Lecture 3
EE 421 / CS 425
Digital System Design

Fall 2025

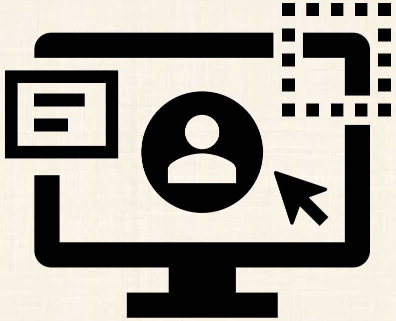
Shahid Masud

Topics

- Quine McCluskey method of Logic Minimization
- Boolean Simplification using n-cube
- WinLogiLab Software Demo
- Examples using WinLogiLab Software Tool

Problem with Logic Minimization

Boolean Algebra and K-Maps can easily solve 4 variables.
Beyond 5 or 6 input variables, we have to look for some other scalable techniques.
The techniques should be systematic and programmable



Quine McCluskey Method – Example 1

$$f(A,B,C,D) = \sum m\,(2,4,6,8,9,10,12,13,15)$$

- Make Table of Minterms
- Count No of 1s

Minterms	Binary	No of 1s
2	0010	1
4	0100	1
6	0110	2
8	1000	1
9	1001	2
10	1010	2
12	1100	2
13	1101	3
15	1111	4

Quine McCluskey – Example 1 - contd

$$f(A,B,C,D) = \sum m(2,4,6,8,10,12,13,15)$$

Table of Minterms
Group as per number of 1s

List 1

Minterms	Binary	No of 1s
2	0010	1
4	0100	1
8	1000	1
6	0110	2
9	1001	2
10	1010	2
12	1100	2
13	1101	3
15	1111	4

Group of one 1s

Group of two 1s

Group of three 1s

Group of four 1s

Quine McCluskey – Example 1 - contd

$$f(A,B,C,D) = \sum m(2,4,6,8,9,10,12,13,15)$$

- Check each entry with an entry in higher block.
- See if there is any entry with One bit change only. These entries can be combined.
- If any entry cannot be combined, it is a Prime Implicant.
- Mark a tick ✓ where combined.
- In Combined terms, put a '–' where both 1 and 0 exist in the two terms

List 1

Minterm	Binary	Combine or PI
2	0010	✓
4	0100	✓
8	1000	✓
6	0110	✓
9	1001	✓
10	1010	✓
12	1100	✓
13	1101	✓
15	1111	✓

List 2

Minterm	Binary	Combined or PI
2, 6	0-10	
2, 10	-010	
4, 6	01-0	
4, 12	-100	
8, 9	100-	
8, 10	10-0	
8, 12	1-00	
9, 13	1-01	
12, 13	110-	
13, 15	11-1	

Quine McCluskey – Example 1 - contd

List 1

Minterm	Binary	Combine or PI
2	0010	✓
4	0100	✓
8	1000	✓
6	0110	✓
9	1001	✓
10	1010	✓
12	1100	✓
13	1101	✓
15	1111	✓

List 2

Minterm	Binary	Combined or PI
2, 6	0-10	PI2
2, 10	-010	PI3
4, 6	01-0	PI4
4, 12	-100	PI5
8, 9	100-	✓
8, 10	10-0	PI6
8, 12	1-00	✓
9, 13	1-01	✓
12, 13	110-	✓
13, 15	11-1	PI7

List 3

Minterm	Binary	Combined or PI
(8, 9), (12, 13)	1-0-	PI1
(8, 12), (9, 13)	1-0-	Same PI1

Quine McCluskey – Example 1 - contd

List 1

Minterm	Binary	Combine or PI
2	0010	✓
4	0100	✓
8	1000	✓
6	0110	✓
9	1001	✓
10	1010	✓
12	1100	✓
13	1101	✓
15	1111	✓

List 2

Minterm	Binary	Combined or PI
2, 6	0-10	PI2
2, 10	-010	PI3
4, 6	01-0	PI4
4, 12	-100	PI5
8, 9	100-	✓
8, 10	10-0	PI6
8, 12	1-00	✓
9, 13	1-01	✓
12, 13	110-	✓
13, 15	11-1	PI7

List 3

Minterm	Binary	Combined or PI
(8, 9), (12, 13)	1-0-	PI1
(8, 12), (9, 13)	1-0-	Same PI1

Quine McCluskey – Example 1 - contd

List 1

Minterm	Binary	Combine or PI
2	0010	✓
4	0100	✓
8	1000	✓
6	0110	✓
9	1001	✓
10	1010	✓
12	1100	✓
13	1101	✓
15	1111	✓

List 2

Minterm	Binary	Combined or PI
2, 6	0-10	PI2
2, 10	-010	PI3
4, 6	01-0	PI4
4, 12	-100	PI5
8, 9	100-	✓
8, 10	10-0	PI6
8, 12	1-00	✓
9, 13	1-01	✓
12, 13	110-	✓
13, 15	11-1	PI7

List 3

Minterm	Binary	Combined or PI
(8, 9), (12, 13)	1-0-	PI1
(8, 12), (9, 13)	1-0-	Same PI1

LUMS

Quine McCluskey Method - contd

PI1=1-0-, PI2=0-10, PI3=-010
PI4=01-0, PI5=-100, PI6=10-0
PI7=11-1

To find minimum number of Prime Implicants, make a PI Chart. PI vs Minterms covered

		✓	✓	✓	✓	✓	✓	✓	✓	✓
PI Needed	Minterm to cover	**2**	**4**	**6**	**8**	**9**	**10**	**12**	**13**	**15**
✓	**PI1**				**X**	**X → Ⓧ**		**X**	**X**	
Eliminate	**PI2**	**X**		**X**						
✓	**PI3**	**X**					**X**			
✓	**PI4**		**X**	**X**						
Eliminate	**PI5**		**X**					**X**		
Eliminate	**PI6**				**X**		**X**			
✓	**PI7**								**X**	**X → Ⓧ**

This **symbol** Ⓧ means this PI is the only cover for this particular Minterm

Quine McCluskey Method - contd

PI1=1-0-, PI2=0-10, PI3=-010
PI4=01-0, PI5=-100, PI6=10-0
PI7=11-1

To find minimum number of Prime Implicants, make a PI Chart. PI vs Minterms covered

		✓	✓	✓	✓	✓	✓	✓	✓	✓
PI Needed	Minterm to cover	**2**	**4**	**6**	**8**	**9**	**10**	**12**	**13**	**15**
✓	**PI1**				**X**	**X → Ⓧ**		**X**	**X**	
Eliminate	**PI2**	**X**		**X**						
✓	**PI3**	**X**					**X**			
✓	**PI4**		**X**	**X**						
Eliminate	**PI5**		**X**					**X**		
Eliminate	**PI6**				**X**		**X**			
✓	**PI7**								**X**	**X → Ⓧ**

This **symbol** Ⓧ means this PI is the only cover for this particular Minterm

Quine McCluskey Method - contd

PI1=1-0-, PI2=0-10, PI3=-010
PI4=01-0, PI5=-100, PI6=10-0
PI7=11-1

To find minimum number of Prime Implicants, make a PI Chart. PI vs Minterms covered

		✓	✓	✓	✓	✓	✓	✓	✓	✓
PI Needed	Minterm to cover	**2**	**4**	**6**	**8**	**9**	**10**	**12**	**13**	**15**
✓	**PI1**				**X**	**X → Ⓧ**		**X**	**X**	
Eliminate	**PI2**	**X**		**X**						
✓	**PI3**	**X**					**X**			
✓	**PI4**		**X**	**X**						
Eliminate	**PI5**		**X**					**X**		
Eliminate	**PI6**				**X**		**X**			
✓	**PI7**								**X**	**X → Ⓧ**

This symbol Ⓧ means this PI is the only cover for this particular Minterm

Quine McCluskey Method - contd

$$f(A,B,C,D) = \sum m(2,4,6,8,9,10,12,13,15)$$

= PI1 + PI3 + PI4 + PI7
= 1-0- + -010 + 01-0 + 11-1
= AC' + B'CD' + A'BD' + ABD

Minterm	Binary	Combine or PI
2, 6	0-10	PI2
2, 10	-010	PI3
4, 6	01-0	PI4
4, 12	-100	PI5
8, 9	100-	✓
8, 10	10-0	PI6
8, 12	1-00	✓
9, 13	1-01	✓
12, 13	110-	✓
13, 15	11-1	PI7

Minterm	Binary	Combineor PI
(8, 9), (12, 13)	1-0-	PI1
(8, 12), (9, 13)	1-0-	Same PI1

Quine McCluskey – Example 1 - contd

$$f(A, B, C, D) = \sum m(2, 4, 6, 8, 10, 12, 13, 15)$$

Answer:

f(A,B,C,D) = PI1 + PI3 + PI4 + PI7

= 1-0- + -010 + 01-0 + 11-1

$$= A\overline{C} + \overline{B}C\overline{D} + \overline{A}B\overline{D} + ABD$$

QM with Don't Care and 5 Variables

$$f(A,B,C,D,E) = \sum m(2,3,7,10,12,15,27)$$

+ d (5, 18, 19, 21, 23) these are don't care terms

Include don't care terms in logic simplification

List all Minterms, note number of 1's

Minterms	Binary	No of 1s
2	00010	1
3	00011	2
7	00111	3
10	01010	2
12	01100	2
15	01111	4
27	11011	5
5	00101	2
18	10010	2
19	10011	3
21	10101	3
23	10111	4

Don't Care Terms

QM with Don’t Care and 5 Variables - contd

$$f(A,B,C,D,E) = \sum m(2,3,7,10,12,15,27)$$

+ d (5, 18, 19, 21, 23) these are don’t care terms

Include don’t care terms in logic simplification

Group minterms as per number of 1s

Minterms	Binary	No of 1s
2	00010	1
3	00011	2
5	00101	2
10	01010	2
12	01100	2
18	10010	2
7	00111	3
19	10011	3
21	10101	3
15	01111	4
23	10111	4
27	11011	4

QM with Don't Care and 5 Variables - contd

$$f(A,B,C,D,E) = \sum m(2,3,7,10,12,15,27)$$

+ d (5, 18, 19, 21, 23)
these are don't care terms

List 1 – Combine with next higher block

Minterms	Binary	Combine or PI
2	00010	✓
3	00011	✓
5	00101	✓
10	01010	✓
12	01100	make PI
18	10010	✓
7	00111	✓
19	10011	✓
21	10101	✓
15	01111	✓
23	10111	✓
27	11011	✓

List 2

Minterms	Binary	Combine or PI
2, 3	0001-	
2, 10	0-010	
2, 18	-0010	
3, 7	00-11	
3, 19	-0011	
5, 7	001-1	
5, 21	-0101	
18, 19	1001-	
7, 15	0-111	
7, 23	-0111	
19, 23	10-11	
19, 27	1-011	
21, 23	101-1	

QM with Don't Care and 5 Variables - contd

List 1 – Combine with next higher block

Minterms	Binary	Combine or PI
2	00010	✓
3	00011	✓
5	00101	✓
10	01010	✓
12	01100	Make PI7
18	10010	✓
7	00111	✓
19	10011	✓
21	10101	✓
15	01111	✓
23	10111	✓
27	11011	✓

List 2 – Combine next higher block

Minterms	Binary	Combine or PI
2, 3	0001-	✓
2, 10	0-010	Make PI4
2, 18	-0010	✓
3, 7	00-11	✓
3, 19	-0011	✓
5, 7	001-1	✓
5, 21	-0101	✓
18, 19	1001-	✓
7, 15	0-111	PI5
7, 23	-0111	✓
19, 23	10-11	✓
19, 27	1-011	PI6
21, 23	101-1	✓

List 3

Minterms	Binary	Combine or PI
(2, 3) , (18, 19)	-001-	PI1
(3, 7), (19, 23)	-0-11	PI2
(5, 7), (21, 23)	-01-1	PI3

QM with 5 Variable, don't care - contd

To find minimum number of Prime Implicants, make a PI Chart. PI vs Minterms covered

		✓	✓	✓	✓	✓	✓	✓
PI Needed	Minterm	**2**	**3**	**7**	**10**	**12**	**15**	**27**
✓	**PI1**	**X**	**X**					
Eliminate	**PI2**		**X**	**X**				
Eliminate	**PI3**			**X**				
✓	**PI4**	**X**			**X →Ⓧ**			
✓	**PI5**			**X**			**X →Ⓧ**	
✓	**PI6**							**X →Ⓧ**
✓	**PI7**					**X →Ⓧ**		

PI1 = -001-
PI2 = -0-11
PI3 = -01-1
PI4 = 0-010
PI5 = 0-111
PI6 = 1-011
PI7 = 01100

This **symbol** Ⓧ means this PI is the only cover for this particular Minterm

QM with Don’t Care and 5 Variables

$$f(A,B,C,D,E) = \sum m(2,3,7,10,12,15,27)$$

+ d (5, 18, 19, 21, 23) these are don’t care terms

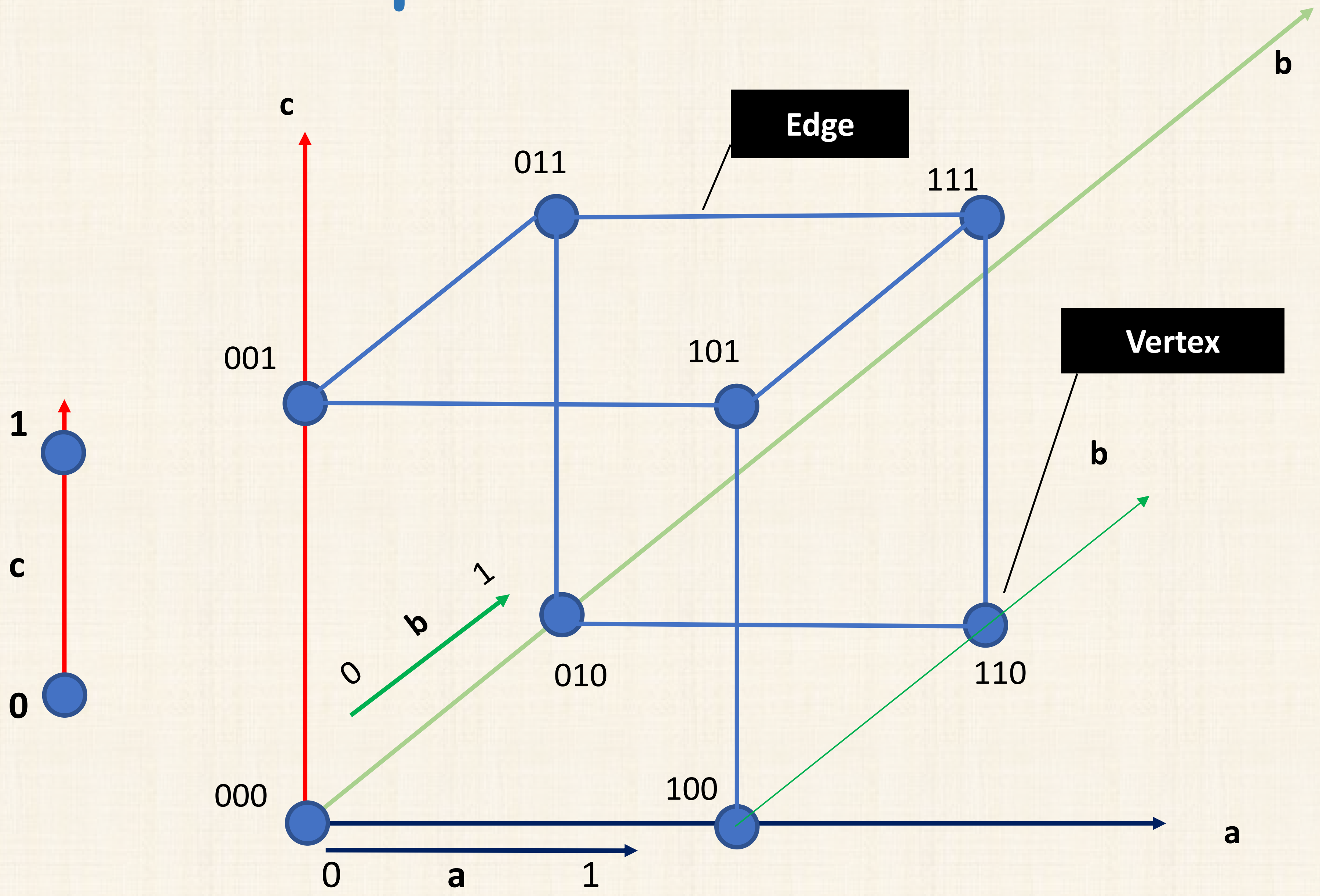
Answer f = PI1 + PI4 + PI5 + PI6 + PI7

= -001- + 0-010 + 0-111 + 1-011 + 01100

$$= \bar{B}\bar{C}D + \bar{A}\bar{C}D\bar{E} + \bar{A}CDE + A\bar{C}DE + \bar{A}BC\bar{D}\bar{E}$$

Boolean Algebra on n-cube

Boolean Expression on n-cube



Minimization using Boolean n-cube

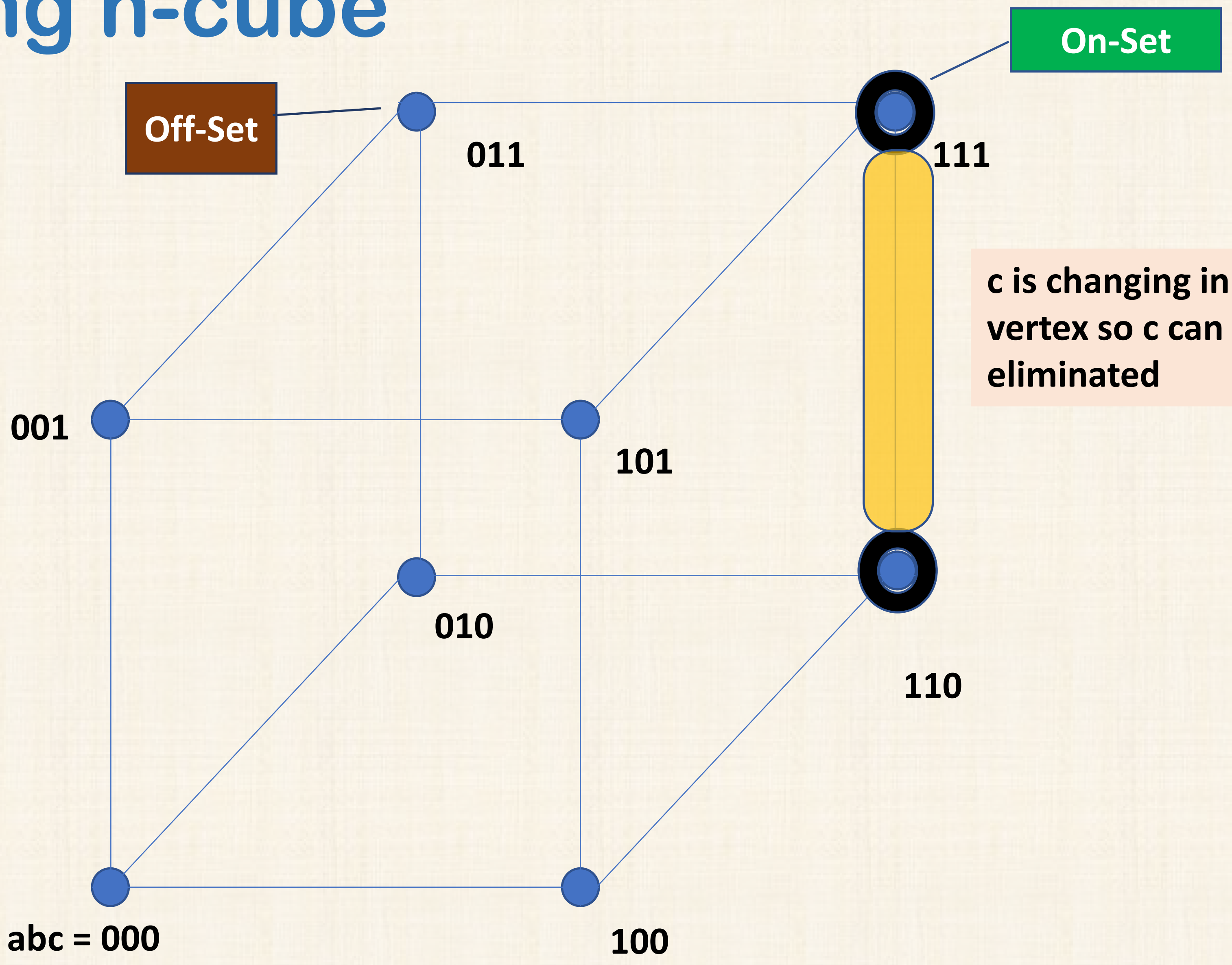
- Repeatedly Combine Cubes that differ in only one literal
- Eliminate Redundant Implicants
- Grouping along multiple dimensions is possible

Example 1 using n-cube

F=abc + abc’
= 111 + 110

Each Edge of Graph is
a minterm
Connection through Vertex
Shows some Adjacent term can be
eliminated

Answer:
F = ab

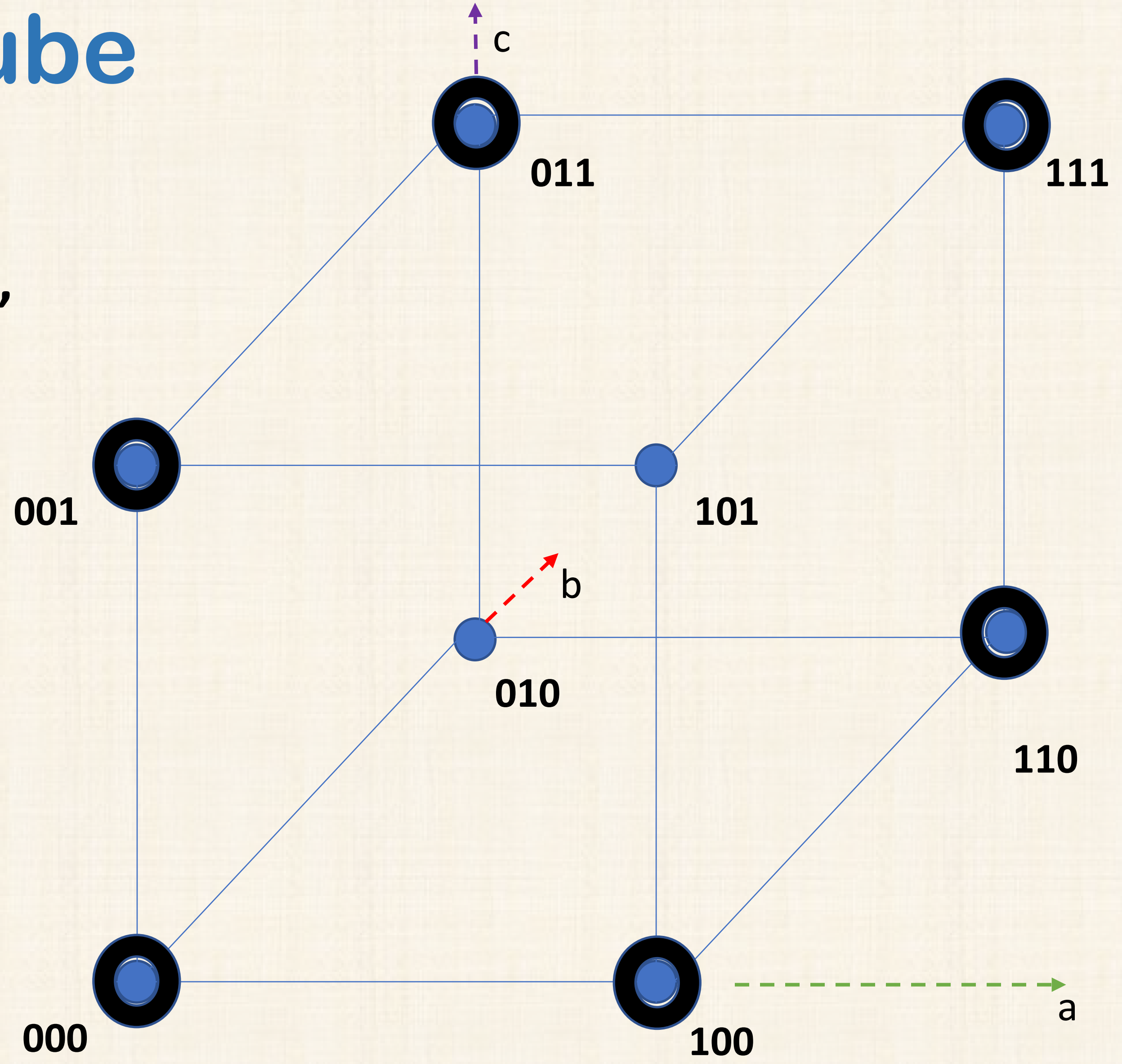


c is changing in this vertex so c can be eliminated

Example 2 using n-cube

F=abc + a’bc + abc’ + a’b’c + ab’c’ + a’b’c’

= 111 + 011 + 110 + 001 + 100 + 000



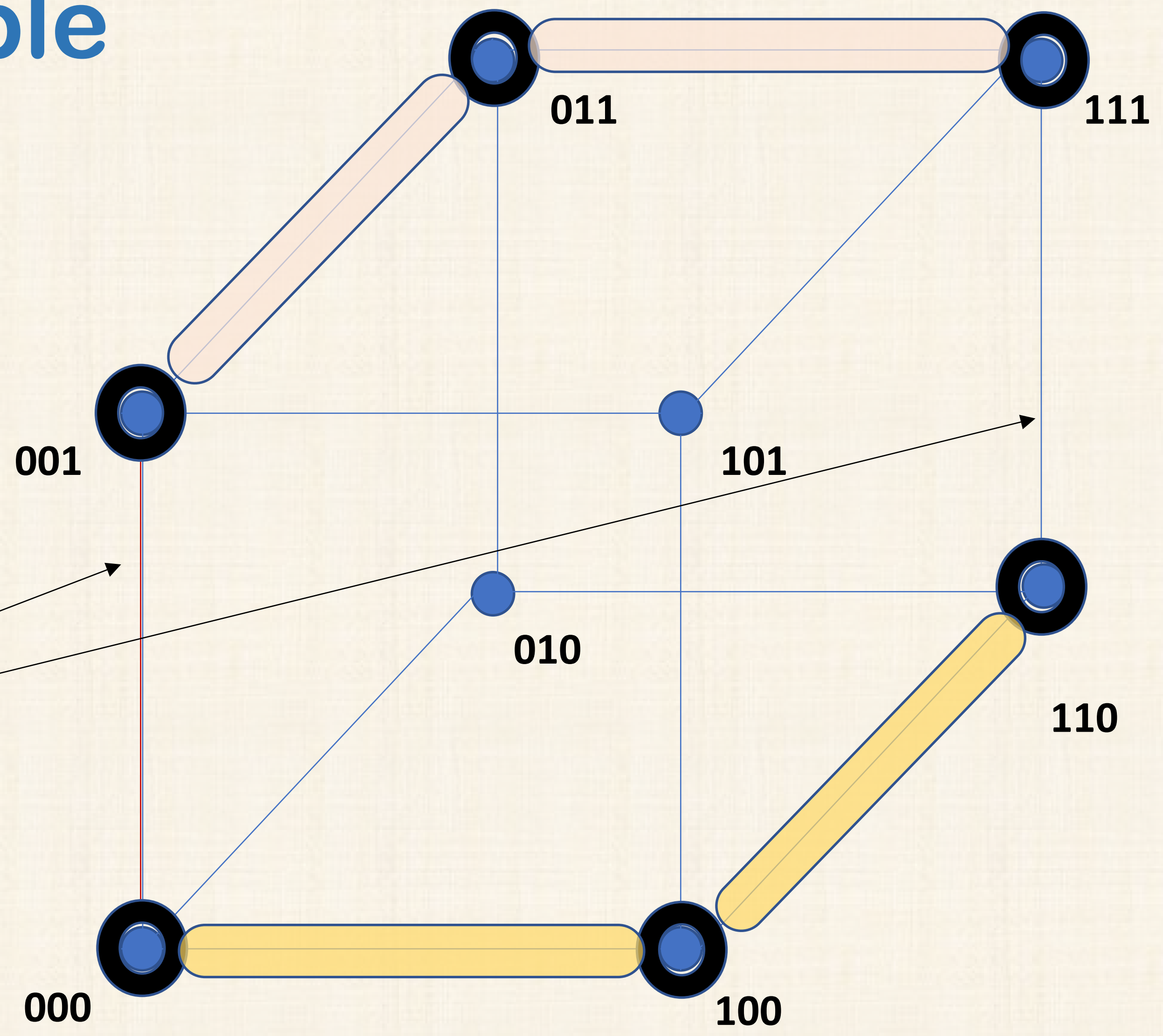
First solution - Example 2 using n-cube

F=abc + a'bc + abc' + a'b'c + ab'c' + a'b'c'
= 111 + 011 + 110 + 001 + 100 + 000

Ignore Redundant Cover of ab and a'b'

Combine Adjacent Vertex into one implicant

Answer:
F= a'c + bc + b'c' + ac'

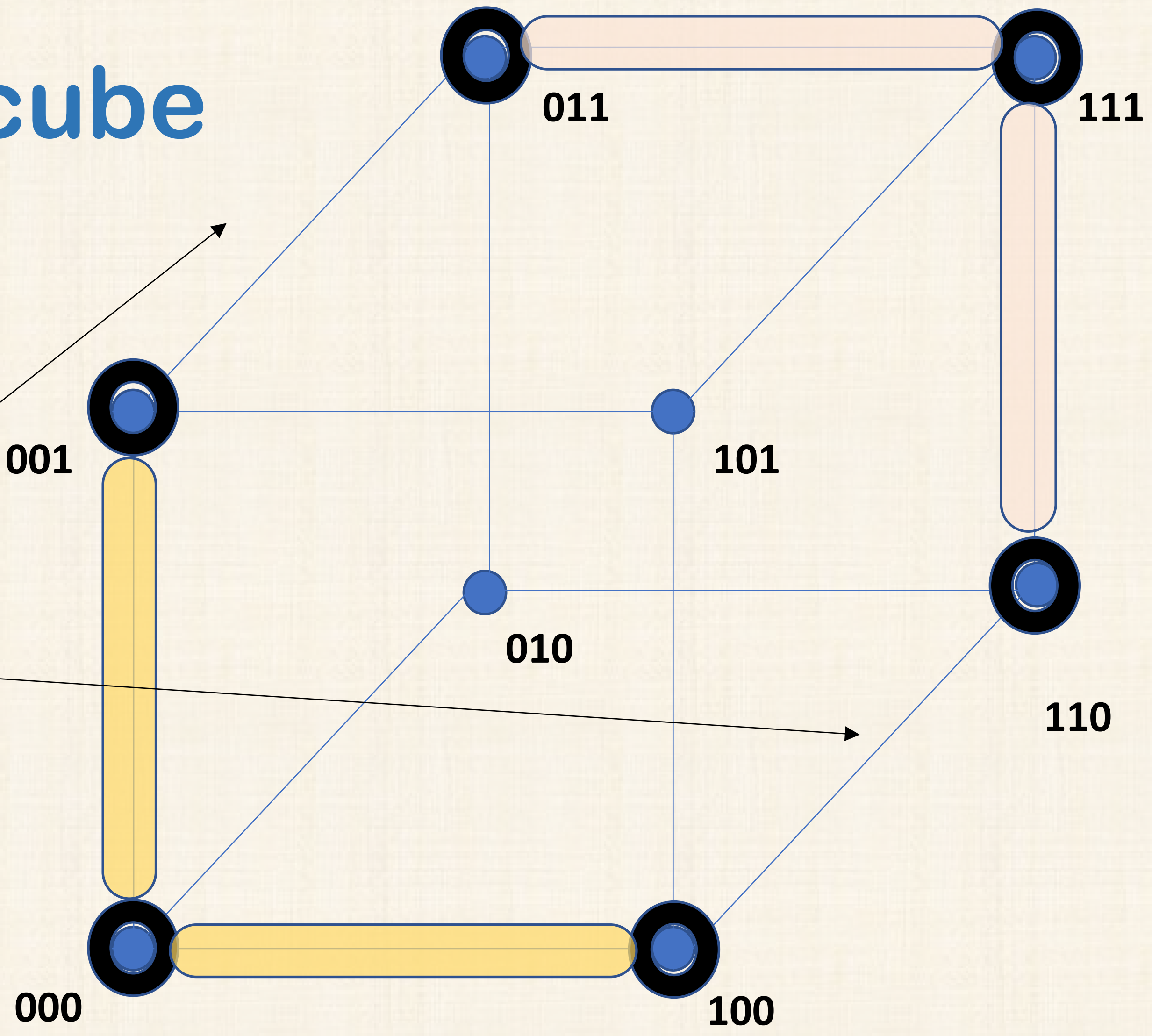


Second solution - Example 2 using n-cube

F=abc + a'bc + abc' + a'b'c + ab'c' + a'b'c'

= 111 + 011 + 110 + 001 + 100 + 000

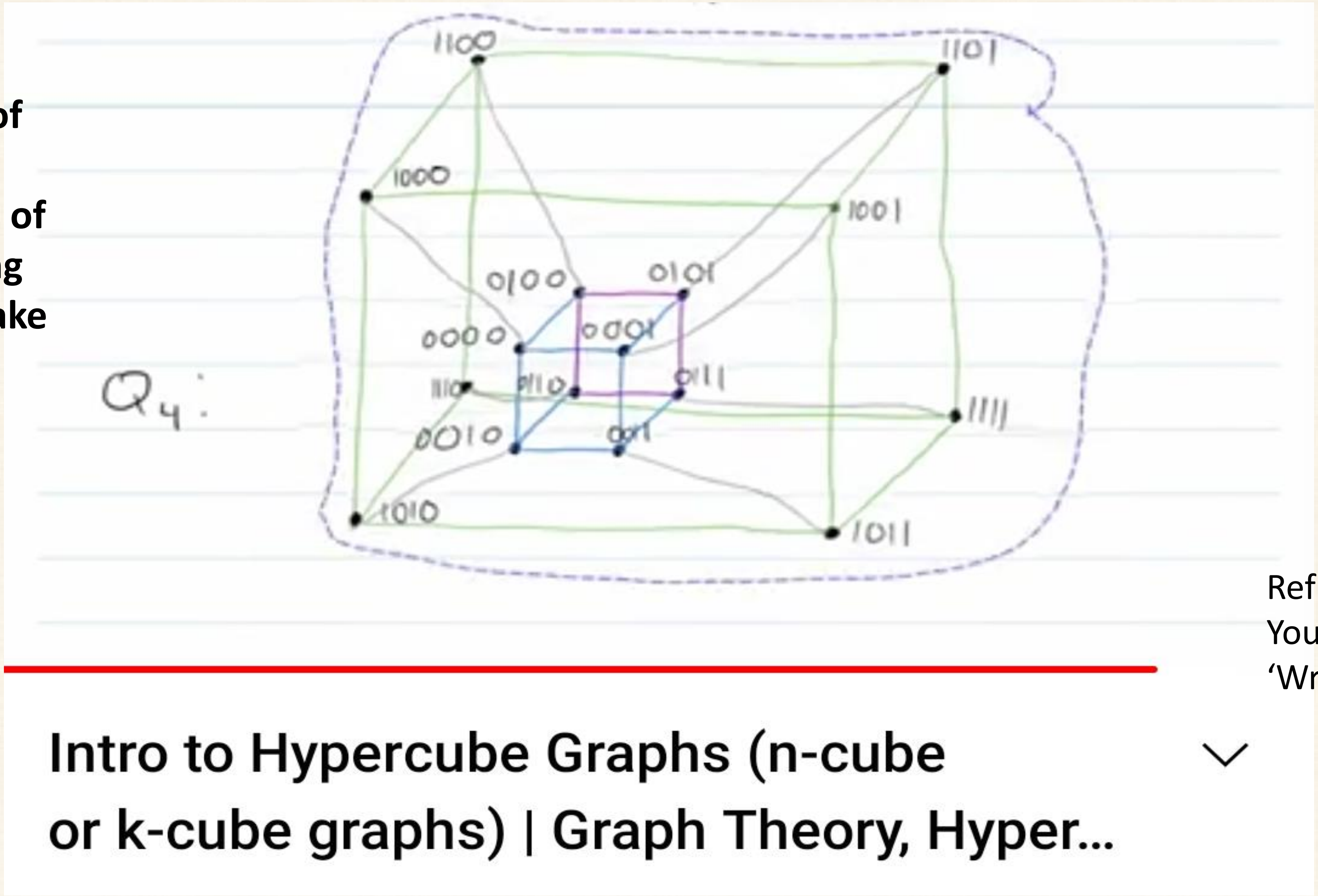
Ignore Redundant Cover of a'c and ac



Answer:
F= a'b' + b'c' + bc + ab

A Boolean n-Cube Graph for 4-variables

Add '0' at MSB of 3-Cube graph Extend all Edges of 3-Cube by adding '1' at MSB to make 4-cube graph

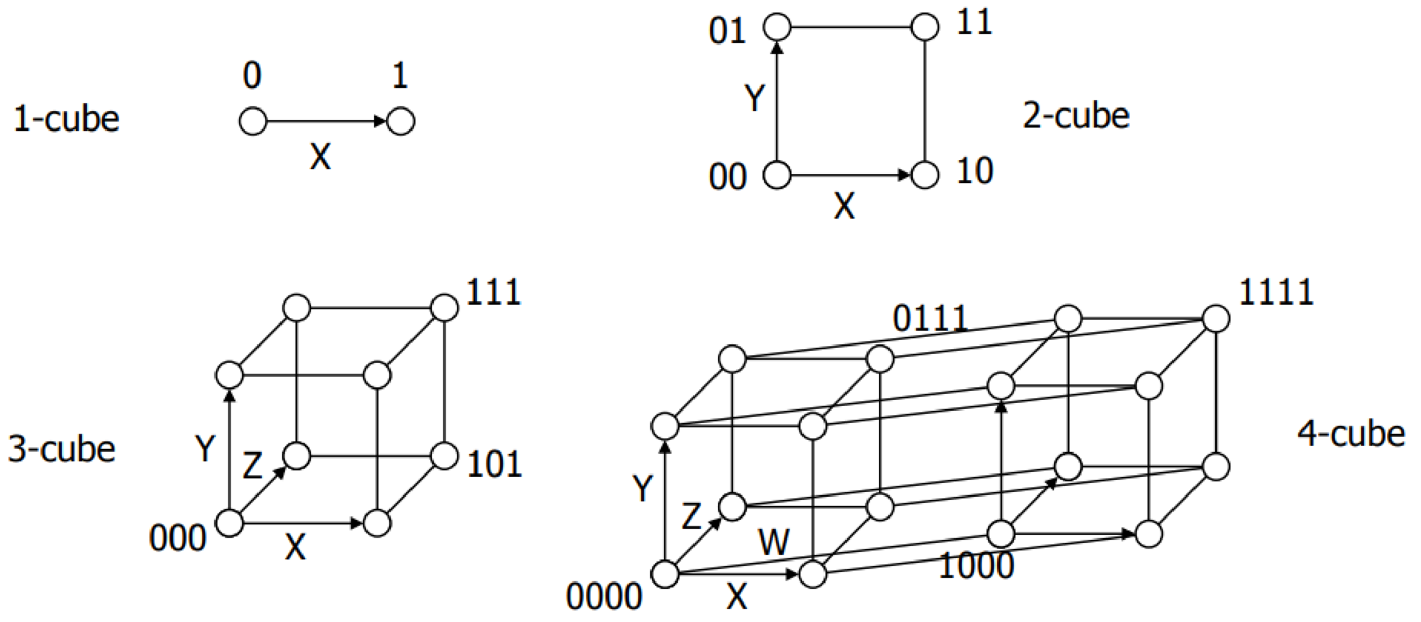


Ref:
Youtube channel
'Wrath of Math'

Graphs for 1 to 4 variables

Boolean cubes

- Visual technique for indentifying when the uniting theorem can be applied
- n input variables = n-dimensional "cube"

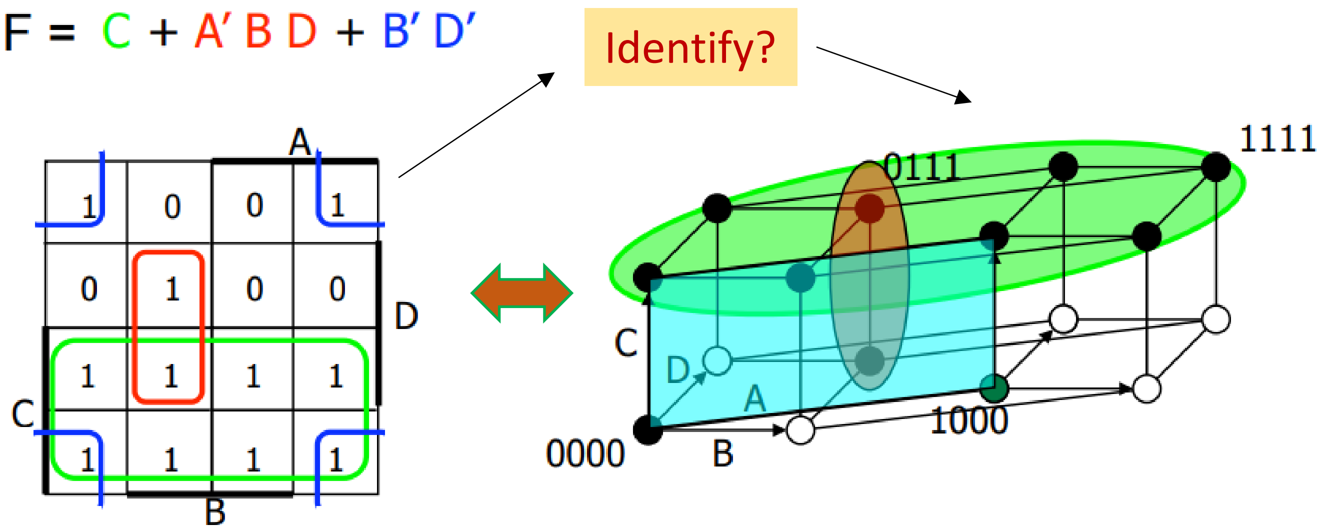


Example Graph for 4 variables

Karnaugh map: 4-variable example

- F(A,B,C,D) = Σm(0,2,3,5,6,7,8,10,11,14,15)

F = C + A′ B D + B′ D′



find the smallest number of the largest possible subcubes to cover the ON-set (fewer terms with fewer inputs per term)

Little bit about Espresso tool

Uses Graph Theory to represent Boolean variables

Finds different groupings to determine local and global minima

Keeps trying for optimal solution until some defined time-limit is reached

PyEDA library has logic minimization and synthesis tools in Python
https://pyeda.readthedocs.io/en/latest/overview.html
https://pyeda.readthedocs.io/en/latest/2llm.html

Exact techniques vs Heuristics based

The before mentioned *Karnaugh Map* and *Quine-McCluskey* process are two *Exact* techniques for Boolean minimisation. The Exact minimisation techniques require the generation of *all* possible solutions, from which the best solution is extracted. When the number of inputs to the Exact minimisation techniques become large, the number of possible solutions grows very quickly. The Exact techniques are therefore known to be extremely inefficient in logic minimisation.

Due to the inefficiencies of the Exact techniques, the computation of logic minimisation have concentrated on Heuristic solving techniques. The Heuristic techniques avoid computing all possible solutions, and successively modifies an initial solution, until a suitable stopping criterion is met. The minimisation is thus accomplished more efficiently and rapidly. However the technique is *not* guaranteed to find the best solution, but instead finds a near optimal solution (Ruddell, R., & Sangiovanni-Vincentelli, A., 1987; Theobald, M., & Norwick, S., 1998; Brayton, R., et al, 1990).

The Heuristic techniques operate by establishing decision cubes, based on the values in the input truth table. An initial partial solution is obtained, and then the nearest neighbours are examined to determine if they are also a solution. If further solutions are found, they are linked with the initial solution. The search then moves onto one of the neighbouring found solutions, and their neighbours are examined. In this way, the final solutions are obtained, by those solutions with the highest linking.

Reference: MS Thesis by Hacker, 2001

An Example of Heuristic solution

To diagrammatically explain the Heuristic procedure, the author has derived a drawing to illustrate the decision cube process. This drawing is presented in Figure 3.9.

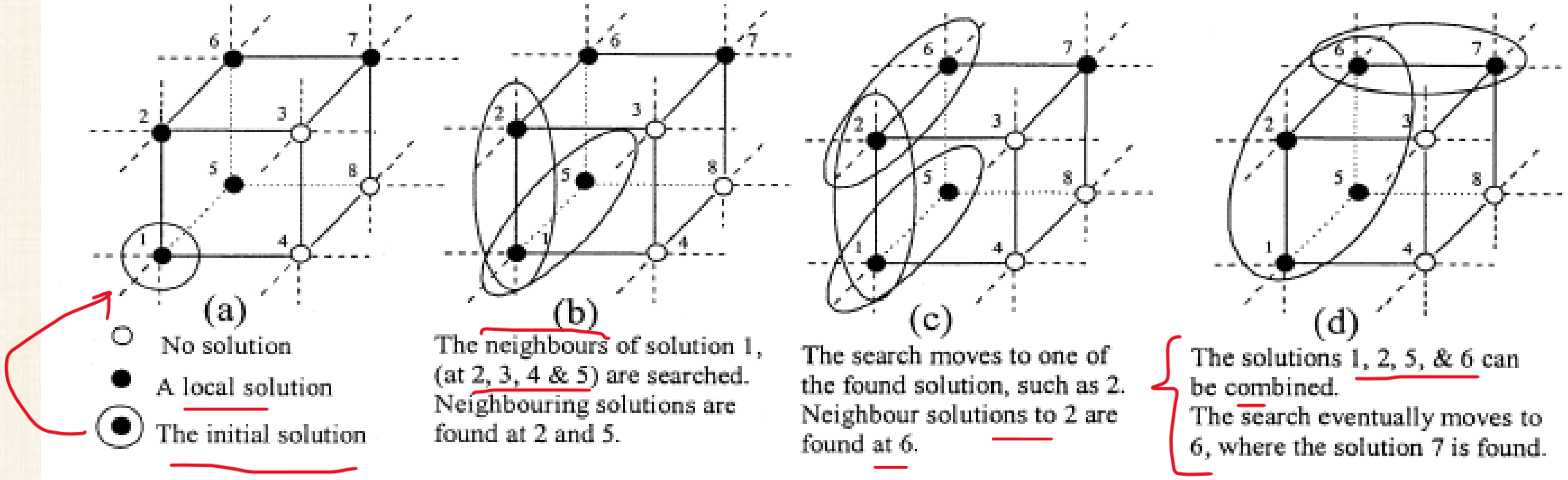
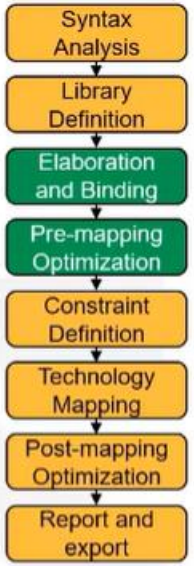


Figure 3.9: Example of the Heuristic procedure.

Reference: MS Thesis by Hacker, 2001

Espresso - 1

Espresso Heuristic Minimizer



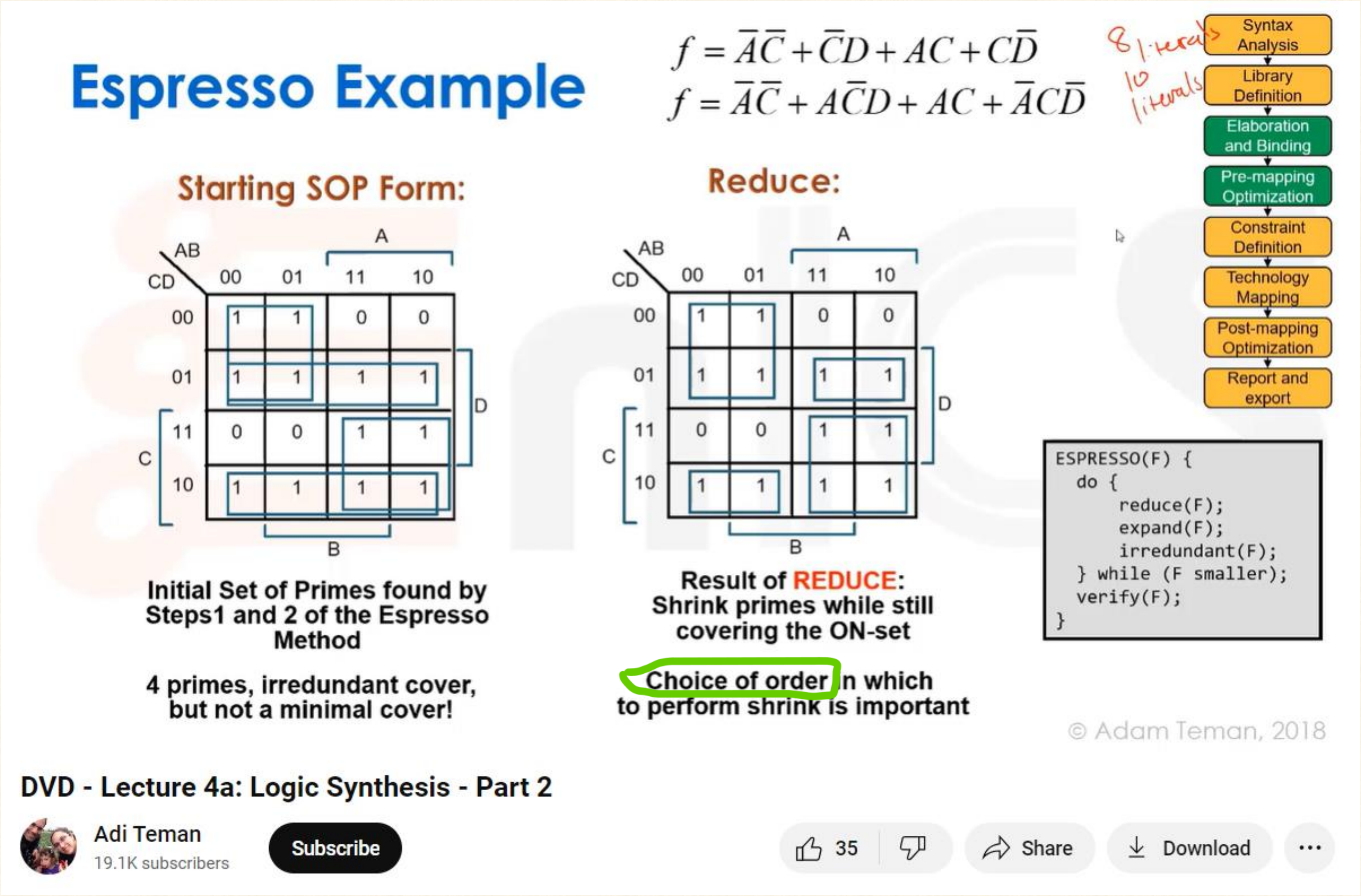

```
ESPRESSO(F) {
  do {
      reduce(F);
      expand(F);
      irredundant(F);
  } while (fewer terms in F);
  verify(F);
}
```

- **Start with an SOP solution.**
  - Expand
    - Make each cube as *large as possible* without covering a point in the OFF-set.
    - Increases the number of literals (worse solution)
  - Irredundant
    - Throw out *redundant* cubes.
    - Remove smaller cubes whose points are covered by larger cubes.
  - Reduce
    - The cubes in the cover are reduced in size.
  - In general, the new cover will be different from the initial cover.
    - "expand" and "irredundant" steps can possibly find out a new way to cover the points in the ON-set.
    - Hopefully, the new cover will be smaller.

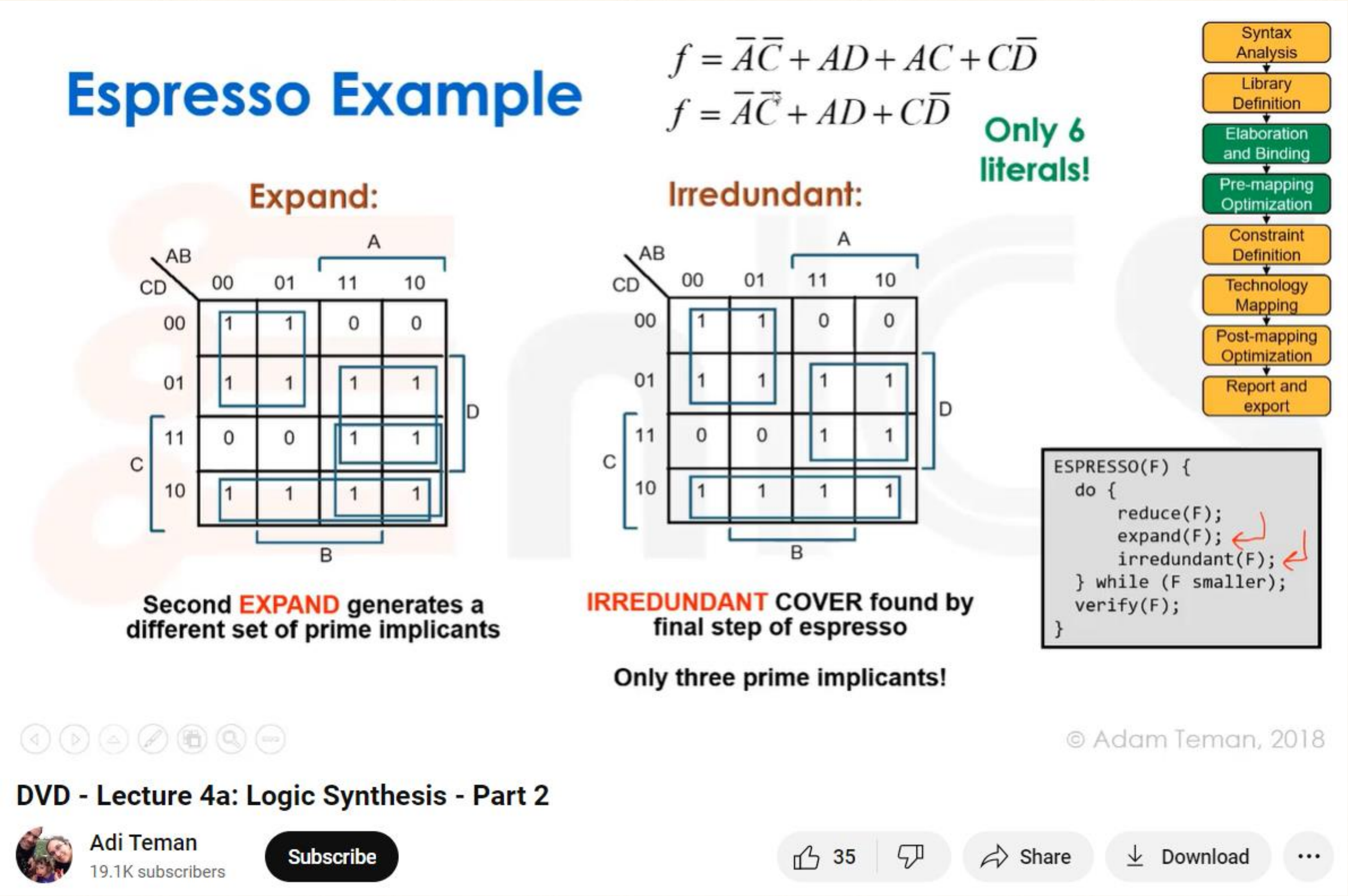
9

DVD - Lecture 4a: Logic Synthesis - Part 2

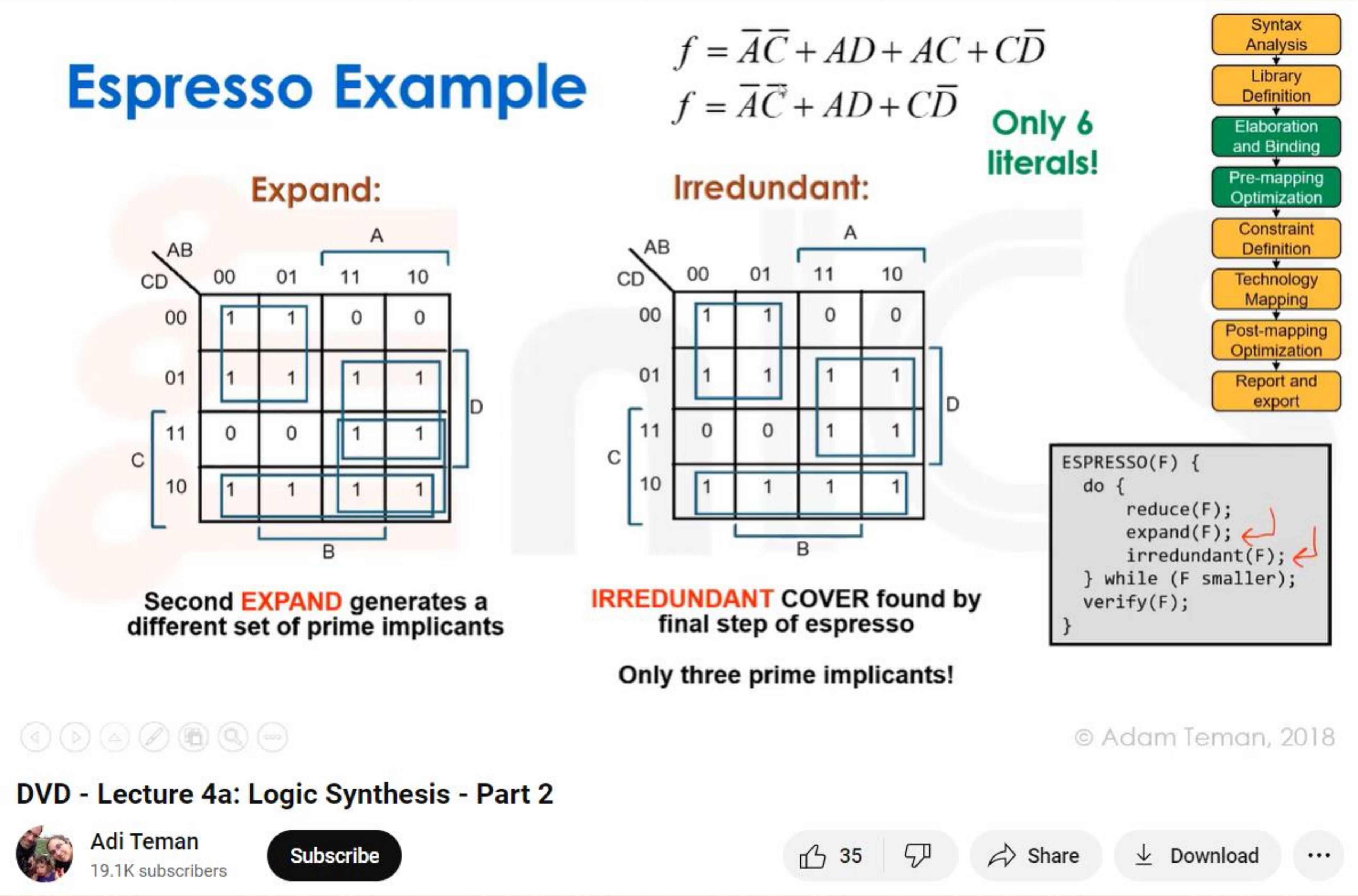
Espresso - 2


Espresso Example
$f = \bar{A}\bar{C} + \bar{C}D + AC + C\bar{D}$
$f = \bar{A}\bar{C} + A\bar{C}D + AC + \bar{A}C\bar{D}$
8 literals
10 literals
Starting SOP Form:
Reduce:
AB
CD
00
01
11
10
A
B
C
D
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
Syntax Analysis
Library Definition
Elaboration and Binding
Pre-mapping Optimization
Constraint Definition
Technology Mapping
Post-mapping Optimization
Report and export
ESPRESSO(F) {
do {
reduce(F);
expand(F);
irredundant(F);
} while (F smaller);
verify(F);
}
© Adam Teman, 2018
DVD - Lecture 4a: Logic Synthesis - Part 2
Adi Teman
19.1K subscribers
Subscribe
35
Share
Download

Espresso - 3


Espresso Example
$f = \bar{A}\bar{C} + AD + AC + C\bar{D}$
$f = \bar{A}\bar{C} + AD + C\bar{D}$
Only 6 literals!
Expand:
Irredundant:
AB
CD
A
B
C
D
00
01
11
10
Second EXPAND generates a different set of prime implicants
IRREDUNDANT COVER found by final step of espresso
Only three prime implicants!
Syntax Analysis
Library Definition
Elaboration and Binding
Pre-mapping Optimization
Constraint Definition
Technology Mapping
Post-mapping Optimization
Report and export
ESPRESSO(F) {
do {
reduce(F);
expand(F);
irredundant(F);
} while (F smaller);
verify(F);
}
© Adam Teman, 2018
DVD - Lecture 4a: Logic Synthesis - Part 2
Adi Teman
19.1K subscribers
Subscribe
35
Share
Download

Espresso - 4


Espresso Example
f = A̅C̅ + AD + AC + CD̅
f = A̅C̅ + AD + CD̅
Only 6 literals!
Expand:
Irredundant:
Second EXPAND generates a different set of prime implicants
IRREDUNDANT COVER found by final step of espresso
Only three prime implicants!
Syntax Analysis
Library Definition
Elaboration and Binding
Pre-mapping Optimization
Constraint Definition
Technology Mapping
Post-mapping Optimization
Report and export
ESPRESSO(F) {
do {
reduce(F);
expand(F);
irredundant(F);
} while (F smaller);
verify(F);
}
© Adam Teman, 2018
DVD - Lecture 4a: Logic Synthesis - Part 2
Adi Teman
19.1K subscribers
Subscribe
35
Share
Download

Learn More on Espresso

- DVD - Lecture 4a: Logic Synthesis - Part 2 (youtube.com)
- https://www.youtube.com/watch?v=yJ5CaAk7Nq8&list=WL&index=1
- Lec 13: ESPRESSO-Heuristic Based Switching Function Minimization (youtube.com)
- https://www.youtube.com/watch?v=NRAoJ8eKIgM