
# Sintesi di Reti Combinatorie

## Ottimizzazione di Reti Combinatorie a Due Livelli: Metodo Euristico

Introduzione

Metodo Euristico per reti a due livelli

Approfondimenti

versione del 15/10/04

# Sintesi di reti combinatorie a due livelli: *Metodi euristici*

- La minimizzazione esatta ha due problemi:
  - L'enorme numero di implicanti primi.
    - Può essere dimostrato che il numero degli implicanti primi di una funzione logica di $n$ ingressi può essere maggiore di $3^n/n$.
  - L'intrattabilità del problema di copertura.
    - E' un problema *NP*-completo.
- Soluzione:
  - Miglioramento iterativo della soluzione.
    - Partendo da una condizione iniziale (*specifiche della funzione*) la copertura è modificata per *cancellazione*, *aggiunta* e *modifica* di implicanti fino a che non è raggiunta una condizione di minimalità (quando nessuna delle operazioni porta a successivi miglioramenti).

# Sintesi di reti combinatorie a due livelli: *Metodi euristici*

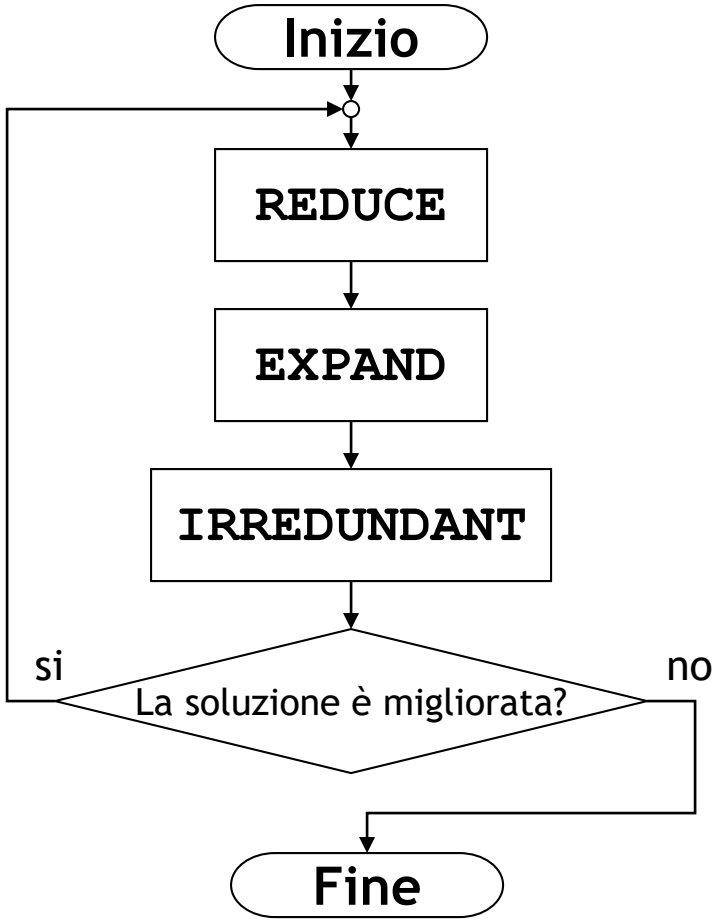
- I metodi euristici di minimizzazione differiscono per *qualità* della soluzione.
  - *Qualità*: Differenza in cardinalità tra la copertura minimale (euristica) e quella minima (ottenuta con metodi esatti).
- Le soluzioni prodotte da *Espresso* (standard per minimizzazione logica a 2 livelli) coincidono spesso con quelle di *Espresso-Exact*, ma in tempi più brevi.
  - Procedura di minimizzazione:
    - *Ingresso*: Lista dei mintermini/implicanti (ON-set) ed il DC-set della funzione.
    - *Condizione iniziale*: La lista degli implicanti rappresenta la copertura iniziale della funzione.
    - *Sviluppo*: La copertura iniziale viene iterativamente manipolata da alcuni operatori.
    - *Termine*: L'operazione si conclude quando nessun operatore migliora la copertura.

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici: operatori*

- Gli operatori utilizzati da Espresso sono:
  - **Expand**
    - espande i cubi rendendoli primi ed eliminando eventuali cubi coperti
    - la copertura ottenuta risulta costituita da soli implicanti primi (*copertura prima*)
  - **Reduce**
    - riduce i cubi per consentire di uscire da minimi locali in cui si è giunti dopo l'espansione
    - la copertura ottenuta dopo una riduzione non è più costituita da soli implicanti primi ma è della stessa cardinalità di quella di partenza.
      - Non è aumentato il numero degli implicanti
      - Per implicanti parzialmente coperti da altri implicanti …
  - **Irredundant**
    - elimina i cubi ridondanti
    - modifica la cardinalità della copertura riducendola

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici*

- Algoritmo:

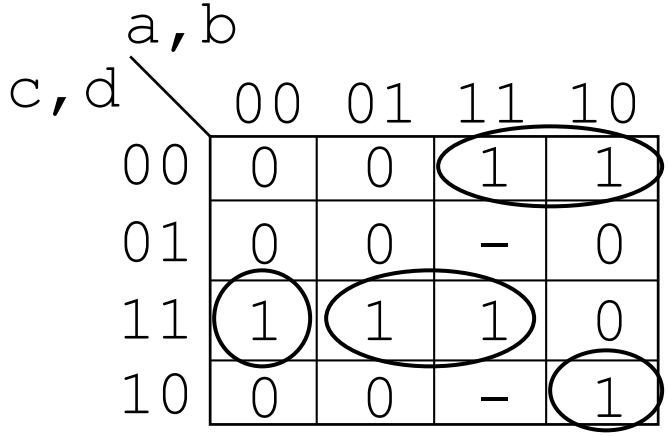


# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici*

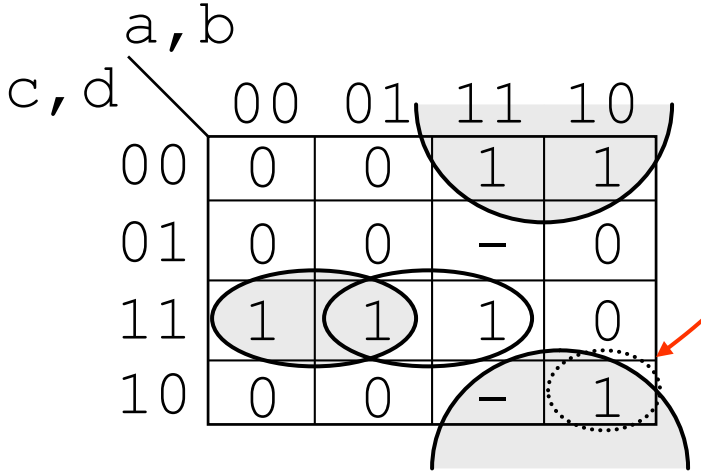
- Esempio:

*Mintermine* eliminato poiché coperto durante l'espansione

**Condizione iniziale:**
deriva direttamente dalle specifiche del problema




Espansione




Cardinalità: 4

Costo in letterali: 14

Nota: copertura non prima. Inoltre, non serve la riduzione poiché non ci sono cubi sovrapposti

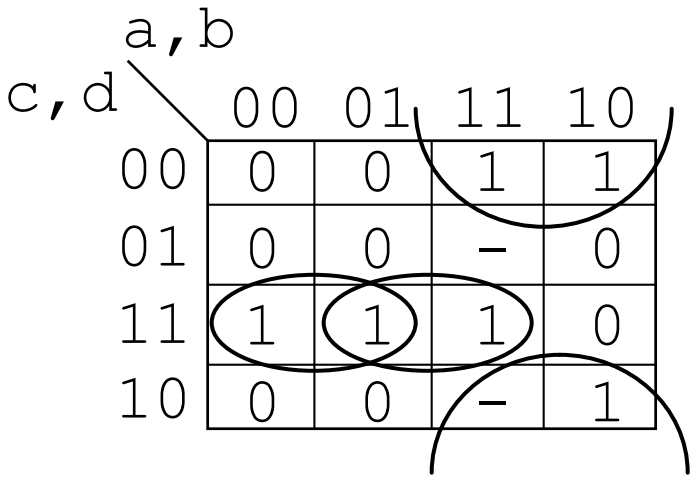
Cardinalità: 3

Costo in letterali: 8

Nota: copertura prima. Inoltre, non esistono implicanti ridondanti

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici*

- Esempio (cont.): la soluzione è migliorata, quindi si prosegue

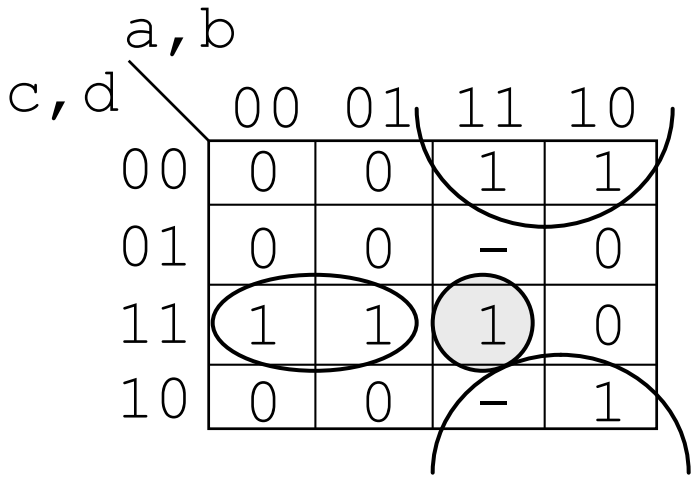



Cardinalità: 3

Costo: 8

Nota: copertura prima

Riduzione



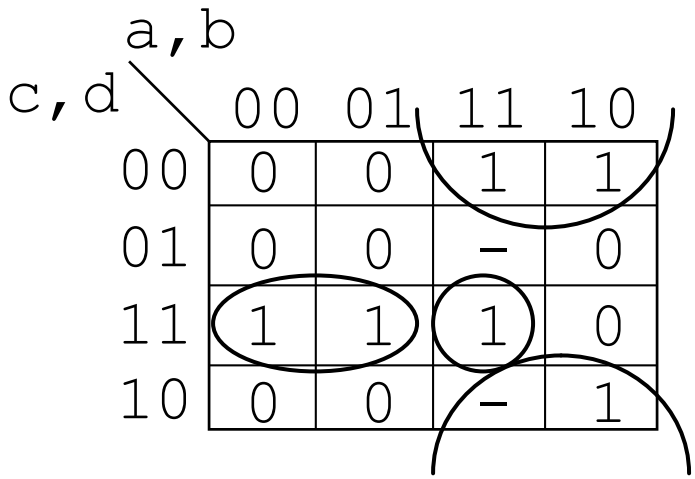

Cardinalità: 3

Costo: 9

Nota: copertura non prima

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici*

- Esempio (cont.):

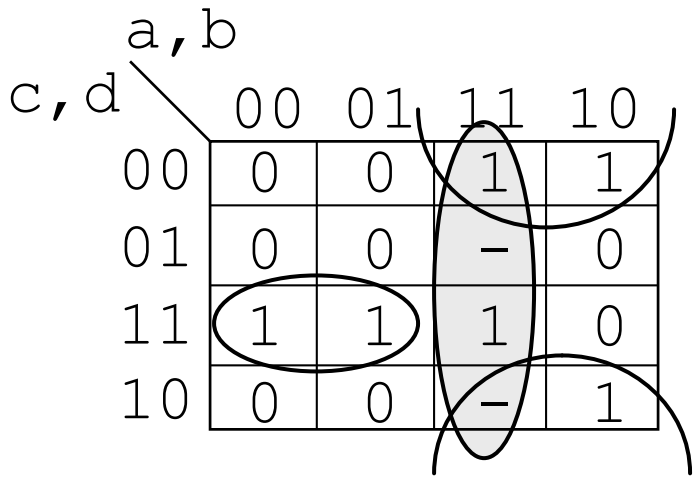



Cardinalità: 3

Costo: 9

Nota: copertura non prima

Espansione




Cardinalità: 3

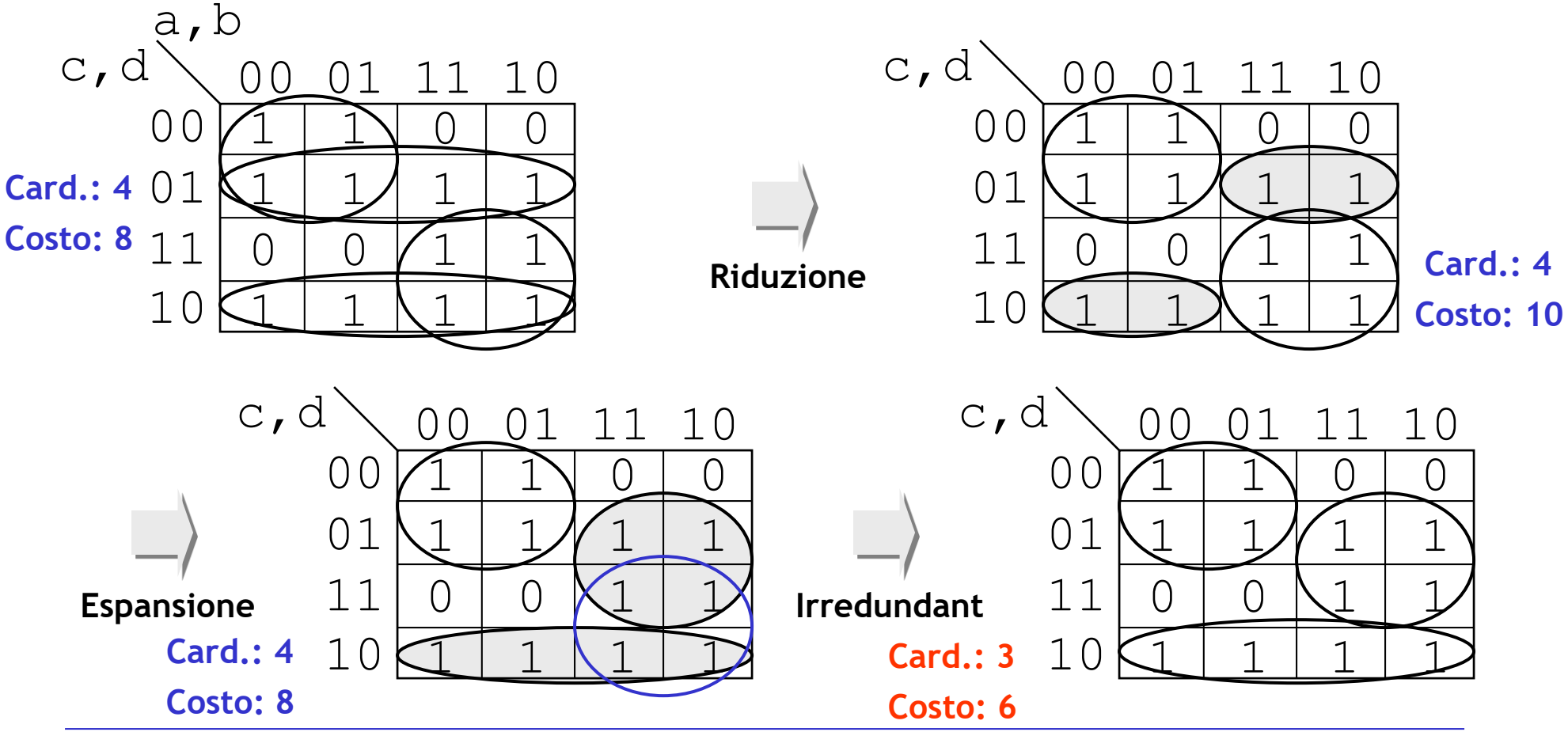
Costo: 7

Nota: copertura prima

Fine della procedura (l'applicazione di una riduzione porta alla soluzione individuata al passo precedente)

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici*

- Esempio 2:




# Sintesi Combinatoria

## Sintesi di reti combinatorie a due livelli
## Metodo Euristico

Approfondimento

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Expand***

- **Expand**
  - Gli implicanti relativi alla copertura sono rielaborati uno alla volta. Ogni implicante è espanso a primo e tutti gli implicanti da esso coperti sono eliminati.
  - La trasformazione `expand()` rende la copertura prima e minimale. L'espansione di un implicante è realizzata aumentando il sotto cubo ad esso associato in una o più direzioni e verificando se l'espanso è ammissibile
  - Verifica dell'ammissibilità dell'espansione:
    - Una espansione è ammissibile se l'implicante ottenuto non interseca l'OFF-set. (E' richiesta la conoscenza dell'OFF-set e questo può essere pesante in termini di memoria utilizzata.)
    - non intersezione: il prodotto logico tra l'implicante ottenuto e ogni mintermine corrispondente all'OFF-set è 0

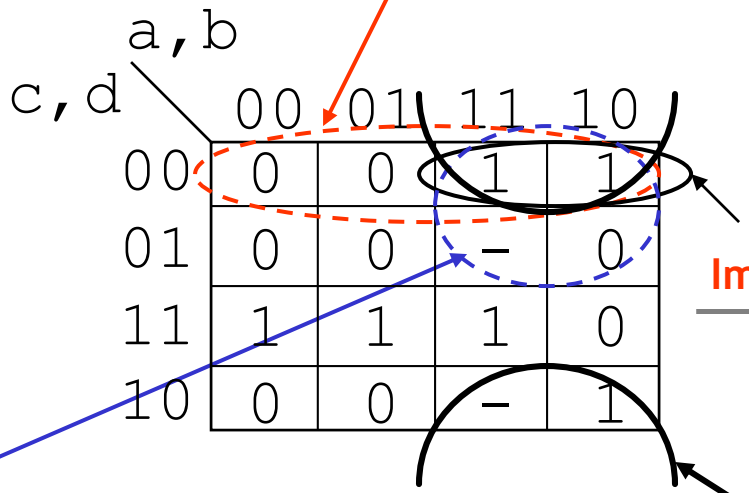
# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Expand**: verifica ammissibilità*

- Esempio:

OFF-Set: a'c'+ab'd+a'd'

**Espansione rispetto ad a** di ac'd' è **c'd'**
*espansione non ammissibile*

Verifica ammissibilità:
OFF-Set*(c'd')= a'c'd'≠ 0
non ammissibile poiché l'intersezione con l'OFF-set non è nulla



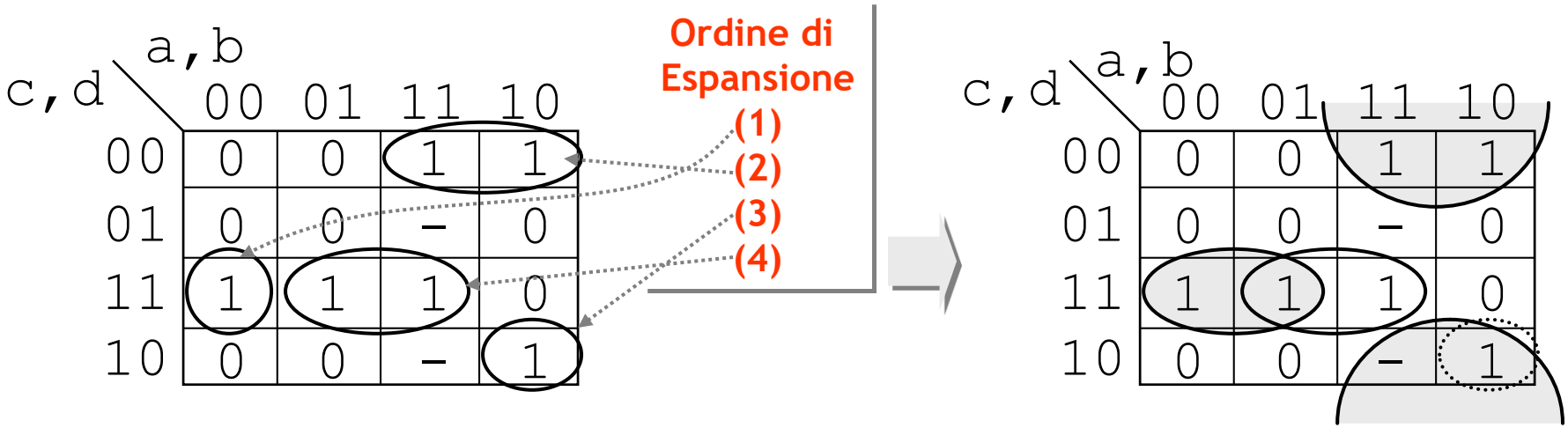

**Implicante da espandere:a c'd'**

**Espansione rispetto a d** di ac'd' è **ac'**
*espansione non ammissibile* (intersezione = ab'c'd)

**Espansione rispetto a c** di ac'd' è **ad'**
***espansione ammissibile*** (intersezione = 0)

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Expand***

- Esempio:




Copertura iniziale:

*on-set*: {ac'd', a'b'cd, bcd, ab'cd'}

*dc-set*: {abc'd, abcd'}

Copertura finale:

*on-set*: {ad', a'cd, bcd}

*dc-set*: {abc'd, abcd'}

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Expand:** ordine di espansione degli implicanti*

- La qualità del risultato dipende da due fattori:
  1. Ordine di espansione del singolo implicante (direzione): si espande in tutte le direzioni possibili seguendo un ordine predeterminato (es. lessico-grafico) e si accetta la soluzione migliore.

  2. Ordine degli implicanti da espandere: si utilizza una euristica che consente di ridurre la probabilità di effettuare delle espansioni inutili
     - Gli implicanti sono ordinati in base alla probabilità sia di essere espansi sia di non essere coperti da altri implicanti;
     - Ad ogni implicante è associato un peso che *misura* la sua *propensione* alla espansione e alla non-copertura da parte di altri implicanti.
     - L'implicante con peso minore è quello che ha più probabilità di essere espanso e non coperto da altri.
     - Per il calcolo del peso si utilizza, per i letterali, il codice *positional-cube notation*:
       - **0** (variabile in forma negata) è codificato con **10**
       - **1** (variabile in forma naturale) è codificato con **01**
       - **–** (variabile non presente) è codificato con **11**
       - 00 non è una codifica ammissibile

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Expand - Esempio***

| | a′a | b′b | c′c | d′d | Peso | |
|---|---|---|---|---|---|---|
| ac′d′ | 0 1 | 1 1 | 1 0 | 1 0 | 3+3+2+1+2=11 | (3) |
| a′b′cd | 1 0 | 1 0 | 0 1 | 0 1 | 2+3+3+2=10 | (1) |
| bcd | 1 1 | 0 1 | 0 1 | 0 1 | 2+3+2+3+2=12 | (4) |
| ab′cd′ | 0 1 | 1 0 | 0 1 | 1 0 | 3+3+3+2=11 | (2) |
| | | | | | | Ordine |
| Conteggio per colonna | **2 3** | **3 2** | **1 3** | **2 2** | | |

- Ogni peso è calcolato come prodotto interno del vettore conteggio per colonna con il vettore relativo all'implicante, espresso in notazione positional-cube.
  - Es: $|01\ 11\ 10\ 10| * |23\ 32\ 13\ 22|^T = 11$

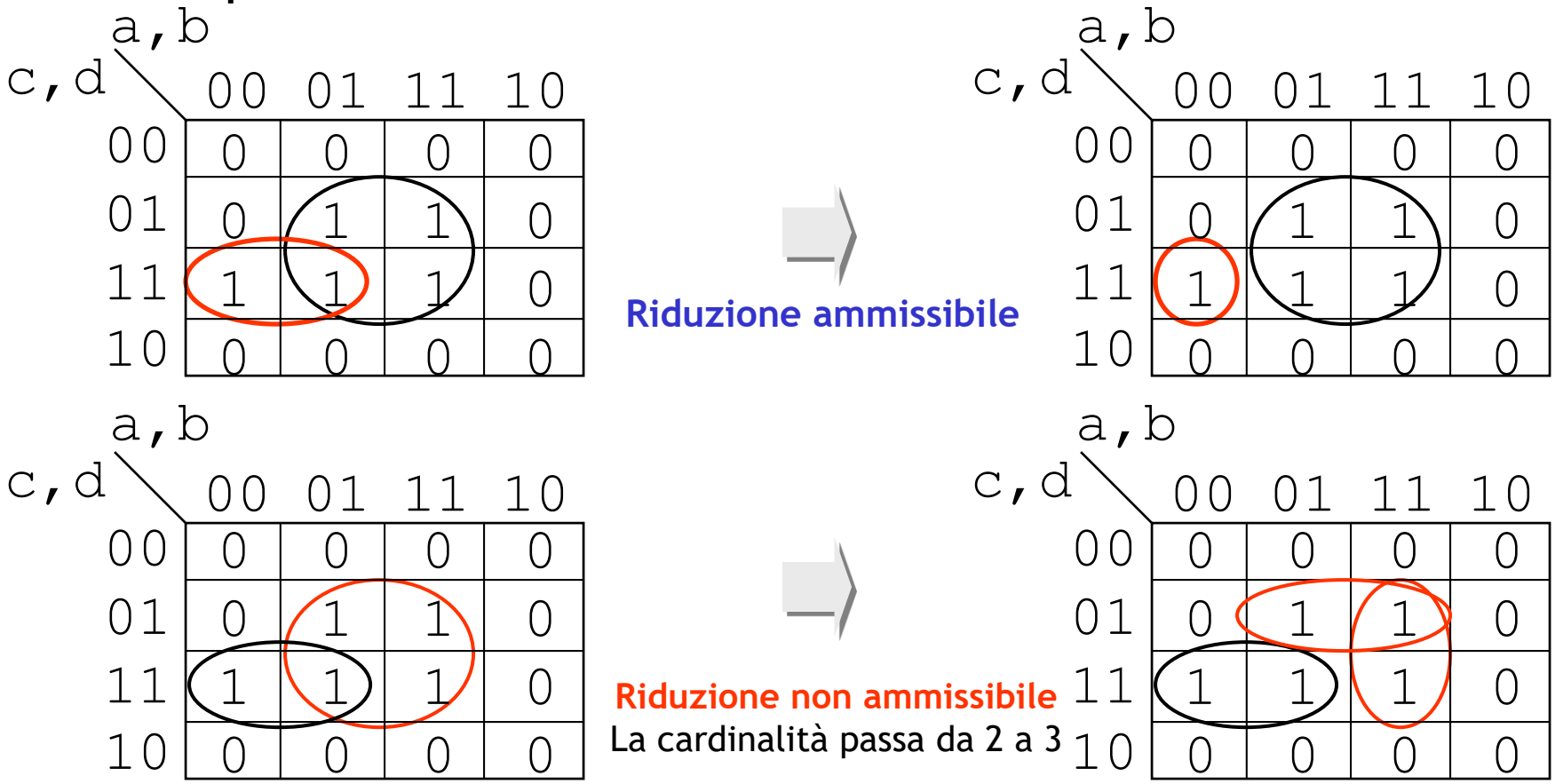
- peso minore ⇒ espansione più probabile
- a parità di peso, più letterali (un cubo piccolo ha più probabilità di essere espanso poiché richiede pochi 1 e - adiacenti)

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Reduce***

- **Reduce**
  - Trasforma la copertura in un'altra non prima della stessa cardinalità.
    - Osservazione: questa trasformazione consente di uscire da minimi locali
  - Gli implicanti sono manipolati uno alla volta.
    - questa operazione può ridurre gli implicanti di dimensione.
    - La trasformazione di un implicante è attuata riducendo il sottocubo ad esso associato in una o più direzioni.
  - Una riduzione è ammissibile se e solo se l'implicante ridotto forma con i rimanenti una copertura per la funzione senza modificarne la cardinalità.

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Reduce***

- Esempio:



# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - **Reduce***

- Per evitare di applicare la trasformazione `reduce()` a implicanti che non possono essere ridotti oppure che portano ad una riduzione non ammissibile si applica una euristica.
- La regola euristica di scelta:
  - Il primo implicante da ridurre è quello con peso maggiore (peso calcolato come in `expand()`).
- Esempio (vedi Expand):

| Implicanti: | a | b | c | d | | | |
|---|---|---|---|---|---|---|---|
| ad' | 01 | 11 | 11 | 10 | ⇒ | 3+2+3+1+3+1=13 | (1) |
| bcd | 11 | 01 | 01 | 01 | ⇒ | 1+3+3+3+2=12 | (3) |
| acd | 01 | 11 | 01 | 01 | ⇒ | 3+2+3+3+2=13 | (2) |
| | | | | | | | Ordine |
| conteggio per colonna: | 13 | 23 | 13 | 12 | | | |

# Irredundant

- Rende la copertura non ridondante.
- Viene scelto un sottoinsieme di implicanti *parzialmente ridondanti* tale per cui ogni implicante non è interamente coperto da un altro dello stesso sottoinsieme
  - la copertura è divisa in tre insiemi
    - *relativamente essenziali*
    - *parzialmente ridondanti*
    - *totalmente ridondanti*
  - Rispetto al metodo esatto, la copertura è costituita da implicanti non tutti necessariamente primi.

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - Espresso*

- (1982) ESPRESSO II è basato sulla applicazione di iterate espansioni e riduzioni. Il risultato prodotto da ESPRESSO è una copertura non ridondante spesso di minima cardinalità.
- I passi seguiti da ESPRESSO II sono:

1. **COMPLEMENT**: Calcola l'OFF-set.
2. **EXPAND**: Espande a primi gli implicanti e rimuove quelli coperti.
3. **ESSENTIAL PRIMES**: Estrae gli implicanti essenziali primi e li unisce al DC-set.
4. **REDUCE**: Riduce ogni implicante a un implicante essenziale minimo.
5. **EXPAND**: Espande a primi gli implicanti e rimuove quelli coperti.
6. **IRREDUNDANT COVER**: Trova la copertura minimale non ridondante. Se la soluzione migliora vai al passo 4.
7. **LASTGASP**: applica per un'ultima volta `REDUCE`, `EXPAND` e `IRREDUNDANT COVER` usando una differente strategia.
8. **COST**: Se questa operazione ha successo vai al passo 4.
9. **MAKESPARSE**: Adatta la soluzione ad una PLA.

# Sintesi di reti combinatorie a due livelli:
## *Metodi euristici - Espresso*

Espresso(on_set,dc_set)

```
off_set=complement(on_set U dc_set)
on_set=expand(on_set, off_set)  /*copertura prima ridondante*/
on_set=irredundant(on_set, dc_set)
essential_set=essentials(on_set, dc_set)
on_set=on_set - essential_set  /* toglie 1 dall'on_set  */
dc_set=dc_set U  essential_set /* e li aggiunge al dc_set */
repeat
        f2=Cost(on_set)
        repeat
                f1=|on_set|
                on_set=reduce(on_set,dc_set)
                on_set=expand(on_set, off_set)
                on_set=irredundant(on_set,dc_set)
        until (|on_set|>= f1)
        on_set=last_gasp(on_set,dc_set,off_set)
until (cost(on_set) >= f2)
on_set=on_set U essential_set
dc_set=dc_set - essential_set
on_set=make_sparse(on_set,dc_set,off_set)
```

*Quando vera termino ciclo*

*Quando vera termino ciclo*

# Sintesi di reti combinatorie a due livelli: *Espresso*

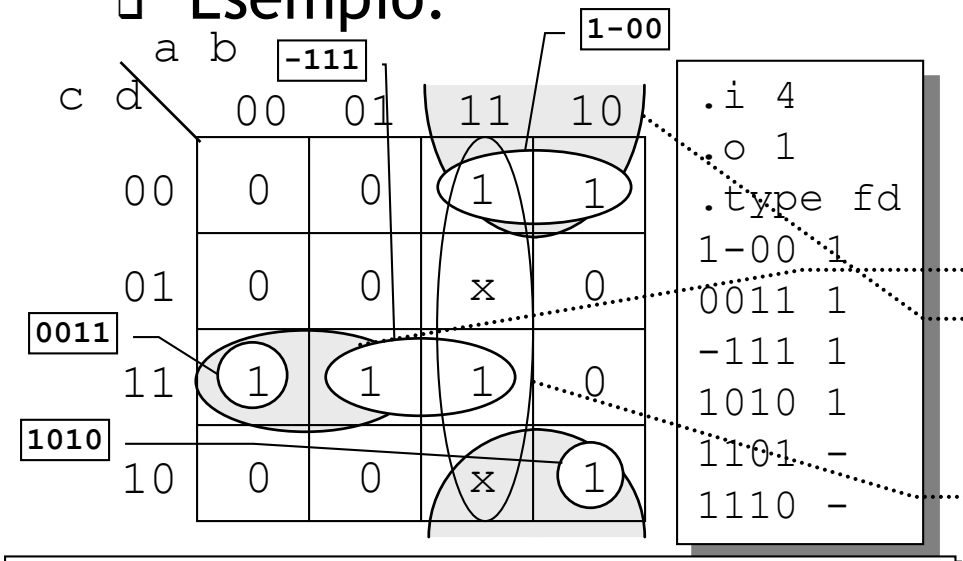
- Comando:
  - espresso [parametri] [file]
- Funzione:
  - minimizzazione di funzioni logiche a due livelli.
- Parametri:
  - -d: debugging
  - -e[opzioni]: seleziona le opzioni di espresso:
    - fast, ness, nirr, nunwrap, onset, pos, strong, eat, eatdots, kiss, random
  - -o[tipo]: seleziona il formato di uscita:
    - f, fd, fr, fdr, pleasure, eqntott, kiss, cons
  - -s: fornisce un breve sommario relativo all'esecuzione;
  - -t: fornisce un ampio sommario relativo all'esecuzione;
  - -x: non visualizza la soluzione;

# Sintesi di reti combinatorie a due livelli: *Espresso*

- **Parametri (continua):**
  - -v[tipo]: messaggi di dettaglio (-v ‘’ per un accurato dettaglio)
  - -D[comando]: esegue il sotto-comando:
    - ESPRESSO, many, exact, qm, single_output, so, so_both, simplify, echo, opo, opoall, pair, pairall, check, stats, verify, PLAverify, equiv, map, mapdc, fsm, contain, d1merge, d1merge_in, disjoint, dsharp, intersect, minterms, primes, separate, sharp, union, xor, essen, expand, gasp, irred, make_sparse, reduce, taut, super_gasp, lexsort, test
  - -Sn: seleziona la strategia per il sotto comando (solo quelli riportati):
    - opo: bit2=esatto, bit1=ripetuto bit0=salta sparse
    - opoall: 0=minimizza, 1=esatto
    - pair: 0=algebrico, 1=strongd, 2=espresso, 3=esatto
    - pairall: 0=minimizza, 1=esatto, 2=opo
    - so_espresso: 0=minimize, 1=exact
    - so_both: 0=minimize, 1=exact

# Sintesi di reti combinatorie a due livelli: *Espresso*

- Esempio:




```
.i 4
.o 1
.type fd
1-00 1
0011 1
-111 1
1010 1
1101 -
1110 -
```

| Cubi | Notazione positional-cube | Pesi | Ordine |
|---|---|---|---|
| 1-00 | 1 0 1 1 0 1 0 1 | 11 | **3** |
| 0011 | 0 1 0 1 1 0 1 0 | 10 | **1** |
| -111 | 1 1 1 0 1 0 1 0 | 12 | **4** |
| 1010 | 1 0 0 1 1 0 0 1 | 11 | **2** |
| | **2 3 3 2 1 3 2 2** | | |

```
ipeca4>espresso -v " ex3.pla
EXPAND: 0011 1 (covered 0)
EXPAND: 1-00 1 (covered 1)
EXPAND: -111 1 (covered 0)
# IRRED: F=3 E=3 R=0 Rt=0 Rp=0 Rc=0 Final=3 Bound=0
ESSENTIAL: 0-11 1
ESSENTIAL: 1--0 1
REDUCE: -111 1 to 1111 1 0.00 sec
EXPAND: 1111 1 (covered 0)
# IRRED: F=1 E=1 R=0 Rt=0 Rp=0 Rc=0 Final=1 Bound=0
REDUCE_GASP: 11-- 1 reduced to 1111 1
# IRRED: F=3 E=3 R=0 Rt=0 Rp=0 Rc=0 Final=3 Bound=0
.i 4
.o 1
.p 3
11-- 1
0-11 1
1--0 1
.e
ipeca4>
```

# Sintesi di reti combinatorie a due livelli: *Espresso*

- Esempio:

```
.i 4
.o 3
.type fr
00-1 1-1
01-- 001
1000 0--
1001 --1
1011 -1-
1100 111
1101 000
1110 -10
1111 01-
```

```
EXPAND: 1100 100 (covered 2)
EXPAND: 1011 010 (covered 3)
EXPAND: 1111 010 (covered 0)
EXPAND: 1100 001 (covered 0)
EXPAND: 01-- 001 (covered 0)
# IRRED: F=5 E=5 R=0 Rt=0 Rp=0 Rc=0 Final=5 Bound=0
ESSENTIAL: 0--- 001
REDUCE: -0-1 111 to -0-1 101 0.00 sec
REDUCE: 1-1- 010 to 1-11 010 0.00 sec
REDUCE: 1-00 011 to 1100 001 0.00 sec
EXPAND: 1100 001 (covered 0)
EXPAND: 1-11 010 (covered 0)
EXPAND: -0-1 101 (covered 0)
# IRRED: F=4 E=4 R=0 Rt=0 Rp=0 Rc=0 Final=4 Bound=0
REDUCE_GASP: 1-00 011 reduced to 1100 001
REDUCE_GASP: 1-1- 010 reduced to 1111 010
REDUCE_GASP: 11-0 110 reduced to 1100 100
REDUCE_GASP: -0-1 111 reduced to -0-1 101
EXPAND: 1100 111 (covered 0)
# IRRED: F=5 E=2 R=3 Rt=0 Rp=3 Rc=1 Final=3 Bound=0
REDUCE: -0-1 111 to -0-1 101 0.01 sec
EXPAND: -0-1 101 (covered 0)
# IRRED: F=3 E=3 R=0 Rt=0 Rp=0 Rc=0 Final=3 Bound=0
… (continua)
```

```
… (continua)
# IRRED: F=3 E=3 R=0 Rt=0 Rp=0 Rc=0 Final=3 Bound=0
REDUCE_GASP: 1-1- 010 reduced to 111- 010
REDUCE_GASP: 1100 111 reduced to 1100 111
REDUCE_GASP: -0-1 111 reduced to -0-1 101
# IRRED: F=2 E=2 R=0 Rt=0 Rp=0 Rc=0 Final=2 Bound=0
# IRRED: F=3 E=2 R=1 Rt=1 Rp=0 Rc=0 Final=2 Bound=0
# IRRED: F=3 E=3 R=0 Rt=0 Rp=0 Rc=0 Final=3 Bound=0
EXPAND: -0-1 101 (covered 0)
.i 4
.o 3
.p 4
1100 111
1-1- 010
-0-1 101
0--- 001
.e
```