Fan-in, fan-out, elementi di libreria…
  - Si raffina il risultato considerando i vincoli strutturali
    - library mapping (o library binding)
- Risultato dell'ottimizzazione
  - Di qualità inferiore rispetto ad una ottimizzazione che considera contemporaneamente i due tipi di vincoli
  - Computazionalmente più semplice
- In questa sezione si analizza solo il punto relativo alla identificazione della soluzione ottimale

Modello

- Un circuito combinatorio è rappresentato mediante un Direct Acyclic Graph (DAG)
- È un grafo orientato *G(V,E)* aciclico
  - *V* insieme dei nodi
  - *E* insieme degli archi
- L'insieme *V* dei nodi è partizionato nei sotto-insiemi
  - V_I nodi di ingresso, primary inputs
  - V_O nodi di uscita, primary outputs
  - V_G nodi interni, moduli della rete combinatoria a cui è associata una funzione combinatoria scalare, cioè ad una sola uscita

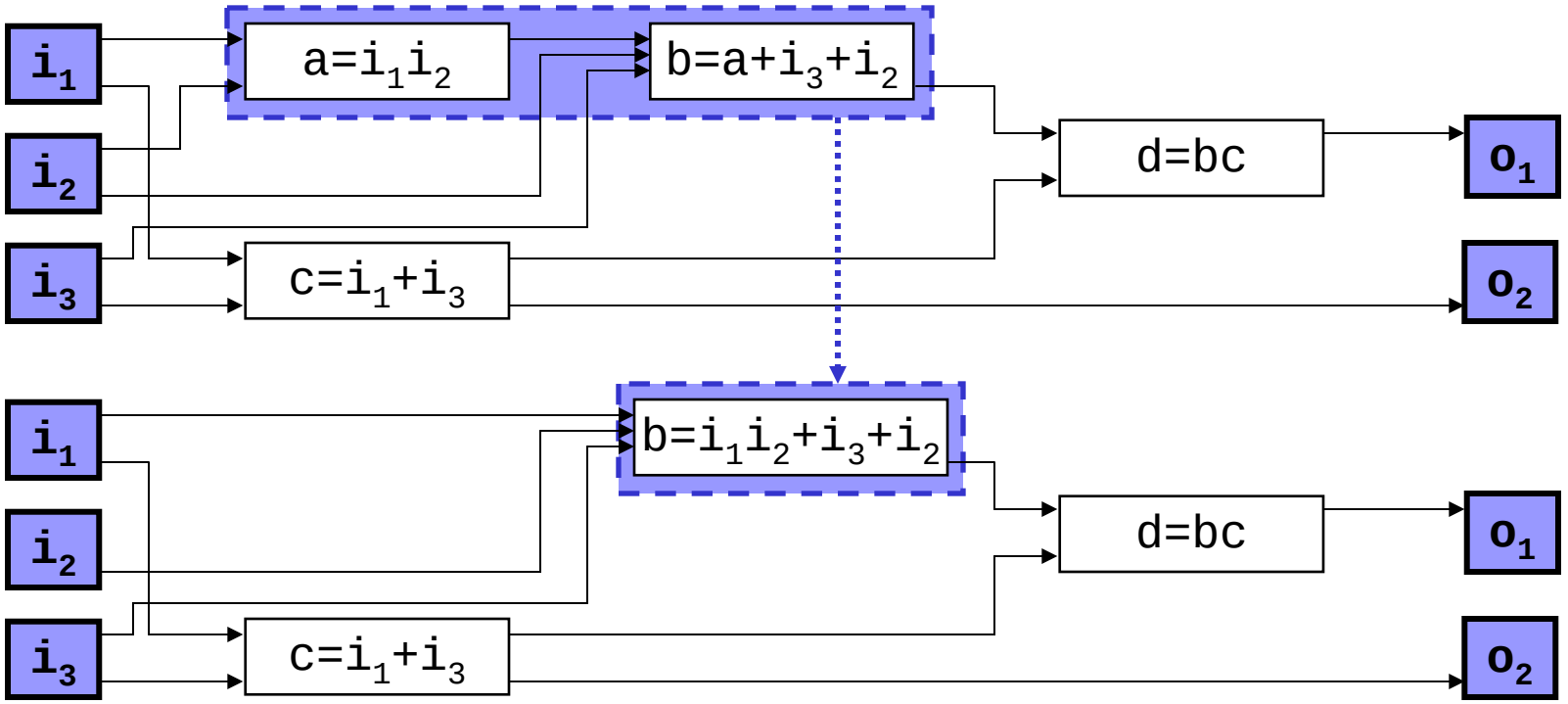
Modello

- Strutturale, non gerarchico
  - Identifica chiaramente le connessioni tra i moduli
- Comportamentale
  - Ad ogni nodo è associata una funzione
  - Nel modello considerato, ogni funzione è a due livelli
- Bipolare
  - Ogni arco può assumere valore 0 o 1



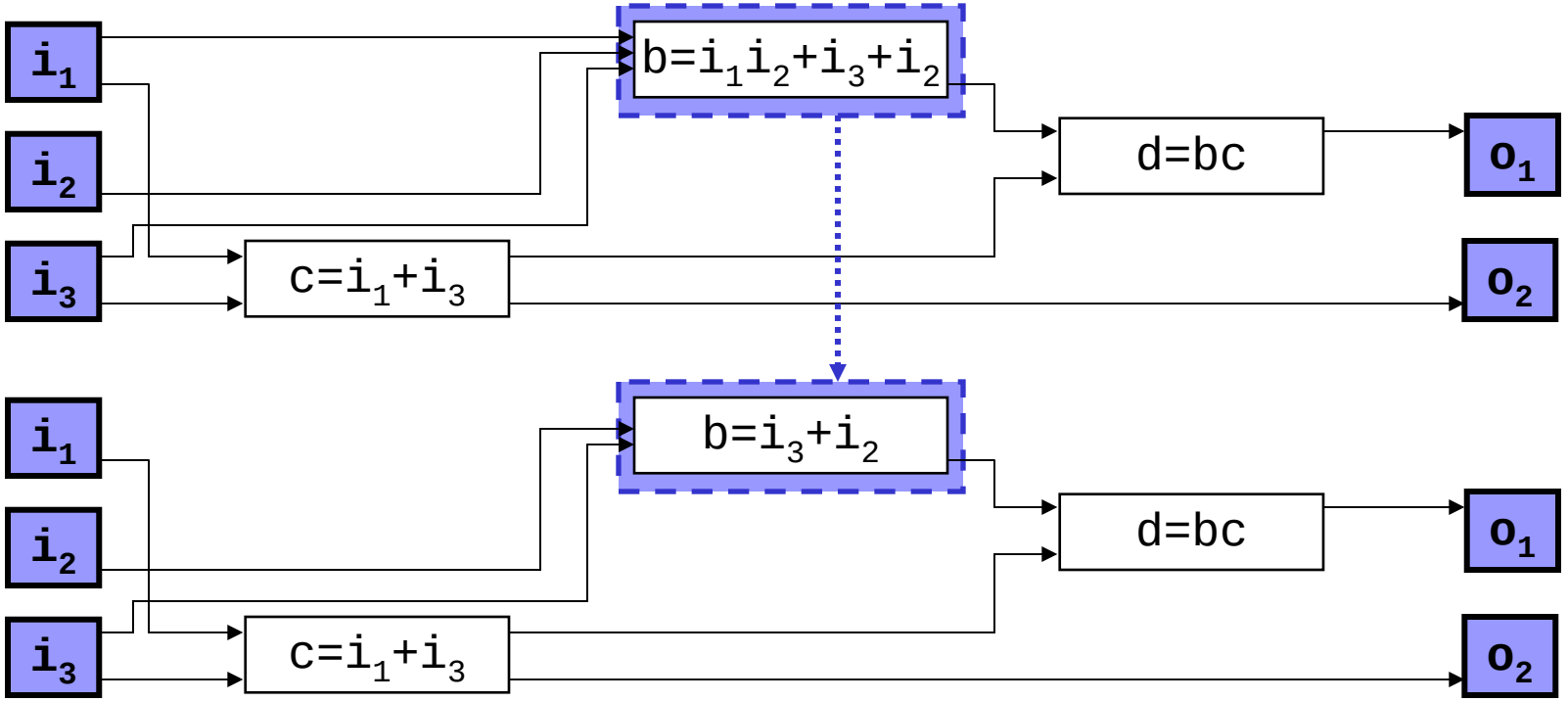
Trasformazioni globali

- Modificano la struttura topologica della rete
  - Modificano i nodi, cioè le funzioni combinatorie
    - Possono portare alla eliminazione di alcuni nodi
  - Modificano gli archi



Trasformazioni locali

- Non modificano la struttura topologica della rete
  - Modificano i nodi, cioè le funzioni combinatorie
  - Possono portare alla eliminazione di alcuni archi



Euristiche

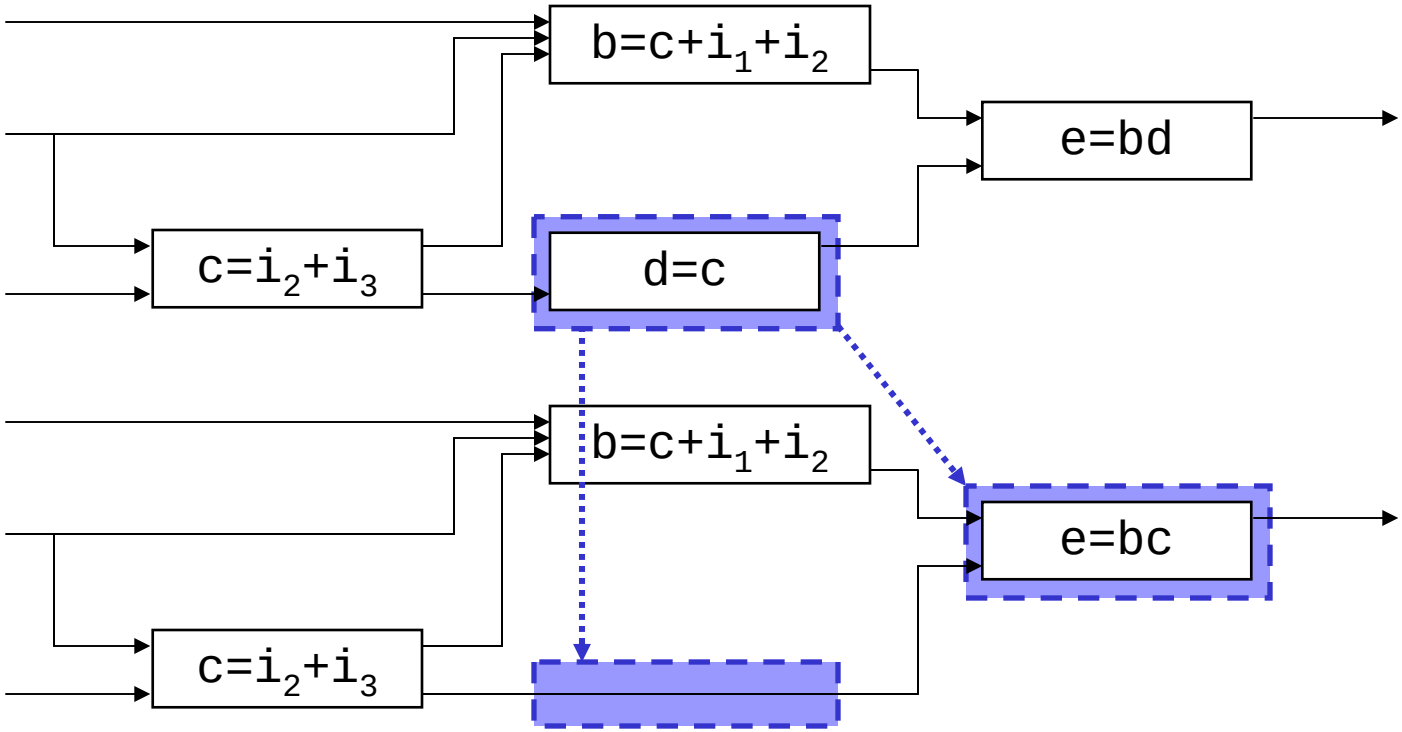
- Le trasformazioni logiche
  - Modificano
    - L'area
    - Le prestazioni
  - Agiscono
    - Sul numero dei letterali
    - Sulle funzioni locali
    - Sulle connessioni
- Si usano cifre di merito per valutare le trasformazioni
  - Trasformazioni non convenienti sono rifiutate
- Le trasformazioni sono applicate in modo iterativo
  - La rete è considerata ottimale quando, rispetto ad un insieme di operatori, nessuno di questi la migliora

Euristiche

- L’approccio euristico utilizzato è quello algoritmico
  - Ogni trasformazione è associata ad un algoritmo
  - L'algoritmo
    - Determina dove può essere applicata la trasformazione
    - Attua la trasformazione e la mantiene se porta benefici
    - L'algoritmo 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

SWEEP

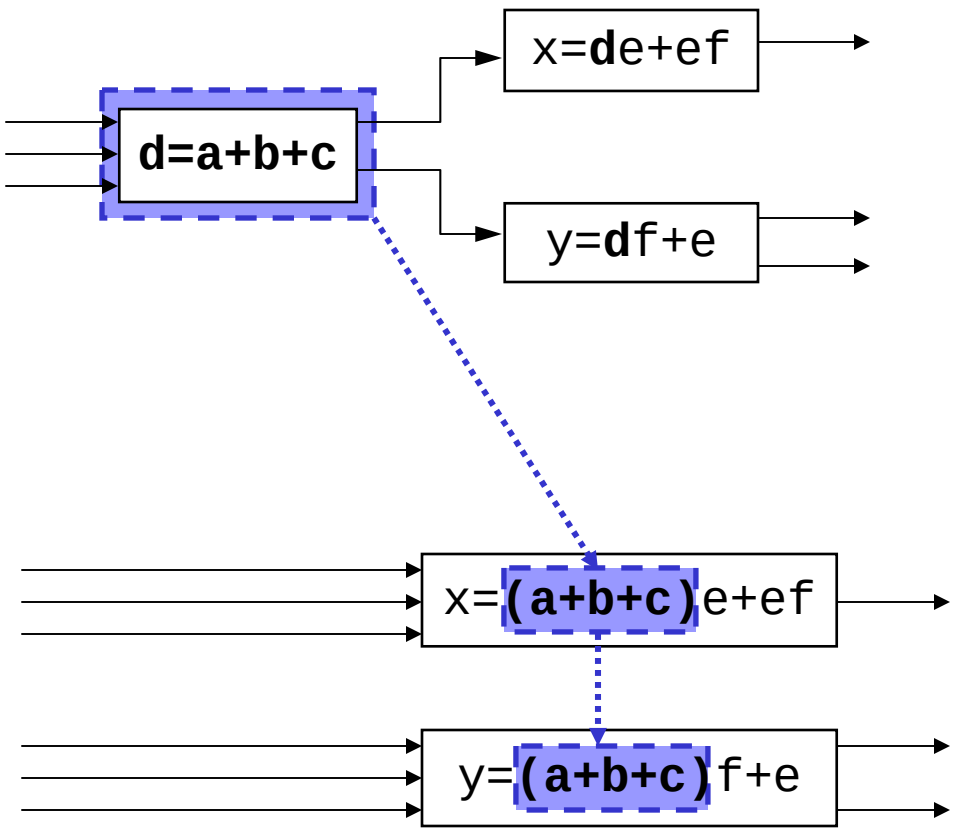
- Elimina dalla rete
  - I nodi con un solo ingresso
  - I nodi relativi a funzioni costanti



ELIMINATE [1]

- Riduzione vincolata con limite *K*
  - Un nodo è assorbito da altri nodi
    - Il nodo assorbito è eliminato
    - L'espressione del nodo è sostituita nei nodi che lo assorbono
  - L'eliminazione di un vertice è accettata se e solo se produce un aumento di area inferiore a *K*
  - L'incremento di area è calcolato come *NxL-N-L* dove
    - *L* Numero di letterali del nodo eliminato
    - *N* 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

ELIMINATE [2]



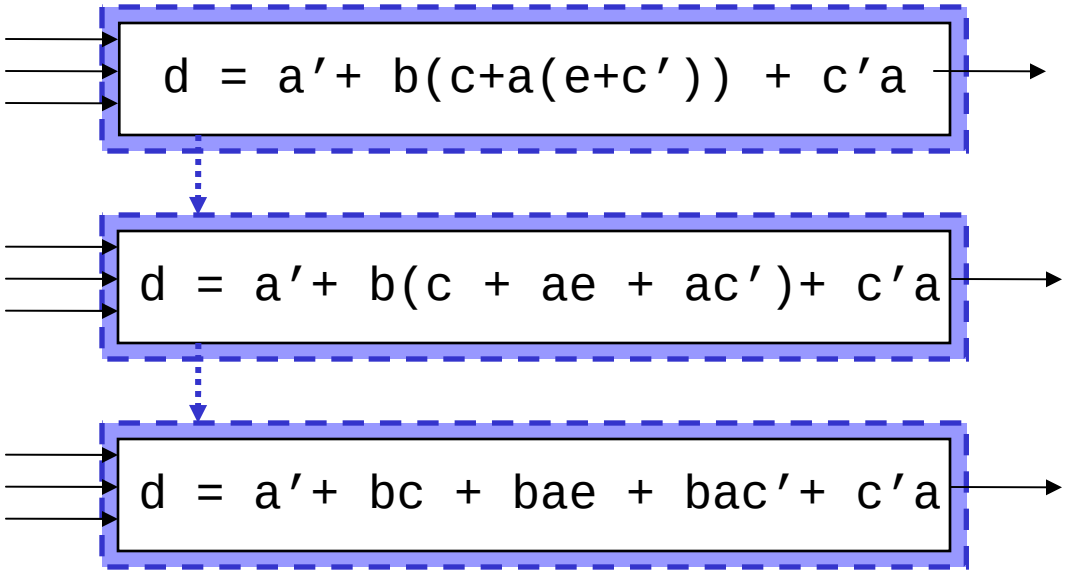
Costo rete originale:
3+4+3 = 10

Incremento di costo:
3x2-3-2 = 1

Costo rete trasformata:
6+5 = 11

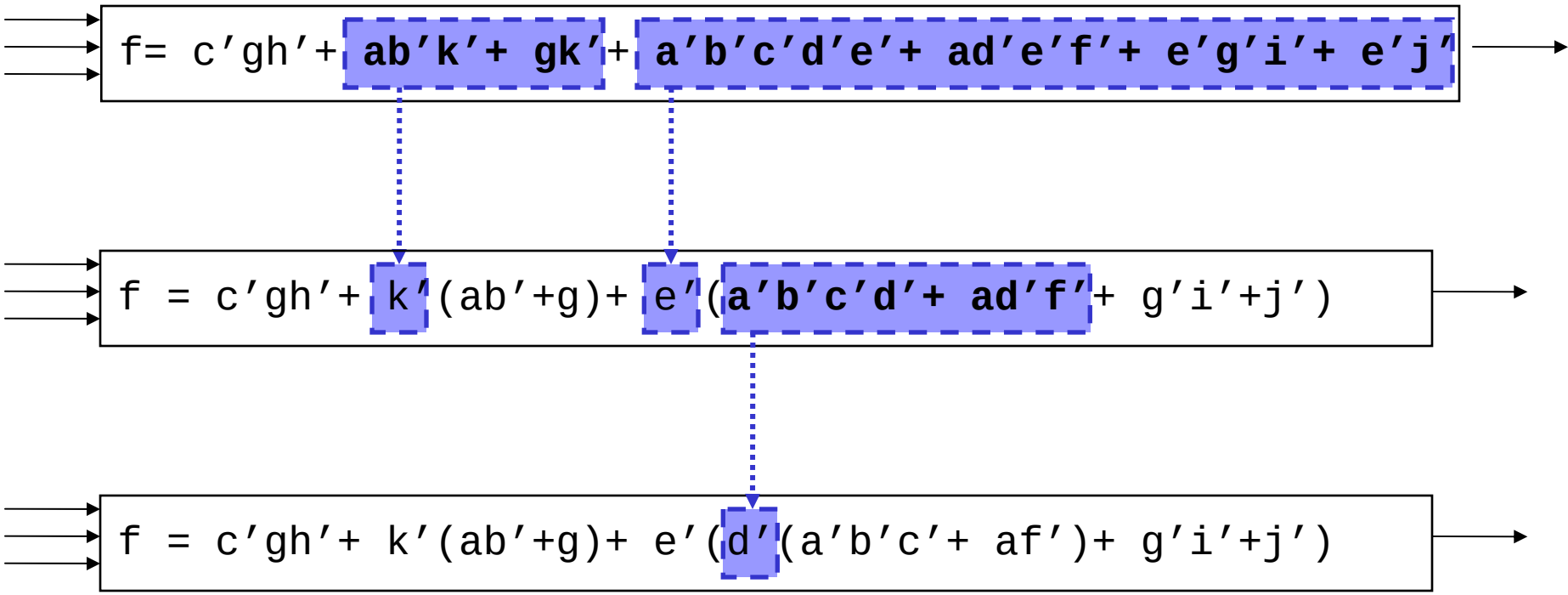
SIMPLIFY

- Trasformazione di ogni nodo in forma a due livelli
- Si possono usare
  - Metodi euristici
  - Metodi esatti (Quine-McClusky)

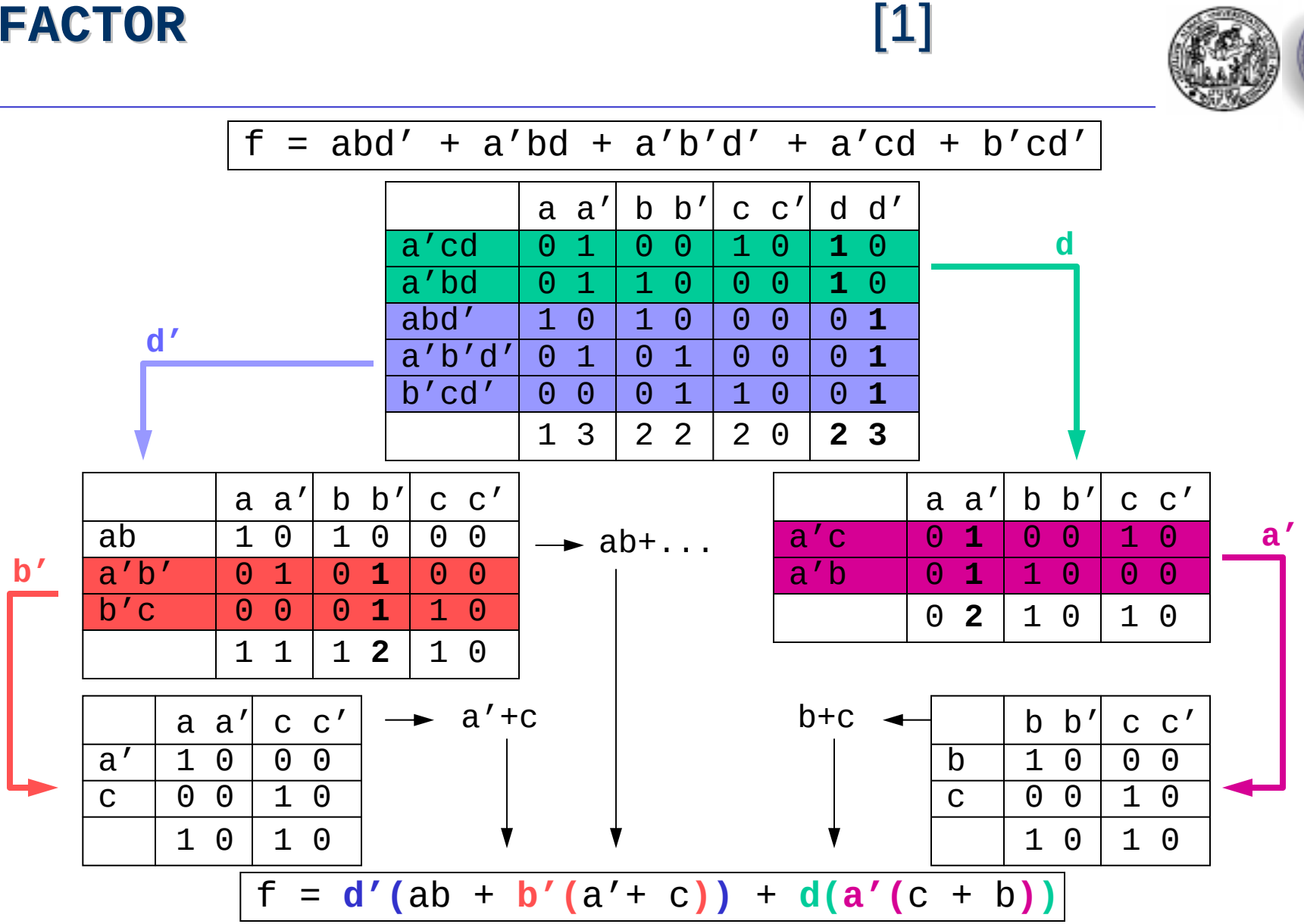


FACTOR [1]

- Raccoglie alcuni termini
  - Localmente ad un nodo



FACTOR [1]



	a a′	b b′	c c′	d d′
a′cd	0 1	0 0	1 0	**1** 0
a′bd	0 1	1 0	0 0	**1** 0
abd′	1 0	1 0	0 0	0 **1**
a′b′d′	0 1	0 1	0 0	0 **1**
b′cd′	0 0	0 1	1 0	0 **1**
	1 3	2 2	2 0	**2 3**

	a a′	b b′	c c′
ab	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

	a a′	b b′	c c′
a′c	0 **1**	0 0	1 0
a′b	0 **1**	1 0	0 0
	0 **2**	1 0	1 0

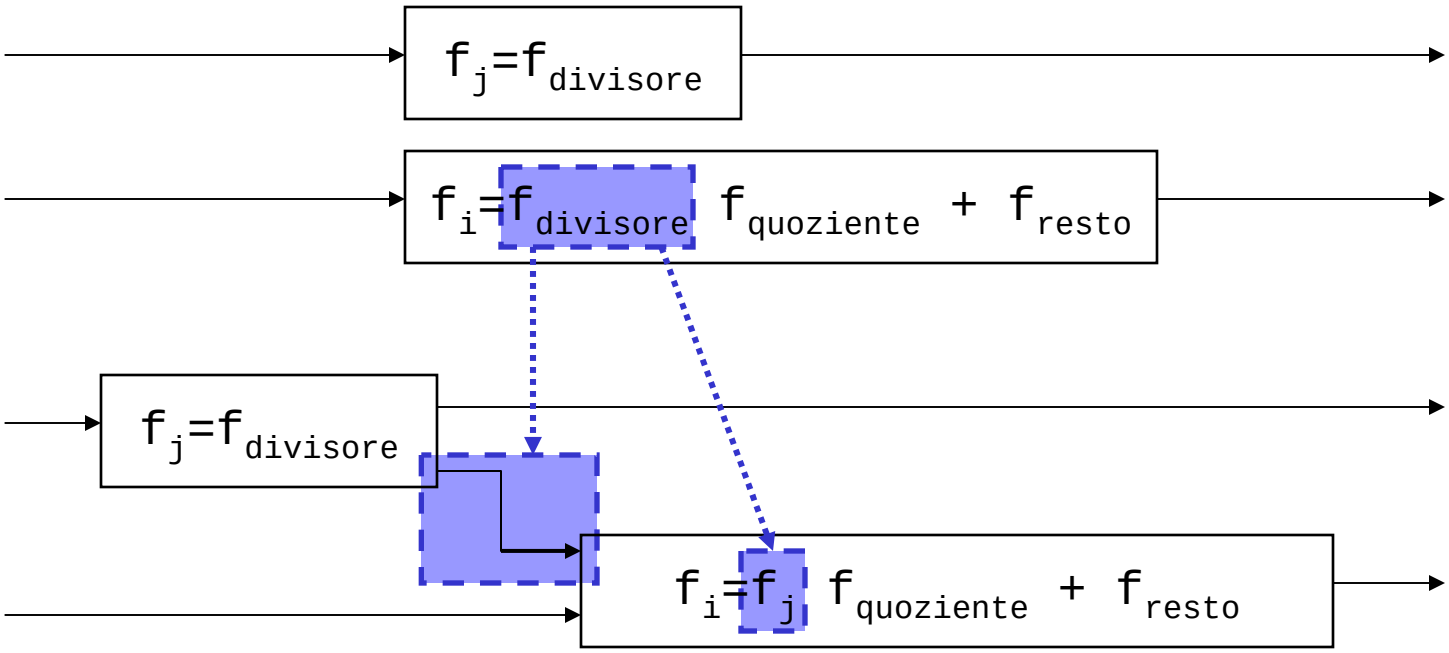
	a a′	c c′
a′	1 0	0 0
c	0 0	1 0
	1 0	1 0

	b b′	c c′
b	1 0	0 0
c	0 0	1 0
	1 0	1 0

f = d′(ab + b′(a′+ c)) + d(a′(c + b))

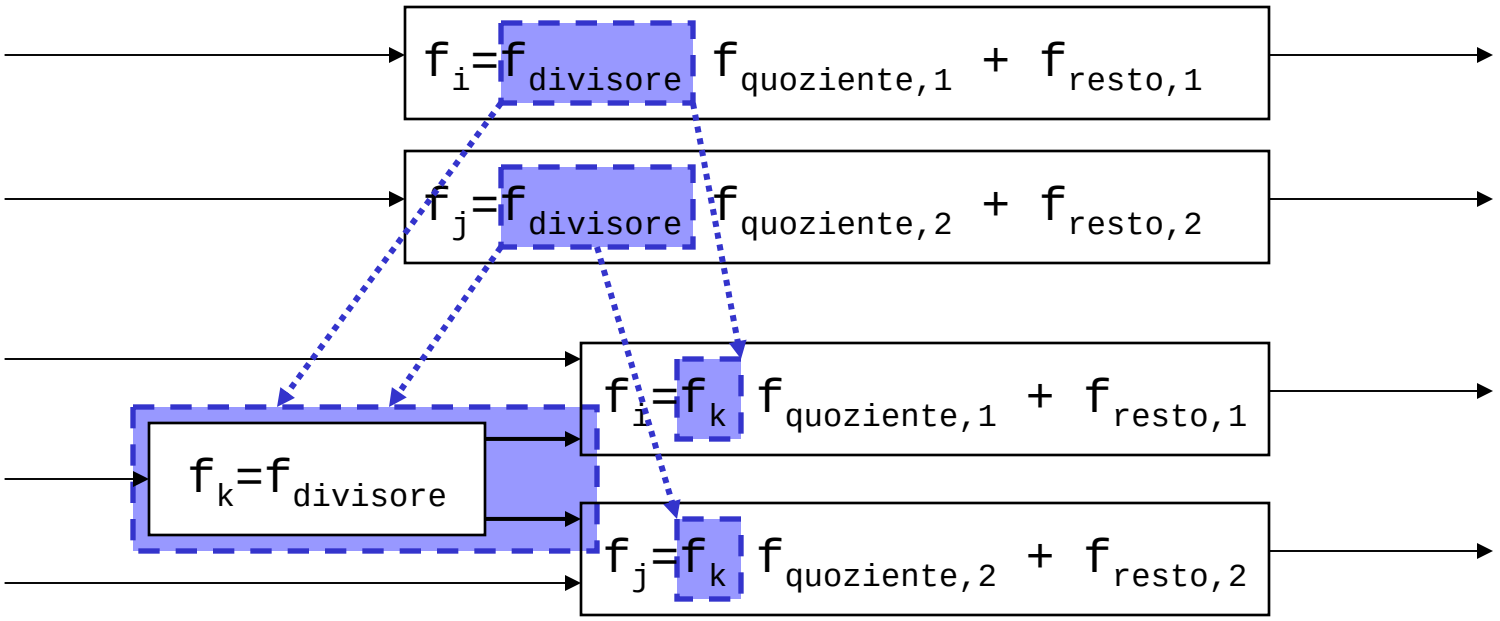
SUBSTITUTE

- Sostituisce una sottoespressione in un nodo usando un nodo già presente nella rete
  - Si basa sulla divisione tra polinomi
  - È accettata se riduce il numero di letterali



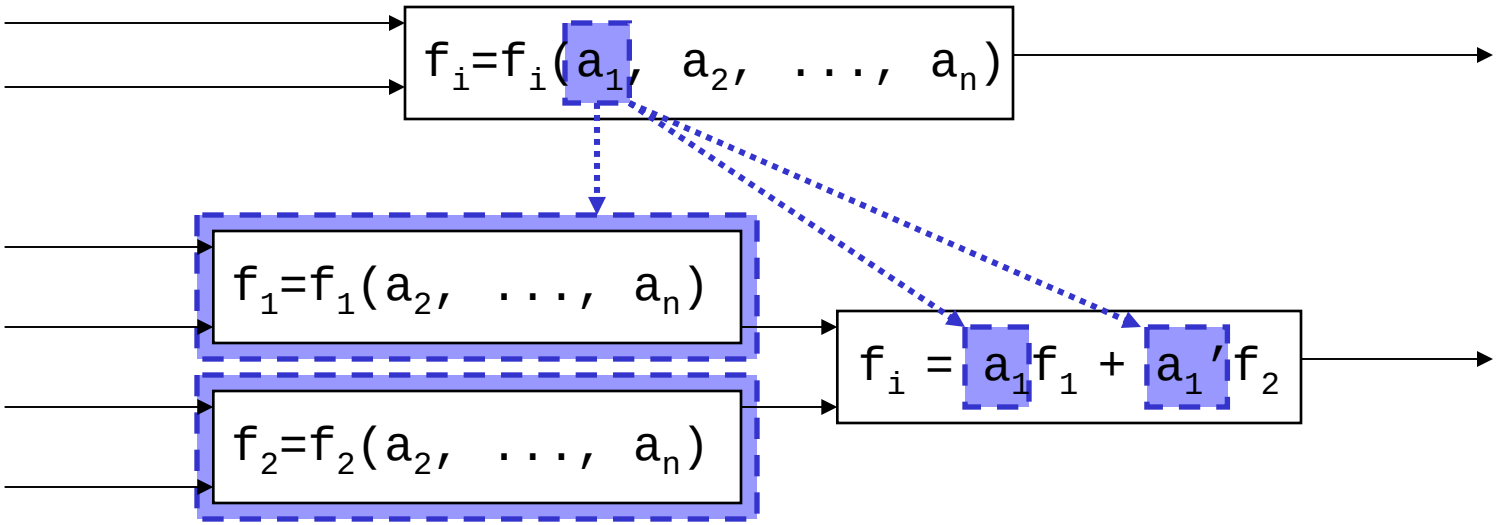
EXTRACT

- Si estrae un n-cubo da gruppi di nodi
  - L'estrazione è ripetuta fino a che è possibile
  - Identificazione un divisore comune a due o più espressioni
  - Il divisore diviene un nuovo nodo

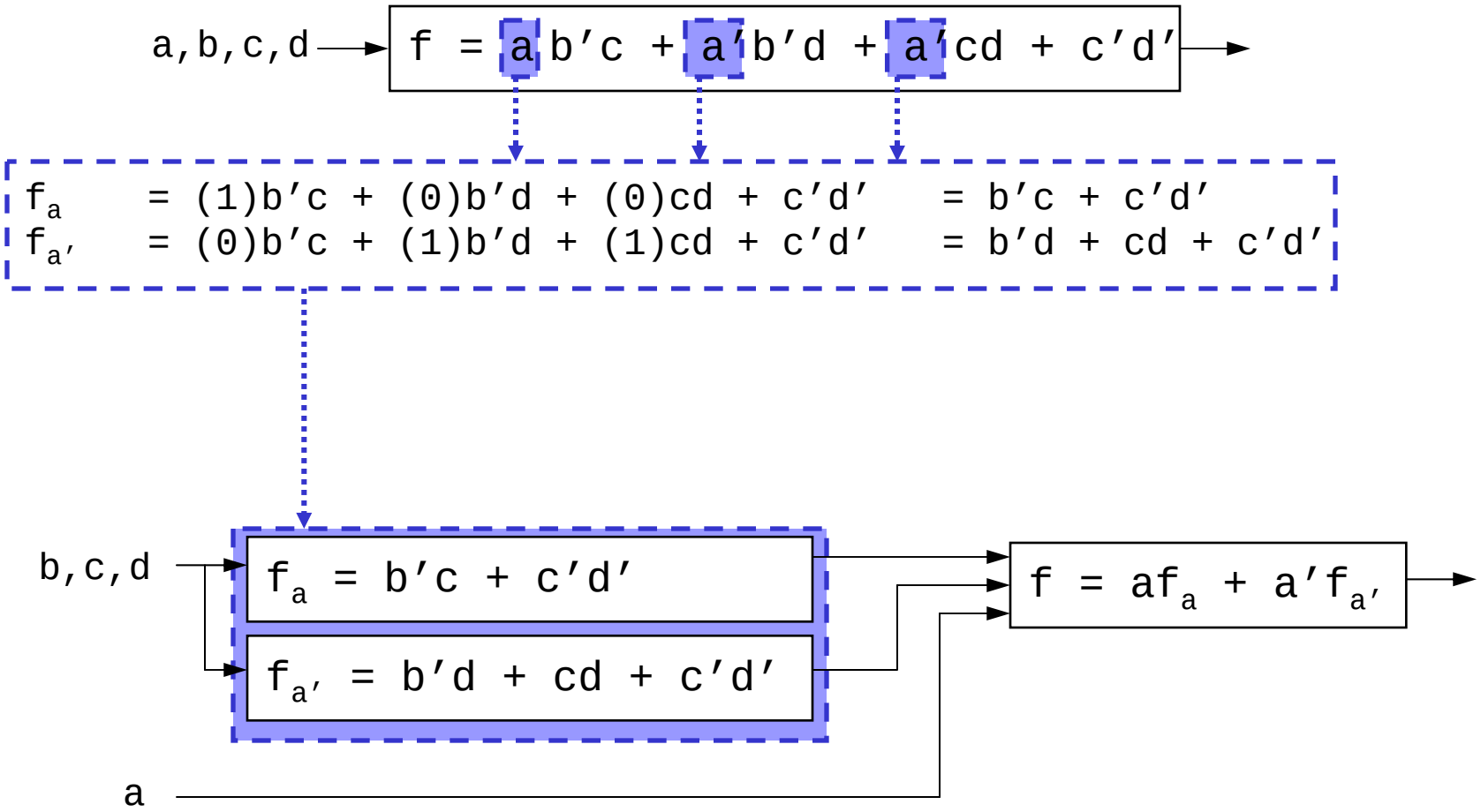


DECOMPOSE [1]

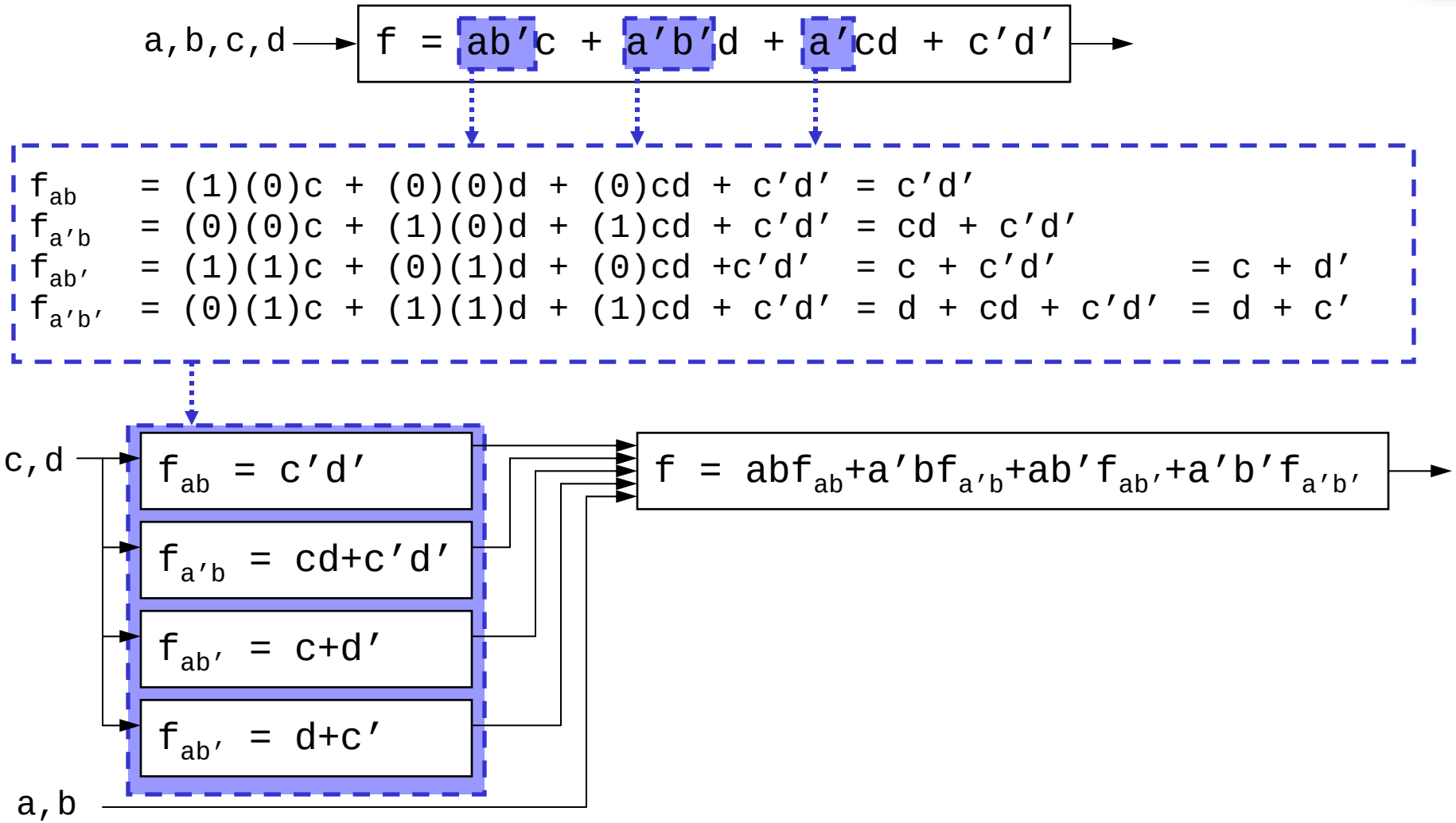
- Riduce le dimensioni di una espressione
  - Rendere più semplice l'operazione di library mapping
  - Aumentare la probabilità di successo della **SUBSTITUTE**
- La decomposizione disgiuntiva semplice può essere applicata ricorsivamente



DECOMPOSE [2]


a,b,c,d
f = ab'c + a'b'd + a'cd + c'd'
f_a = (1)b'c + (0)b'd + (0)cd + c'd' = b'c + c'd'
f_a' = (0)b'c + (1)b'd + (1)cd + c'd' = b'd + cd + c'd'
b,c,d
f_a = b'c + c'd'
f_a' = b'd + cd + c'd'
f = af_a + a'f_a'
a

DECOMPOSE [3]


a,b,c,d
f = ab′c + a′b′d + a′cd + c′d′
f_ab = (1)(0)c + (0)(0)d + (0)cd + c′d′ = c′d′
f_a′b = (0)(0)c + (1)(0)d + (1)cd + c′d′ = cd + c′d′
f_ab′ = (1)(1)c + (0)(1)d + (0)cd +c′d′ = c + c′d′ = c + d′
f_a′b′ = (0)(1)c + (1)(1)d + (1)cd + c′d′ = d + cd + c′d′ = d + c′
c,d
f_ab = c′d′
f_a′b = cd+c′d′
f_ab′ = c+d′
f_ab′ = d+c′
a,b
f = abf_ab+a′bf_a′b+ab′f_ab′+a′b′f_a′b′