Chương 3: Tổng hợp mạch 2 mức

- Tối giản đại số Bool mạch 2 mức
- Phương pháp Quine-McCluskey
- Luật suy diễn 2 mức (Two-Level Tautology)
- Bù (complementation)
- Phương pháp tối giản chính xác (Exact Minization)
- Phương pháp tối giản tối ưu (Heuristic Minimization)

Tối giản đại số Bool mạch 2 mức

- **Two-level Boolean minimization** được sử dụng để tìm dạng tổng các tích (sum-of-products) cho hàm số Bool đó là tối ưu theo một hàm chi phí cho trước.
- 2 bước trong quá trình tối giản là:
  - Sinh tập prime implicant
  - Chọn tập một tập prime implicant tối thiểu

Phương pháp Quine-McCluskey

- Các bước thực hiện
  - Sinh Prime Implicant
  - Thành lập bảng Prime Implicant
  - Tìm Essential Prime Implicant
  - Tìm Dominated Column
  - Tìm Dominating Row
  - Tối ưu các Dominated Comlumn và các dominating Row.
  - Tóm tắt quy trình

Sinh Prime Implicant

- Liệt kê tập các **minterm** của hàm (biểu diễn ở dạng các số nhị phân).
- Trộn các cặp **minterm**. Các cặp **minterm** chỉ khác nhau 1 bit để tạo thành 1 **cube**.
- Tiếp tục trộn các cặp **cube** thành một **cube** mới cho đến khi không thể trộn tiếp.
- Đánh dấu các **cube** đã được trộn
- Tập các **cube** không được đánh dấu là tập **prime implicant** của hàm.
- Trong trường hợp hàm không đầy đủ thì các **minterm** sẽ bao gồm trong cả tập **ON-set** và **DC-set** của hàm đó.

0	0000	0, 8	_000
5	0101	5, 7	01_1
7	0111	7, 15	_111
8	1000	8, 9	100_✶
9	1001	8, 10	10_0✶
10	1010	9, 11	10_1✶
11	1011	10, 11	101_✶
14	1110	10, 14	1_10✶
15	1111	11, 15	1_11✶
		14, 15	111_✶

E, D, C

8, 9, 10, 11 → 10_ _ → B

10, 11, 14, 15 → 1_1_ → A

Bảng Prime Implicant

- Các hàng của bảng là các **minterm** (ở dạng nhị phân)
- Các cột là các prime implicant
- Đánh dấu "**x**" ở những nơi giao nhau giữa hàng và cột nếu các **prime implicant** chứa các **minterm**

	A	B	C	D	E
0000					X
0101				X	
0111			X	X	
1000		X			X
1001		X			
1010	X	X			
1011	X	X			
1110	X				
1111	X		X		

Essential Prime Implicant

- Cột mà có hàng chỉ chứa 1 dấu "x" là **essential prime implicant**.
- Ví dụ: E là **essential prime implicant**
- **Essentical prime implicant** phải có mặt trong biểu thức tối giản
- Các **prime** được chọn: A, B, D, E

	A	B	C	D	E
0000					X
0101				X	
0111			X	X	
1000		X			X
1001		X			
1010	X	X			
1011	X	X			
1110	X				
1111	X		X		

Dominated Columns

- Nếu **Không** có Essential Prime implicant.
- Chọn một cột: giả sử là A
- Rút gọn A và các hàng được chứa bởi A (hàng 0 và 1)

	A	B	C	D	E	F	G	H
0000	X							X
0001	X	X						
0101		X	X					
0111			X	X				
1000							X	X
1010						X	X	
1110					X	X		
1111				X	X			

Dominated Columns

- Một cột U được gọi là **dominate** cột V nếu cột U chứa tất cả các hàng có trong V.
- Ví dụ:
  - Cột B **dominated** (C)
  - Cột H **dominated** (G)
- Chúng ta có thể bỏ đi các cột **dominated** mà không ảnh hưởng đến kết quả rút gọn

	B	C	D	E	F	G	H
0101	X	X					
0111		X	X				
1000						X	X
1010					X	X	
1110				X	X		
1111			X	X			

	C	D	E	F	G
0101	X				
0111	X	X			
1000					X
1010				X	X
1110			X	X	
1111		X	X		

Dominated Columns

- Chọn C và G, do đây là cặp **Relatively Essential Prime Implicant**
- Chọn E
- Kết quả f = {A, C, E, G}
- Vì việc chọn ngẫu nhiên A ban đầu nên kết quả không đán tin cậy.
- Cần phải thực hiện việc **backtrack** và bỏ A khỏi bảng, tiếp tục thực hiện
- Một kết quả khác f = {B, D, F, H}

	C	D	E	F	G
0101	X				
0111	X	X			
1000					X
1010				X	X
1110			X	X	
1111		X	X		

Dominating Rows

- Hàng i của bảng **prime implicant** được gọi là **dominate** một hàng j khác nếu i có x tại tất cả những cột mà trong đó j được đánh dấu x
- Ta có thể lượt bỏ những cột **dominating**
- Ví dụ:
  - Hàng 0111 **dominate** hàng 0101
  - Hàng 1010 **dominate** hàng 1000

	C	D	E	F	G
0101	X				
0111	X	X			
1000					X
1010				X	X
1110			X	X	
1111		X	X		

	C	D	E	F	G
0101	X				
1000					X
1110			X	X	
1111		X	X		

Tóm tắt

- Các bước thực hiện
  - Xây dựng bảng prime implicant
  - Xóa những cột dominated và những hàng dominating
  - Tìm hết những essential prime. Đưa chúng vào tập lựa chọn
  - Nếu tập lựa chọn các prime này lớn hơn hoặc bằng lời giải tốt nhất thì dừng lại.
  - Chọn một prime một cách tối ưu
  - Thêm prime vào tập lựa chọn và quay trở lại với bảng đã được tối giản từ việc chọn prime.
  - Loại prime đó ra khỏi tập lựa chọn và quay trở lại với bảng đã được tối giản này để thực hiện lại quy trình.

Luật suy diễn 2 mức - Tautology

- Hàm f là **tautology** nếu và chỉ nếu tập **ON-set** của f là tập vũ trụ. Ký hiệu $f \cong 1$
- $f(x) = 1, \forall x \in B^n \Leftrightarrow f \cong 1$
- Hàm f được gọi là **monotone increasing** (decreasing) theo biến x_i nếu việc chuyển x_i từ 0 lên 1 làm cho f chuyển từ 0 lên 1 hoặc giữ nguyên giá trị (1 xuống 0 hoặc giữ nguyên)

Hàm Unate

- Hàm f là **unate** theo x_j nếu nó **monotone increasing** hoặc **monotone decreasing** theo x_j
- Ví dụ:
  - $f = x_1 \cdot \overline{x_2} + \overline{x_2} \cdot x_3$ **monotone increasing** theo x_1 và x_3 và **monotone decreasing** theo x_2 → f **unate**
  - $f = x_1 \cdot \overline{x_2} + \overline{x_1} \cdot x_2$, f là **nonunate** (binate) theo x_1 và x_2

Hàm Unate

- Cho một **cover** của một hàm f nếu một biến x_j là _ hoặc 1 (_ hoặc 0) trong tất cả các **cube** của **cover** đó thì f **unate** theo x_j
- Ví dụ:
  - $f = x_1 \cdot \overline{x_2} + \overline{x_2} \cdot x_3$, f **unate** với tất cả các x_1, x_2, x_3 do **cover** C = {10_, _01}
  - $f = x_1 \cdot x_2 \cdot x_3 + x_2$, có **cover** C = {110, _0_} có x_2 là 1 và 0, vì vậy mà f không **unate** theo x_2
- Kiểm tra tính **unate** của một hàm thông qua tập **cover** là một cách kiểm tra khá dễ dàng.
- Một **unate cover** nếu f **unate** trên tất cả các biến.

Hàm Unate

- Định lý: Một **unate cover** là một **tautology** nếu và chỉ nếu **cover** đó chứa một hàng toàn x (hay _)
- Định lý: Cho một **cover** C của hàm f. Giả sử các hàng và cột của f được sắp xếp như sau:

$$C = \begin{pmatrix} A & T \\ T & D \end{pmatrix}$$

Với T là một khối tất cả x (hay _). Giả sử A không **tautology** thì f **tautology** khi và chỉ khi D là **tautology**

Hàm Unate

- Hệ quả: Lấy U là tập các cột **unate** trong một **cover** C. Sắp xếp C các cột U trước và các cột B sau:

C = (U B)

Nếu U không có hàng nào chứa toàn x (hay _) thì f không **tautology**

Tautology

- Quy trình xác định tính **tautology** của hàm f:

```
Tautology(F){
  T = SPECIAL_CASE(F);
  if(T ≅ -1)  (T, F) = UNATE_REDUCTION(F);
  if(T ≅ -1)  (T, F) = COMP_REDUCTION(F);
  if(T ≅ -1) return(T);
  j = BINATE_INPUT_SELECT(F);
  if(TAUTOLOGY(F_xj) ≅ 0) return(T=0);
  if(TAUTOLOGY(F_x'j) ≅ 0) return(T=0);
  Return (T = 1);
}
```


Tautology

- Định lý: f có ma trận như hình sau:

$$M_f = \begin{pmatrix} A_1 & D & \dots & D \\ D & A_2 & \dots & D \\ \dots & \dots & \dots & \dots \\ D & D & \dots & A_P \end{pmatrix}$$

Với D là một khối toàn x. M_f là một **tautology** khi và chỉ khi có k, với $1 \le k \le P$, mà $A_k \cong 1$

Bù

- **Bù** của hàm f được ký hiệu là: $\overline{f}$

$$\overline{f} = x_j \overline{f_{x_j}} + \overline{x_j}\,\overline{f_{\overline{x_j}}}$$

Nếu F là **monotone increasing** theo x_j thì:

$$\overline{f} = \overline{x_j}\,\overline{f_{\overline{x_j}}} + \overline{f_{x_j}}$$

Nếu F là **monotone decreasing** theo x_j thì:

$$\overline{f} = x_j \overline{f_{x_j}} + \overline{f_{\overline{x_j}}}$$

Bù

- Quy trình tính bù:

```
COMPLEMENT(F){
  if (row of all 2's) return(ϕ);
  if (F is a unate cover in all variables)
    return(UNATE_COMPLEMENT(F));
  c = first cube in F;
  for(i=1; i≤ N; i=i+1){
    if(c[i] not equal to d[i] for any cube d∈F)  c[i]=2;
  }
  R = UNATE_COMPLEMENT(c);
  F = F_c;
  j = BINATE_INPUT_SELECT(F);
  R = R ∪ (x_j. COMPLEMENT(F_xj));
  R = R ∪ (x'_j. COMPLEMENT(F_x'j));
  Return(R);
}
```

Ví dụ

- Cho hàm f với tập **cover**
- X_2 chứa toàn 1 vì vậy mà R = _0_ _ _
- F_{x2} được thu gọn từ F

F

X_1	X_2	X_3	X_4	X_5
_	1	1	1	1
_	1	1	1	0
_	1	0	1	_
_	1	0	_	1

F_{X2}

X_1	X_3	X_4	X_5
_	1	1	1
_	1	1	0
_	0	1	_
_	0	_	1

Ví dụ

- Chọn biến **binate** x_3. Ta có $F_{X2.X3}$ và $F_{X2.X'3}$ từ F_{X2}
- Chọn biến **binate** x_5. Ta có $F_{X2.X3X5}$ và $F_{X2.X3X'5}$ từ F_{X2X3}
- Suy ra F'_{X2X3}

F_{X2X3}

X_1	X_4	X_5
_	1	1
_	1	0

$F_{X2X'3}$

X_1	X_4	X_5
_	1	_
_	_	1

F_{X2X3X5}

X_1	X_4
_	1

$F_{X2X3X'5}$

X_1	X_4
_	1

F'_{X2X3}

X_1	X_4	X_5
_	0	1
_	0	0

Ví dụ

- $F'_{X2X'3}$ (Sử dụng công thức **unate complement**)
- Suy ra F'_{X2}
- Suy ra F' bằng cách trộn F'_{X2} và X'_2

$F'_{X2X'3}$

X_1	X_4	X_5
_	0	0

F_{X2}

X_1	X_3	X_4	X_5
_	1	0	1
_	1	0	0
_	0	0	0

F

X_1	X_2	X_3	X_4	X_5
_	0	_	_	_
_	_	1	0	1
_	_	1	0	0
_	_	0	0	0

Recursive Complement Operation

Theorem: $\overline{f} = x \cdot \overline{f}_x + \overline{x} \cdot \overline{f}_{\overline{x}}$

Proof:

$$g = x \cdot \overline{f}_x + \overline{x} \cdot \overline{f}_{\overline{x}}$$
$$f = x \cdot f_x + \overline{x} \cdot f_{\overline{x}}$$

$$\left.\begin{aligned} f \cdot g &= 0 \\ f + g &= 1 \end{aligned}\right\} \Rightarrow g = \overline{f}$$

COMPLEMENT Operation

```
Algorithm COMPLEMENT(List_of_Cubes C) {
  if(C contains single cube c) {
    C_res = complement_cube(c)  // generate one cube per
    return C_res                // literal l in c with ^l
  }
  else {
    x_i = SELECT_VARIABLE(C)
    C_0 = COMPLEMENT(COFACTOR(C,^x_i)) Ù ^x_i
    C_1 = COMPLEMENT(COFACTOR(C,x_i))  Ù x_i
    return OR(C_0,C_1)
  }
}
```

Recursive Complement – termination rules

- The cover F is void: $\overline{F} \equiv 1$.
- F has a row of all 2's (dc's): $F \equiv 1$

  $\overline{F} = 0$
- F has single row with one 0 or 1 at x:

  $\overline{F} = x$ or $\overline{x}$
- There is a column x of all 1's: $F = xF_x$

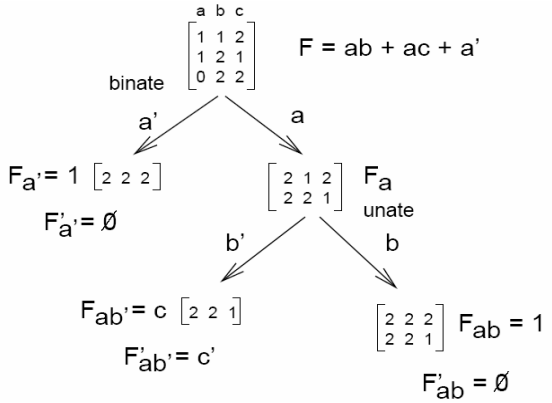
  $\overline{F} = \overline{x} + \overline{F_x}$
- There is a column x of all 0's: $F = \overline{x}F_{\overline{x}}$,

  $\overline{F} = x + \overline{F_{\overline{x}}}$

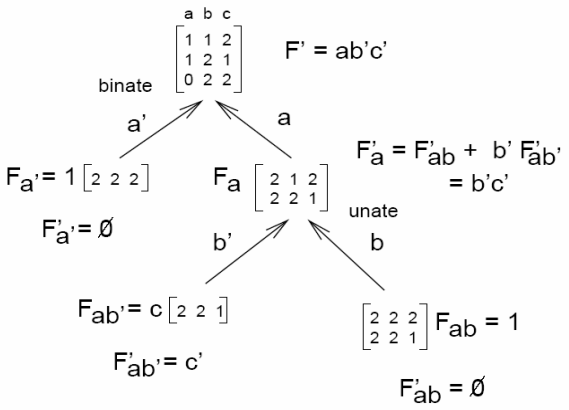
If cover is positive (negative) unate at x, use

$\overline{F} = \overline{F_x} + \overline{x}\,\overline{F_{\overline{x}}}$ (or $\overline{F} = x\overline{F_x} + \overline{F_{\overline{x}}}$)

Recursive Complement – example (split)



Recursive Complement – example (merge)



Phương pháp tối giản chính xác

- Các hạn chế của phương pháp Quine-McCluskey:
  - Việc sinh các prime bắt nguồn từ một tập tường tận các minterm trong tập ON-set và DC-set của hàm
  - Các minterm được kiểm tra theo từng cặp, hầu hết là không thể thực hiện được
  - Phương pháp covering không sử dụng bất kỳ một bounding strategy nào.
- ESPRESSO-EXACT sinh ra tất cả các prime implicant của hàm (ở dạng sum-of-product).

Sinh Prime Implicant

- Mở rộng (Expanding) một cube nghĩa là
  - Đưa các literals 0 hoặc 1 ở phần ngõ nhập của một cube lên _, hoặc
  - Đừa literal 0 trong phần ngõ xuất của một cube lên 1.
- C^0_j là một biến Bool mà biến j của cube c được đưa từ 0 lên _
- C^1_j là một biến Bool mà biến j của cube C được đưa từ 1 lên _

Sinh Prime Implicant

- Cặp biến $(r^i)^0_j$ và $(r^i)^1_j$
  - Có giá trị là 0 và 1 nếu biến j trong cube r^i là 0
  - Có giá trị là 1 và 0 nếu biến j trong cube r^i là 1
  - Có giá trị là 1 và 1 nếu biến j trong cube r^i là _
- Với một biến x_j, cube r^i và mở rộng của c được là disjoint theo x_j Khi:

$$\overline{G_{ij}} = (r^i)^0_j c^0_j + (r^i)^1_j c^1_j = 0$$

$$G_{ij} = \left(\overline{(r^i)^0_j} + \overline{c^0_j}\right)\left(\overline{(r^i)^1_j} + \overline{c^1_j}\right) = 1$$

Sinh Prime Implicant

- Ri và c là disjoint nếu nó disjoint ở một xj nào đó:

$$H_i = \bigcup_{j=1}^{N} G_{ij} = 1$$

- Mở rộng của c là disjoint từ R chỉ khi nó disjoint với tất cả các cube ri R:

$$I = \bigcap_{i=1}^{|R|} H_i = 1$$

- Suy ra:

$$I = \bigcap_{i=1}^{|R|} \bigcup_{j=1}^{N} \left(\overline{\left(r^i\right)^0_j} + \overline{c^0_j}\right)\left(\overline{\left(r^i\right)^1_j} + \overline{c^1_j}\right) = 1$$

Ví dụ

- Cho hàm f gồm các implicant
- Tập OFF-Set R của f là R={001_, 00_1, 01_0, 110_}

0	0000	$\overline{c_1^1}+\overline{c_2^1}+\overline{c_3^0}$	$\overline{c_1^1}+\overline{c_2^1}+\overline{c_3^0}\,\overline{c_4^0}$
5	0101		
7	0111	$\overline{c_1^1}+\overline{c_2^1}+\overline{c_4^0}$	
8	1000		
9	1001	$\overline{c_1^1}+\overline{c_2^0}+\overline{c_4^1}$	$\overline{c_1^0}\,\overline{c_4^1}+\overline{c_1^1}\,\overline{c_3^1}+\overline{c_2^0}+\overline{c_3^1}\,\overline{c_4^1}$
10	1010		
11	1011		
14	1110	$\overline{c_1^0}+\overline{c_2^0}+\overline{c_3^1}$	
15	1111		

$$\overline{c_1^1}\,\overline{c_3^1}+\overline{c_1^1}\,\overline{c_2^0}+\overline{c_1^0}\,\overline{c_2^1}\,\overline{c_4^1}+\overline{c_2^1}\,\overline{c_3^1}\,\overline{c_4^1}+\overline{c_2^0}\,\overline{c_3^0}\,\overline{c_4^0}$$

$$\Rightarrow \{1-1-,10--,01-1,-111,-000\}$$

So sánh kết quả với phương pháp trước (slide 4)

Rút gọn bảng Prime Implicant

- Biểu diễn hàm f dưới dạng các tập:
  - ON-set: F
  - DC-set: D
  - OFF-set: R
- Gọi Q: tập các prime implicant cho hàm f cần phải tối thiểu hoá.
- Chia Q thành 2 tập:
  - Relatively essential: E_r
  - Relatively redundant: R_r

Rút gọn bảng Prime Implicant

- Với c là một cube thuộc F
  - c ∈ E_r nếu Q ∪ D – c không chứa c.
  - c ∈ R_r nếu Q ∪ D – c chứa c.
- Er sẽ nằm trong mọi cover của hàm f
- Rr sẽ được chia thành 2 tập
  - R_t: tập dư thừa toàn phần (totally)
  - R_p: tập dư thừa bán phần (partially)
- Với c là một cube thuộc Rr
  - c ∈ R_t nếu E_r ∪ D chứa c.
  - c ∈ R_p nếu E_r ∪ D không chứa c.

Rút gọn bảng Prime Implicant

- R_t :có thể loại bỏ hoàn toàn
- R_p: một phần sẽ thuộc cover tối thiểu của f
- Xét H = E_r ∪ R_p – c, với c ∈ R_p

Rút gọn bảng Prime Implicant

- Ví dụ

  Q = {0_1_, _01_, 01_ _, 10_ _, _101, 1_01}

  → E_r = {01_ _, 10_ _ }

  R_p = {0_1_, _01_, _101, 1_01}

- Gán c = 0_1_ ∈ R_p

  H = $E_r \cup R_p$ – c = {01_ _, 10_ _, _01_, _101, 1_01}

  Ta tính H_c bằng cách tính $H_{x1' x3}$

  $H_{x1'}$ = {1_ _, 01_, 101}

→ $H_c = H_{x1' x3}$ = {1_, 0_ } (là một tautology trên x_2 và x_4)

Các cube tương ứng là 01_ _ ∈ E_r và _01_ ∈ R_p

Rút gọn bảng Prime Implicant

- x_2 là một biến binate và các Hc tương ứng với x_2 và x_2'
- cube 1_ ứng với 01_ _ ∈ E_r → bỏ qua
- Thêm 1 hàng vào bảng các prime implicant ứng với c.x_2' (001_)

→ Hàng này bị chứa bởi c và _01_

	0_1_	_01_	_101	1_01
001_	x	x		

Rút gọn bảng Prime Implicant

- Thêm cube 0_1_ vào Er, ta có

  E_r = {01_ _, 10_ _, 0_1_ }

  R_p = { _01_, _101, 1_01}

Gán c = _01_

→ H_c = {0_, 1_ } (trên x_1 và x_4)

Các cube tương ứng 0_1_ và 10_ _∈ Er bị bỏ qua

Rút gọn bảng Prime Implicant

- Thêm cube _01_ vào E_r, ta có

  E_r = {01_ _, 10_ _, 0_1_, _01_ }

  R_p = { _101, 1_01}

- Gán c = _101

  → H_c = {0, 1 } (trên x1)

  Các cube tương ứng
  - 01_ _∈ E_r bị bỏ qua
  - _101 ∈ R_p

  Thêm 1 hàng vào bảng các prime implicant ứng với c.x1 (1101)

	0_1_	_01_	_101	1_01
001_	x	x		
1101			x	x

Phương pháp tối giản tối ưu

- Hai loại chính:
  - Các phương pháp tối ưu mà theo chiến lược exact minimizer nhưng không cần thiết sinh toàn bộ tập các prime implicant và giải quyết vấn đề covering một cách tối ưu
  - Các phương pháp tối ưu dựa trên mở rộng lặp, giảm (reduction), và reshaping các implicant trong một cover

Tối ưu dựa trên Exact Minimization

- Một tập các prime implicant mà chứa một particular cube c có thể được sinh ra sử dụng một sự thay đổi đơn giản mà đã được trình bày ở phần trước (sinh prime implicant).

Tối ưu dựa trên Exact Minimization

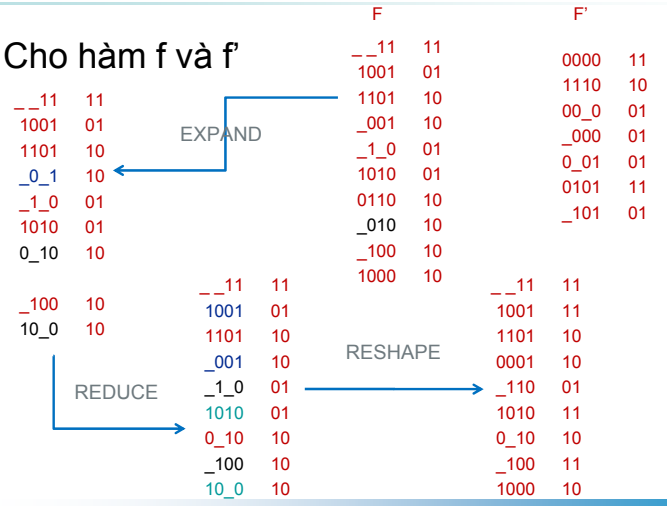
- Ví dụ:
  - Nếu một cube c có 1 tại vị trí j thì ta thiết lập biến C^0_j là 0 trong biểu thức I (xem slide 27).
  - Nếu một cube c có _ tại vị trí j thì ta thiết lập cả c^0_j và c^1_j là 0 trong biểu thức I
  - Các prime implicant của I sẽ tương ứng với các prime implicant của hàm f
  - Các prime implicant có thể được sinh mà chứa một tập hợp các cube được chọn một cách tối ưu trong cover của hàm f
  - Giải thuật covering tối ưu (mà không quay lui) có thể được sử dụng trong bảng prime implicant sinh ra từ tập con này của các prime.

Tối ưu dựa trên Iterative Improvement

- Ba bước hiệu chỉnh cơ bản:
  - Mỗi implicant được rút gọn với kích thước nhỏ nhất có thể
  - Các implicant được kiểm tra từng cặp để xem liệu nó có thể được reshape bằng cách rút gọn một cái và làm lớn cái còn lại.
  - Với các implicant được làm lớn (đến mức lớn nhất của nó) thì phải bỏ đi những implicant khác mà bị nó chứa.

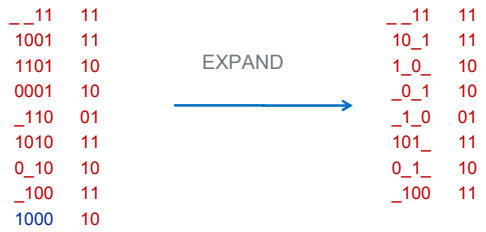
Tối ưu dựa trên Iterative Improvement

- Cho hàm f và f'



Tối ưu dựa trên Iterative Improvement

- Expand lần 2



Lặp tối giản ESPRESSO

```
MINIMIZE(F, D);
{
    R = COMPLEMENT(F D)I
    Do {
      ϕ1= |F|;
      F = REDUCE(F, D);
      F = EXPAND(F, R);
      F = IRREDUNDANT(F, D);
    } while(|F| < ϕ1);
    F = MAKE_SPARSE(F, D, R);
    Return(F);
}
```


Espresso Inputs and Outputs

f(A,B,C,D) = m(4,5,6,8,9,10,13) + d(0,7,15)

Espresso Input

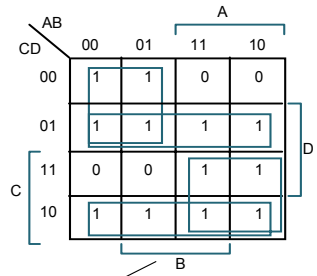
```
.i 4                -- # inputs
.o 1                -- # outputs
.ilb a b c d        -- input names
.ob f               -- output name
.p 10               -- number of product terms
0100  1             -- A'BC'D'
0101  1             -- A'BC'D
0110  1             -- A'BCD'
1000  1             -- AB'C'D'
1001  1             -- AB'C'D
1010  1             -- AB'CD'
1101  1             -- ABC'D
0000  -             -- A'B'C'D' don't care
0111  -             -- A'BCD don't care
1111  -             -- ABCD don't care
.e                  -- end of list
```

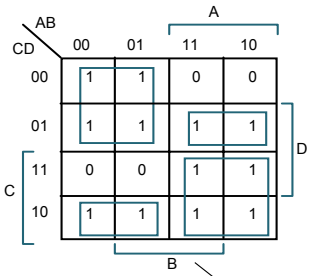
Espresso Output

```
.i 4
.o 1
.ilb a b c d
.ob f
.p 3
1-01  1
10-0  1
01--  1
.e
```

f= A C' D + A B' D' + A' B

Espresso: Why Iterate on Reduce, Irredundant Cover, Expand?





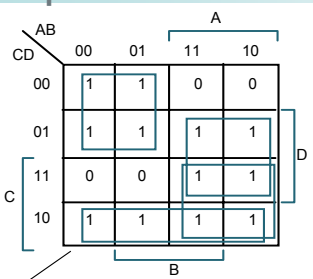
Initial Set of Primes found by Steps1 and 2 of the Espresso Method

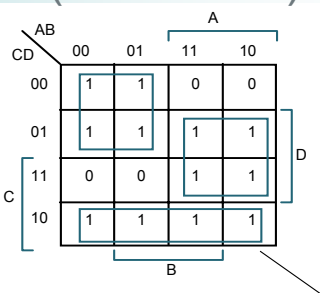
4 primes, irredundant cover, but not a minimal cover!

Result of REDUCE: Shrink primes while still covering the ON-set

Choice of order in which to perform shrink is important

Espresso Iteration (Continued)



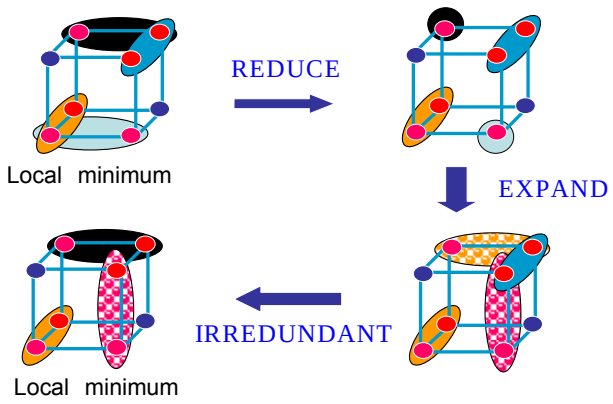


Second EXPAND generates a different set of prime implicants

IRREDUNDANT COVER found by final step of espresso

Only three prime implicants!

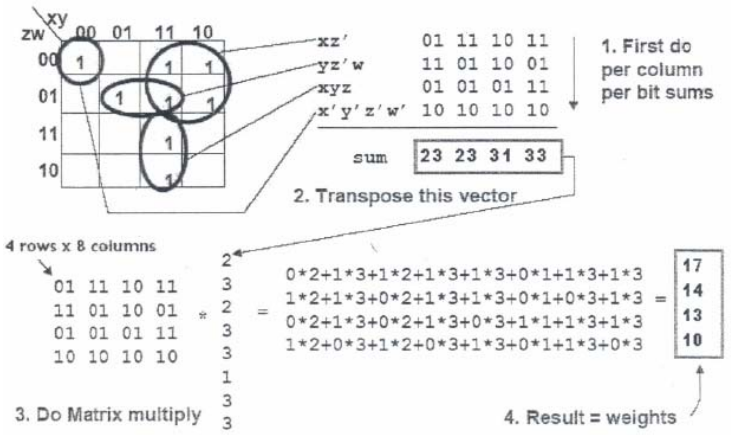
ESPRESSO Illustrated



Expand

- Expand the cubes that are unlikely to be covered by other cubes
- Selection:
  - Compute vector of column sums
  - *Weight*: inner product of cube and vector
  - Sort implicants in ascending order of weight
- Rationale:
  - Low weight correlates to having few 1s in densely populated columns

Expand Weighting Cubes



Example

- f = a'b'c' + ab'c' + a'bc' + a'b'c

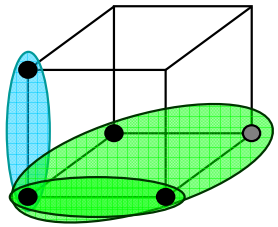
  DC-set = abc'

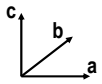
  10 10 10
  01 10 10
  10 01 10
  10 10 01

- Ordering:
  - Vector: $[3\ 1\ 3\ 1\ 3\ 1]^T$
  - Weights: (9, 7, 7, 7)
- Select second implicant.

Example (2)

α 10 10 10
β 01 10 10
γ 10 01 10
δ 10 10 01





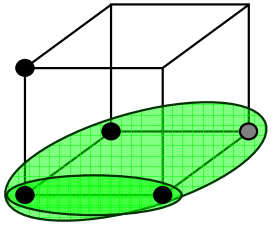
Example (3)

- OFF-set:

  01 11 01
  11 01 01

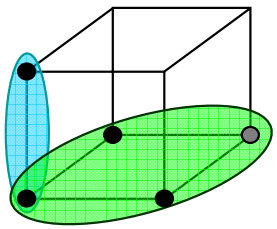
- Expand 01 10 10:
  - 11 10 10 valid.
  - 11 11 10 valid.
  - 11 11 11 invalid.
- Update cover to:

  11 11 10
  10 10 01



Example (4)

```
11  11  10
10  10  01
```



- Expand 10 10 01:
  - 11 10 01 invalid.
  - 10 11 01 invalid.
  - 10 10 11 valid.
- Expand cover:

```
11  11  10
10  10  11
```

Reduce

- Sort implicants
  - Heuristics: sort by descending weight
  - Opposite to the heurstic sorting for expand
- Maximal reduction can be determine exactly
- Theorem:
  - Let α be in F and Q = F U D – { α }
    Then, the maximally reduced cube is:
    $\underline{\alpha} = \alpha \cap \text{supercube}(Q'_\alpha)$

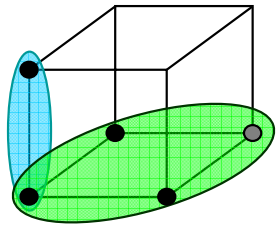
Example

- Expand cover:

```
α 11  11  10
β 10  10  11
```

$Q = F \cup D - \{ \alpha \}$

$\underline{\alpha} = \alpha \cap \text{supercube}(Q'\alpha)$



- Select first implicant:
  - Q'=

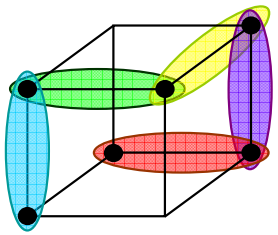
```
01  11  11
11  01  11
```

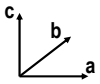
  - Supercube= 11 11 11
  - Cannot be reduced.
- Select second implicant: Q'= 11 11 01
  - $\underline{\beta} = \beta \cap \text{supercube}(Q'_\beta) = 10\ 10\ 01$
- Reduced cover:

```
11  11  10
10  10  01
```

Irredundant cover

α	10	10	11
β	11	10	01
γ	01	11	01
δ	01	01	11
ε	11	01	10





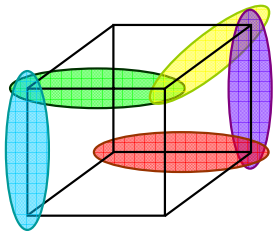
Irredundant cover

- Relatively essential set E^r
  - Implicants covering some minterms of the function not covered by other implicants
  - Important remark: we do not know all the primes!
- Totally redundant set R^t
  - Implicants covered by the relatively essentials
- Partially redundant set R^p
  - Remaining implicants

Example

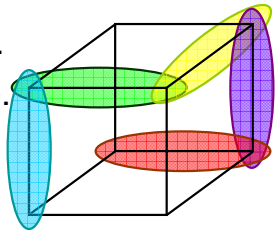
α	10	10	11
β	11	10	01
γ	01	11	01
δ	01	01	11
ε	11	01	10

- $E^r = \{\alpha, \varepsilon\}$
- $R^t = \varnothing$
- $R^p = \{\beta, \gamma, \delta\}$



Example (2)

- Covering relations:
  - β is covered by $\{\alpha, \gamma\}$.
  - γ is covered by $\{\beta, \delta\}$.
  - δ is covered by $\{\gamma, \varepsilon\}$.



- Minimum cover: $\gamma \cup E^r$