KUNGL TEKNISKA HÖGSKOLAN

International Master Program in System-on-Chip Design

L6: Two-level minimization

Reading material

- de Micheli pp. 269 - 343

Formulation of the two-level minimization problem

input: a Boolean function $f(x_1,x_2, \ldots,x_n)$

output: a SOP expression for f of type

$$f(x_1,x_2, \ldots,x_n) = P_1 + P_2 + \ldots + P_k$$

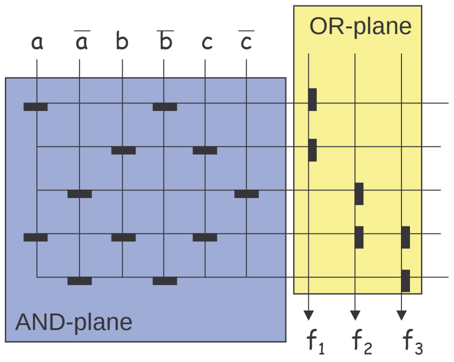
with the minimal number k of products P_k

- minimal sum-of-product form has at most 2^{n-1} product-terms

Goal of two-level minimization

- To reduce the size of SOP, since the size of a PLA is directly proportional to the size of a SOP:
  - the number of columns equals to the number of products in SOP form
  - the number of connections per column equals to the number of variables in the products

PLA



Cube table

abc	$f_1f_2f_3$
10-	1 ~ ~
-11	1 ~ ~
0-0	~ 1 ~
111	~ 1 1
00-	~ ~ 1

Single vs multplie-output functions

- The goal of two-level minimization of a single-output function is to reduce the number of products in the SOP form
- The goal of two-level minimization of a multiple-output function is to reduce the total number of products required to represent all output functions

Notation

- There is one-to-one correspondence between the cube notation and Boolean expression notation:
  - cube = product-term
    - 0-1 is a cube; $x_1' x_3$ is a product-term
  - set of cubes = sum-of-product expression
    - {0-1, -1-} is a set of cubes; $x_1' x_3 + x_2$ is the corresponding sum-of-product expression

Definitions: implicants

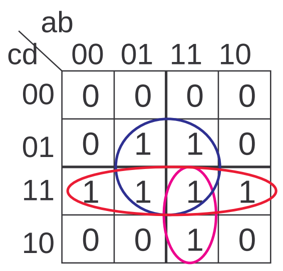
- A cube c is an implicant iff $c \subseteq F_f$
- Prime implicant is an implicant which is not contained in any other implicant
- A prime implicant is essential if there is a minterm covered by that implicant, but no other prime implicant
  - Example: f = abc + bd + cd; all implicants are prime; abc is essential, since abcd' $\in$ abc, but not in bd or cd

Definitions: cover

- A set of cubes S is called cover for f iff $F_f \subseteq S \subseteq F_f \cup D_f$
- Minimum cover is a cover with the minimum number of cubes
- Irredundant cover is a cover S such that for any cube $c \in S$, $F_f \not\subseteq S - \{c\}$
- Prime cover is a cover consisting of only prime implicants

Example

- Example: f = abc + bd + cd is minimum, prime and irredundant cover for f

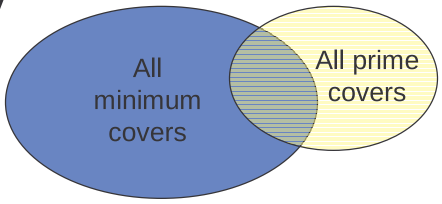


cd \ ab	00	01	11	10
00	0	0	0	0
01	0	1	1	0
11	1	1	1	1
10	0	0	1	0

Exact minimization

- Quine's Theorem: For any Boolean function, there exists a minimum cover which is prime

Consequence: the search for a minimum cover can be restricted to covers by prime implicants only



Quine-Mc Cluskey procedure

- Step 1: Compute all prime implicants P of $F_f \cup D_f$
- Step 2: Compute all minterms m of F_f
- Step 3: Bulid a matrix B with columns representing prime implicants and rows representing minterms

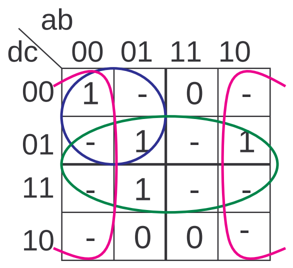
$$B_{ab} = 1 \text{ if } m_a \in P_b$$
$$= 0 \text{ otherwise}$$

- Step 4: Solve the minimum column covering problem for B

Example

F_f = a'b'c'd'+a'bcd'+ab'cd'+a'bcd

D_f = b'd+abc+a'b'cd'+a'bc'd'+ ab'c'd'



dc \ ab	00	01	11	10
00	1	-	0	-
01	-	1	-	1
11	-	1	-	-
10	-	0	0	-

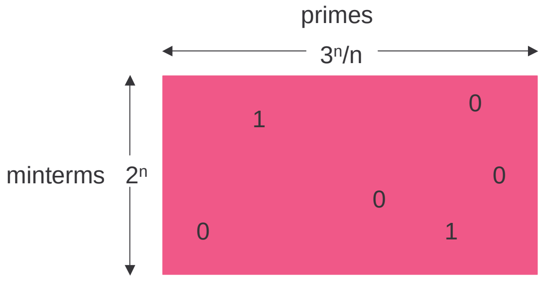
Primes: b'+c+a'd'

	b'	c	a'd'
a'b'c'd'	1	0	1
a'bcd'	0	1	1
ab'cd'	1	1	0
a'bcd	0	1	0

Covering table solution:
{1,2} $\Rightarrow$ b'+c is a minimum prime cover (also c + a'd')

Complexity analysis

Note: ~ 2^n minterms, ~ $3^n/n$ primes



Thus O(2^n) rows and O($3^n/n$) columns AND minimum covering problem is NP-complete. (Hence can probably be double exponential in size of input, i.e. difficulty is O(2^{3^n}))

Essential prime implicants

Primes of $F_f \cup D_f$

Minterms of F_f

	b'	c	a'c
a'b'c'd'	1	0	1
a'bc'd	0	1	1
ab'c'd	1	1	0
a'bcd	0	1	0

$\leftarrow$ Row singleton

$\uparrow$ Essential prime

Definition: Prime implicant is essential if there is a minterm covered by that implicant, but no other prime implicant

Row and Column Dominance

Definition: A row i_1 whose set of primes is contained in the set of primes of row i_2 is said to dominate i_2.

Example:

i_1	**011010**
i_2	**011110**

i_1 dominates i_2

We can remove row i_2, because we have to choose a prime to cover i_1, and any such prime also covers i_2. So i_2 is automatically covered.

Row and Column Dominance

Definition: A column j_1 whose rows are a superset of another column j_2 is said to dominate j_2.

Example:

j_1 dominates j_2

j_1	j_2
1	0
0	0
1	1
0	0
1	1

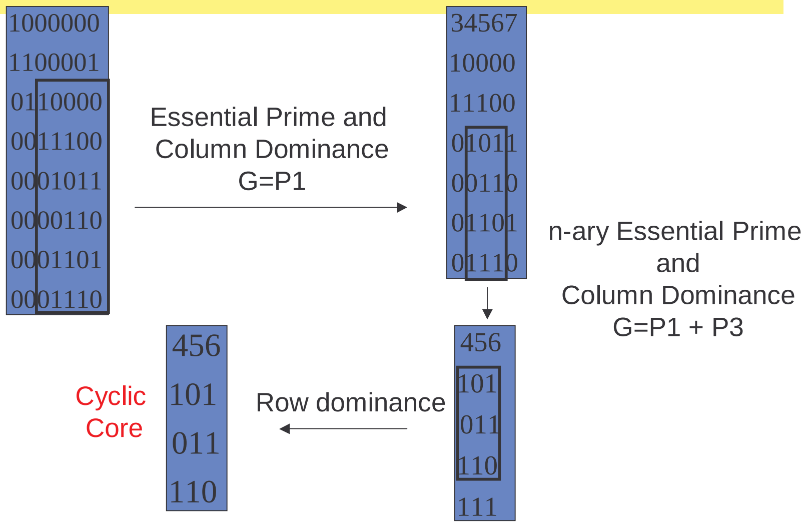
We can remove column j_2 since j_1 covers all those rows and more. We never choose j_2 in a minimum cover since it can always be replaced by j_1.

Pruning the covering table

1. Remove all rows covered by essential columns (columns in row singletons). Put these columns in the cover G.
2. Remove dominated rows. For equal rows, keep one row to represent them.
3. Remove dominated columns. For equal columns, keep one column to represent them.
4. Newly formed row singletons define n-ary essential primes.
5. Go to 1 if covering table decreased.

- The resulting reduced covering table is called the cyclic core. This has to be solved. A minimum solution is added to G -the set of n-ary essential primes. The resulting G is a minimum cover.

Example



Quine-McCluskey Summary

Q-M:

1. Generate cover of all primes for F_f U D_f
2. Make it irredundant (in optimum way)

Note: Q-M is exact i.e. it gives an exact minimum

Heuristic Methods:

1. Generate a cover using some of the primes
2. Make it irredundant (may be not optimally)
3. Keep best result - try again (i.e. go to 1)

Heuristic minimization algorithms

- Provide irredundant covers with reasonably small cardinality
  - a close to optimal instead of optimal solution
- Fast and therefore applicable to large practical functions
- Avoid problems of exact minimization:
  - generation and storage of all prime implicants
  - minimum covering problem (NP-complete)
- All minimization tools use heuristics

Heuristic minimization principles:

- Given an initial cover:
  - Make it prime
  - Make it irredundant
  - Iterate to decrease the size of the cover by modifying the implicants

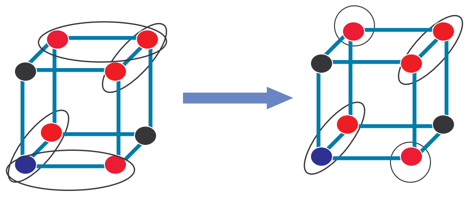
Heuristic minimization procedures

- Reduce()
    - reduces the size of each implicant which preserving the cover
- Expand()
    - makes implicants prime
    - removes covered implicants
- Irredundant()
    - makes the cover irredundant
- Reshape()
    - modifies implicant pairs by enlarging one and reducing the other

Purpose of applying the procedures

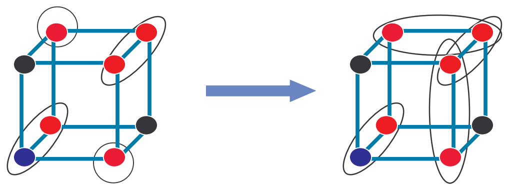
- Reduce()
    - attempts to cover a minterm with minimum (possibly just one) number of implicants
    - some implicants may get eliminated
    - the resulting cover is of the same size or less than the original cover

● on
● off
● don't care



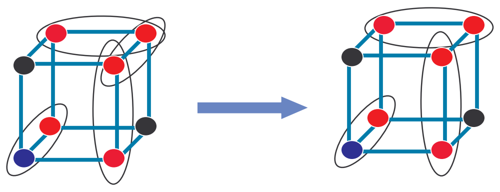
Purpose of applying the procedures

- Expand()
  - results in merging many implicants to form prime implicants
  - reduces the size of the cover (and hence the PLA)



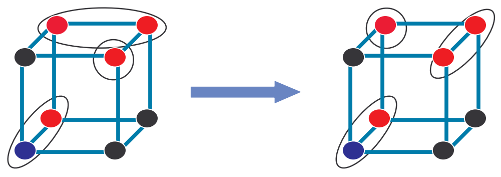
Purpose of applying the procedures

- Irredundant()
  - checks whether any of the implicant in the cover can be dropped
  - leads to the reduction in the size of the cover



Purpose of applying the procedures

- Reshape()
  - generates a different cover of the same size
  - gives the possibility of avoiding a local minima



Example

Minterms

0000 1
0010 1
0100 1
0110 1
1000 1
1010 1
0101 1
0111 1
1001 1
1011 1
1101 1

Prime implicants

a	0--0 1
b	-0-0 1
c	01-- 1
d	10-- 1
e	1-01 1
g	-101 1

Example: Expansion

- Starting point: include all minterms in the cover
{0000,0010,0100,1000,1010,0101,0111,1001,1011,1101}
- Take every minterm in the cover and expand it to a prime implicant.
- Drop the minterms covered by the generated prime implicant
  - Expand 0000 to a = 0--0. Drop 0100, 0010, 0110 from the cover.
  - Expand 1000 to b = -0-0. Drop 1010 from the cover.
  - Expand 0101 to c = 01--. Drop 0111 from the cover.
  - Expand 1001 to d = 10--. Drop 1011 from the cover.
  - Expand 1101 to e = 1-01.
- Resulting cover = {a, b, c, d, e}

Example: Reduction

- Starting cover = {a, b, c, d, e}
- Pick up an implicant from the cover and replace a don’t care "-" in it with a 0 or 1 in such a way that the resulting implicant along with the other implicants in the cover still give a valid cover
  - Reduce b = -0-0 to b´ = 00-0
  - Reduce e = 1-01 to e´ = 1101
  - Reduce a = 0--0 to nothing, i.e. remove it from the cover
  - It is not possible to reduce g and d
- Resulting cover = {b´, c, d, e´}

Example: Reshape

- Starting cover = {b´, c, d, e´}
- Pick up an implicant pair. Expand one and reduce the other implicant in such a way that this modification results in a valid cover. All pairs have to be tried twice - expand first and reduce second, and vice versa
  - Reshape the pair {b´, d} by expanding b´ and reducing d.
  - b´ = 00-0 is expanded to -0-0 = b
  - d = 10-- is reduced to 10-1 = d´
  - Trying to reshape any other pair results in no change
- Resulting cover = {b, c, d´, e´}

Example: Second Expansion

- Starting cover = {b, c, d´, e´}
  - Expand d´ = 10-1 to d = 10--
  - Expand e´ = 1101 to e = 1-01
- Resulting cover = {b, c, d, e}
- This sequence of procedures is use in MINI. Expand, Reduce, Reshape are repeatedly applied until there is no improvement

Alternative example: Espresso

- Espresso uses a sequence of procedures Expand, Reduce, Irredundant.
- They are repeatedly applied until there is no improvement
- Espresso uses multiple-valued logic: a multiple-output Boolean function f: $B^n \rightarrow (B\cup\{-\})^k$ is treated as an (n+1)-variable 1-output function of type

$$f: B^n \cup \{0,1,\ldots,k-1\} \rightarrow B\cup\{-\}$$

Pseudocode of ESPRESSO

```
ESPRESSO($\Im$)
{
  (F,D,R) $\leftarrow$ DECODE($\Im$)
  F $\leftarrow$ EXPAND(F,R)
  F $\leftarrow$ IRREDUNDANT(F,D)
  E $\leftarrow$ ESSENTIAL_PRIMES(F,D)
  F $\leftarrow$ F-E;  D $\leftarrow$ D+E
  do{
    do{
      F $\leftarrow$ REDUCE(F,D)
      F $\leftarrow$ EXPAND(F,R)
      F $\leftarrow$ IRREDUNDANT(F,D)
    }while fewer terms in F
    //LASTGASP
    G $\leftarrow$ REDUCE_GASP(F,D)
    G $\leftarrow$ EXPAND(G,R)
    F $\leftarrow$ IRREDUNDANT(F+G,D)
    //LASTGASP
  }while G $\neq \emptyset$
  F $\leftarrow$ F+E;  D $\leftarrow$ D-E
  LOWER_OUTPUT(F,D)
  RAISE_INPUTS(F,R)
  error $\leftarrow$ ($F_{on} \not\subset$ F) or (F $\not\subset F_{old}$ + D)
  return (F,error)
}
```

New Exact Methods

For 40 years, Q-M methods remained basically unchanged. In1992, two fundamentally new methods developed:

- McGeer's method
  - based on essential signature cubes concept
- Coudert and Madre method
  - implicit QM based on BDD's
- In both cases, results are superior to ESPRESSO-EXACT (both in speed and in the number of problems solved)