Logic Optimization

Heuristic Based

Virendra Singh

Professor

Department of Electrical Engineering &

Dept. of Computer Science & Engineering

Indian Institute of Technology Bombay

http://www.ee.iitb.ac.in/~viren/

E-mail: viren@{ee, cse}.iitb.ac.in

EE-677: Foundations of VLSI CAD

Lecture 27 on 12 Oct 2021

CADSL

Logic Minimization

Exact

Run time

- Minimal ✓

Minimum ✗

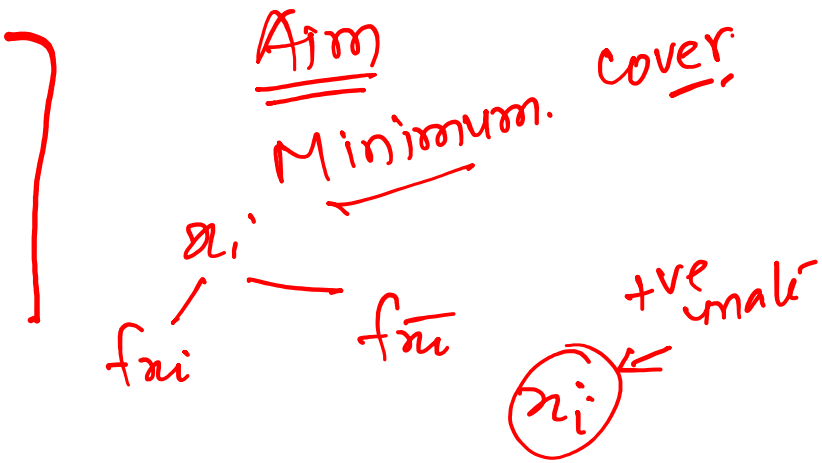
Matrix representation of logic covers

- Representations used by logic minimizers
- Different formats
  - Usually one row per implicant
- Symbols:
  - 0, 1, * , …
- Encoding:

∅	00
0	10
1	01
*	11

Operations on logic covers

- Recursive paradigm
  - Expand about a mv-variable
  - Apply operation to co-factors
  - Merge results
- Unate heuristics
  - Operations on unate functions are simpler
  - Select variables so that cofactors become unate functions
- Recursive paradigm is general and applicable to different data structures
  - Matrices and binary decision diagrams



+ve unate w.r.t x_i

$f = x_i f_{x_i} + \bar{x}_i \cdot f_{\bar{x}_i}$

$\equiv x_i f_{x_i} + f_{\bar{x}_i}$

$f = x_1 x_2 + \bar{x}_2 x_3$

x_1

$f_{x_1} = x_2 + \bar{x}_2 \cdot x_3$

$f_{\bar{x}_1} = \bar{x}_2 \cdot x_3$

$x_1 (x_2 + \bar{x}_2 x_3) + \bar{x}_1 \cdot (\bar{x}_2 x_3)$

$= x_1 x_2 + x_1 \bar{x}_2 x_3 + \bar{x}_1 \bar{x}_2 x_3 = x_1 x_2 + \bar{x}_2 x_3$

$= x_1 (x_2 + \bar{x}_2 x_3) + \bar{x}_2 x_3$

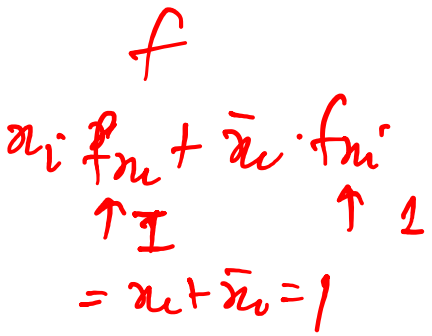
$= x_1 x_2 + \bar{x}_2 x_3$.

$f = x_i \cdot f_{x_i} + f_{\bar{x}_i}$ +ve unate wrt x_i

$f = x_i f_{x_i} + \bar{x}_i \cdot f_{\bar{x}_i}$ -ve unate wrt x_i.

Tautology

- Check if a function is always TRUE
- Recursive paradigm:
  - Expend about a mvi variable
  - If all cofactors are TRUE, then the function is a tautology
- Unate heuristics
  - If cofactors are unate functions, additional criteria to determine tautology
  - Faster decision



Recursive tautology

- TAUTOLOGY:
  - The cover matrix has a row of all 1s. (Tautology cube)
- NO TAUTOLOGY:
  - The cover has a column of 0s. (A variable never takes a value)
- TAUTOLOGY:
  - The cover depends on one variable, and there is no column of 0s in that field
- Decomposition rule:
  - When a cover is the union of two subcovers that depend on disjoint sets of variables, then check tautology in both subcovers

Containment

- Theorem:
  - A cover F contains an implicant α if and only if F_α is a tautology

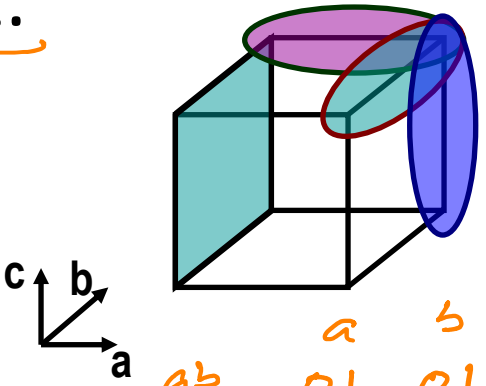
- Consequence:
  - Containment can be verified by the tautology algorithm

Example: f = ab + ac + a'

- Check covering of bc : 11 01 01.
- Take the cofactor:

a	b	c
01	11	11
01	11	11
10	11	11

- Tautology – bc is contained by f.



	a	b	c
ab	01	01	11
ac	01	11	01
$\bar{a}$	10	11	11
	11	01	01
AND	01	01	01
	01	01	01
	10	01	01
	00	10	10
	01	11	11
	01	11	11
	10	11	11

Complementation

- Recursive paradigm
  - $f' = x f'_x + x' f'_{x'}$
- Steps:
  - Select variable
  - Compute co-factors
  - Complement co-factors

$$f = x_i f_{x_i} + \bar{x}_i \cdot f_{\bar{x}_i}$$

$$\bar{f} = \overline{x_i f_{x_i}} \cdot \overline{(\bar{x}_i \cdot f_{\bar{x}_i})}$$

$$= (\bar{x}_i + \bar{f}_{x_i}) \cdot (x_i + \bar{f}_{\bar{x}_i})$$

$$= \bar{x}_i \cdot \bar{f}_{\bar{x}_i} + x_i \bar{f}_{x_i} + \bar{f}_{x_i} \bar{f}_{\bar{x}_i}$$

$$= \bar{x}_i \cdot \bar{f}_{\bar{x}_i} + x_i \bar{f}_{x_i}$$

- Recur until cofactors can be complemented in a straightforward way

ON set OFF set f'

Termination rules

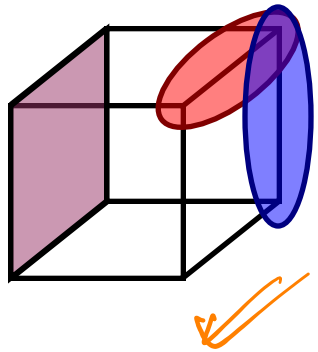
- The cover F is void
  - Hence its complement is the universal cube
- The cover F has a row of 1s
  - Hence F is a tautology and its complement is void
- The cover F consists of one implicant.
  - Hence the complement is computed by DeMorgan's law
- All implicants of F depend on a single variable, and there is not a column of 0s.
  - The function is a tautology, and its complement is void

Unate functions

- Theorem:
  - If f is positive unate in x, then
    - $f' = f'_x + x' f'_{x'}$
  - If f is negative unate in x, then
    - $f' = x f'_x + f'_{x'}$
- Consequence:
  - Complement computation is simpler
  - Follow only one branch in the recursion
- Heuristics
  - Select variables to make the cofactor unate

# Example
f = ab + ac + a'

- Select binate variable a

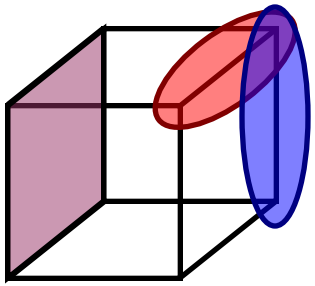


- Compute cofactors:
  - $F_{a'}$ is a tautology, hence $F'_{a'}$ is void.
  - F_a yields:

11	01	11
11	11	01

Example (2)

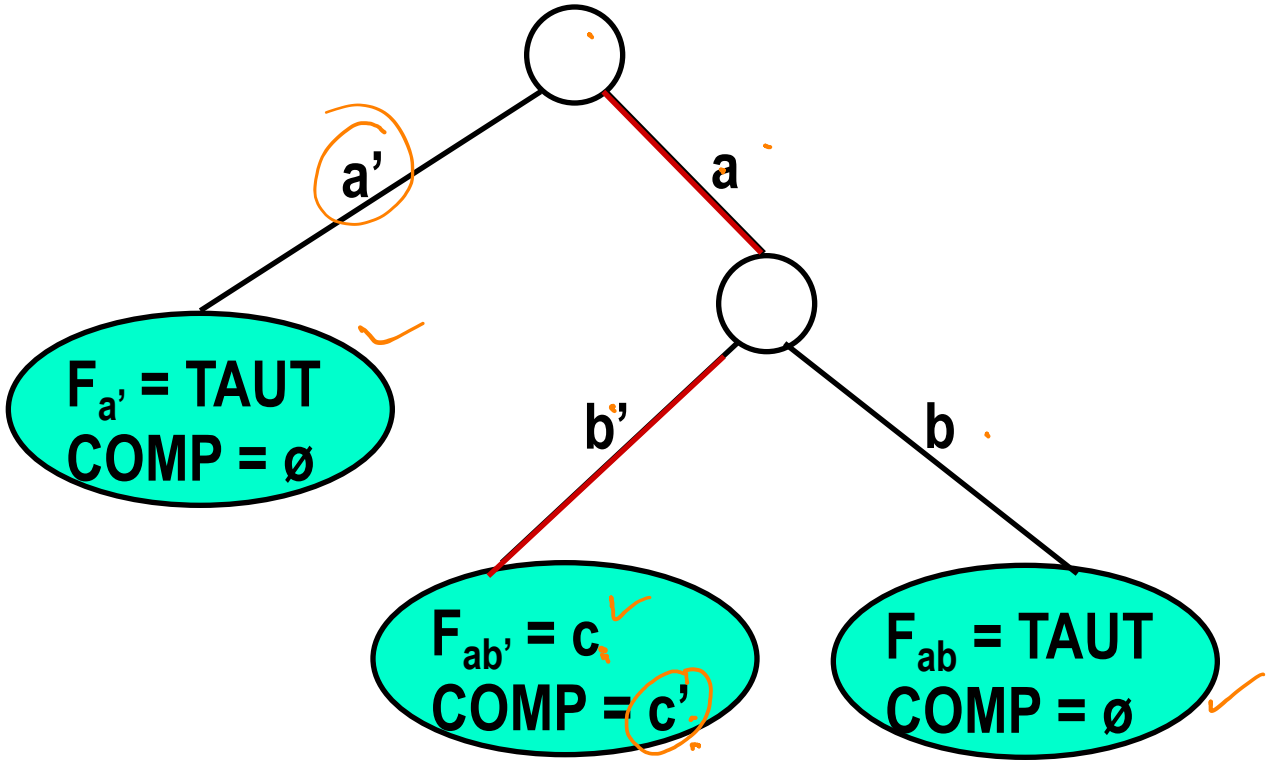
- Select unate variable b
- Compute cofactors:
  - F_{ab} is a tautology, hence F'_{ab} is void
  - $F_{ab'}$ = 11 11 01 and its complement is 11 11 10



- Re-construct complement:
  - 11 11 10 intersected with *Cube*(b') = 11 10 11 yields 11 10 10
  - 11 10 10 intersected with *Cube*(a) = 01 11 11 yields 01 10 10
- Complement: F' = 01 10 10

Example (3)

- Recursive search:



Complement: a b'c'

Boolean cover manipulation summary

- Recursive methods are efficient operators for logic covers
  - Applicable to matrix-oriented representations
  - Applicable to recursive data structures like BDDs
- Good implementations of matrix-oriented recursive algorithms are still very competitive
  - Heuristics tuned to the matrix representations

Heuristic minimization - operators

- Expand
  - Make implicants prime
  - Removed covered implicants
- Reduce
  - Reduce size of each implicant while preserving cover
- Reshape
  - Modify implicant pairs: enlarge one and reduce the other
- Irredundant
  - Make cover irredundant

Thank You