Two-Level Logic Optimization

(4541.554 Introduction to Computer-Aided Design)

School of EECS

Seoul National University

Minimization of Two-Level Functions

- **Goals:**
  - **Minimize cover cardinality**
  - **Minimize number of literals**

  PLA implementation:
  - **Minimize number of rows**
  - **Minimize number of transistors**

  --> Minimize area and time
- **Karnaugh map: manual minimization**

f=Σm(0,2,3,6,7,8,9,10,13)

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

f=b’d’+a’c+ac’d

- **Quine-McCluskey Method**
  - **Exact minimization (global minimum)**
  - **Generate all prime implicants**
    - **Start from 0-dimensional cubes (minterms)**
    - **Find k-dimensional cubes from (k-1)-dimensional cubes (find cubes that are different in only one position)**
  - **Find a minimum prime cover**
    - **Minimum set of prime implicants covering all the minterms**

1's	Minterms	
0	m_0	0000 √
1	m_2	0010 √
	m_8	1000 √
2	m_3	0011 √
	m_6	0110 √
	m_9	1001 √
	m_{10}	1010 √
3	m_7	0111 √
	m_{13}	1101 √

1-Cubes	
0,2	00x0 √
0,8	x000 √
2,3	001x √
2,6	0x10 √
2,10	x010 √
8,9	100x *
8,10	10x0 √
3,7	0x11 √
6,7	011x √
9,13	1x01*

2-Cubes	
0,2,8,10	x0x0 *
2,3,6,7	0x1x *

essential

		0	2	3	6	7	8	9	10	13
*	0,2,8,10	√	√				√		√	
*	2,3,6,7		√	√	√	√				
	8,9						√	√		
*	9,13							√		√
		√	√	√	√	√	√	√	√	√

$$f = m_{0,2,8,10} + m_{2,3,6,7} + m_{9,13}$$

$$= b'd' + a'c + ac'd$$

- **Petrick's method**

	6	7	15	38	46	47	
a	√	√					00011x
b			√			√	x01111
c		√	√				00x111
d					√	√	10111x
e				√	√		10x110
f	√			√			x00110

"covering proposition"

CP=(a+f)(a+c)(b+c)(e+f)(d+e)(b+d)

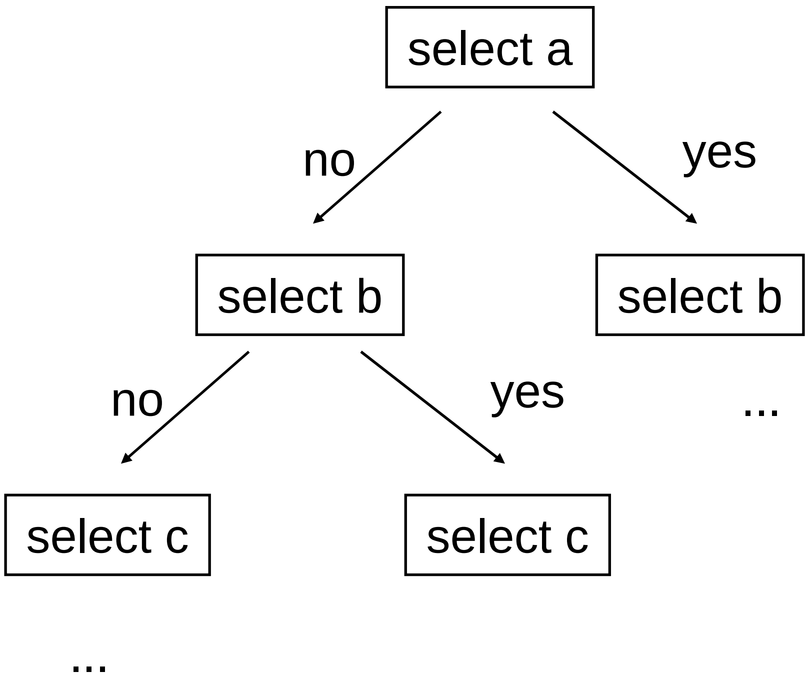
$m_6 \quad m_7 \quad m_{15} \quad m_{38} \quad m_{46} \quad m_{47}$

=abe+abdf+acde+bcef+cdf (exponential)

select terms with fewest product terms (abe, cdf)

---> then select the term with fewest literals

- **Branching method**



If worse, prune the branch

- **Complexity:**
  - **Number of minterms: ~ 2^n**
  - **Number of prime implicants: ~ $3^n/n$**
  - **Very large table**
  - **Covering problem is NP-complete --> Branch and bound technique**
  - **Use heuristic minimization**
    - **Find a minimal cover using iterative improvement**
    - **Example: MINI, PRESTO, ESPRESSO**

MINI

- S.J.Hong, R.G. Cain, and D.L.Ostapco, "MINI: a heuristic approach for logic minimization," IBM J. of Res. and Dev., Sep. 1974.
- Three processes
  - Expansion: Expand implicants and remove those that are covered
  - Reduction: Reduce implicants to minimal size
  - Reshape: Modify implicants

- **Expansion**
  - **Iterate on implicants**
  - **Make implicants as large as possible**
  - **Remove covered implicants**
  - **Goal: Minimal prime cover w.r.t. single cube containment**
  - **Most minimizers use this technique**
  - **Algorithm:**

    For each implicant {

    For each care literal {

    Replace literal by *

    If (implicant not valid) restore

    }

    Remove all implicants covered by expanded implicants

    }
  - **Validity check:**
    - **Check intersection of expanded implicant with X^{OFF}**

– **Example**

0	0	0	1
1	0	0	1
0	1	0	1
0	0	1	1
1	0	1	0
0	1	1	0
1	1	1	0
1	1	0	*

Take 0 0 0
Expand * 0 0 ok
Expand * * 0 ok
Expand * * * not ok
Restore * * 0
Remove covered implicants
* * 0
0 0 1

Take 0 0 1
Expand * 0 1 not ok
Restore 0 0 1
Expand 0 * 1 not ok
Restore 0 0 1
Expand 0 0 * ok
Remove covered implicants
* * 0
0 0 *

- **Reduce**
  - **Iterate on implicants**
  - **Reduce implicant size while preserving cover cardinality**
  - **Goal: Escape from local minima**
  - **Alternate with expansion**
  - **Algorithm:**

```
For each implicant {
    For each don't care literal {
        Replace literal by 1 or 0
        If (implicant not valid) restore
    }
}
```

  - **Heuristics: ordering**
  - **Example:**

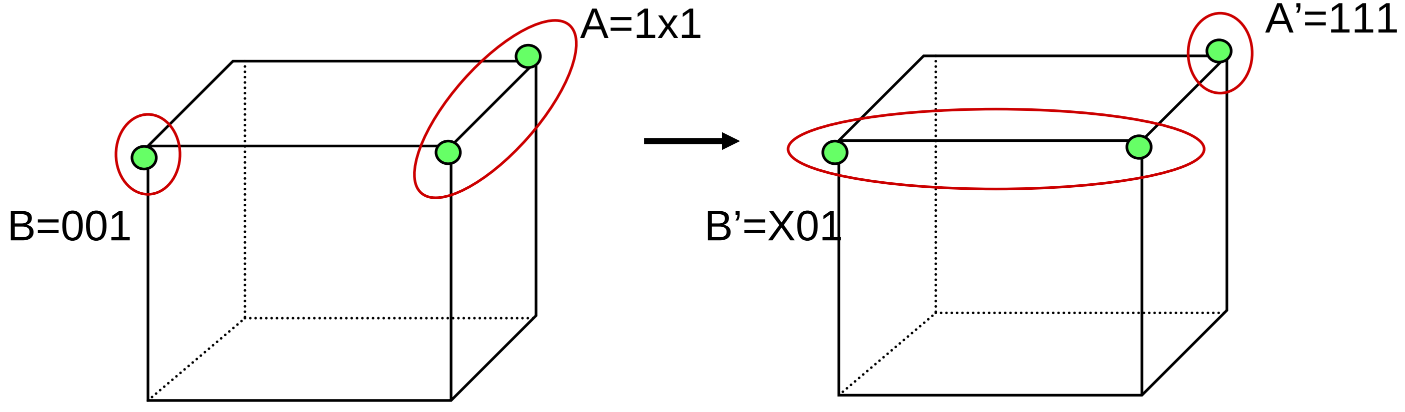
```
* * 0        can't be reduced
0 0 *        can be reduced to 0 0 1
```

- **Reshape**
  - **Modify implicants while preserving cover cardinality**
  - **Goal: Escape from local minima**

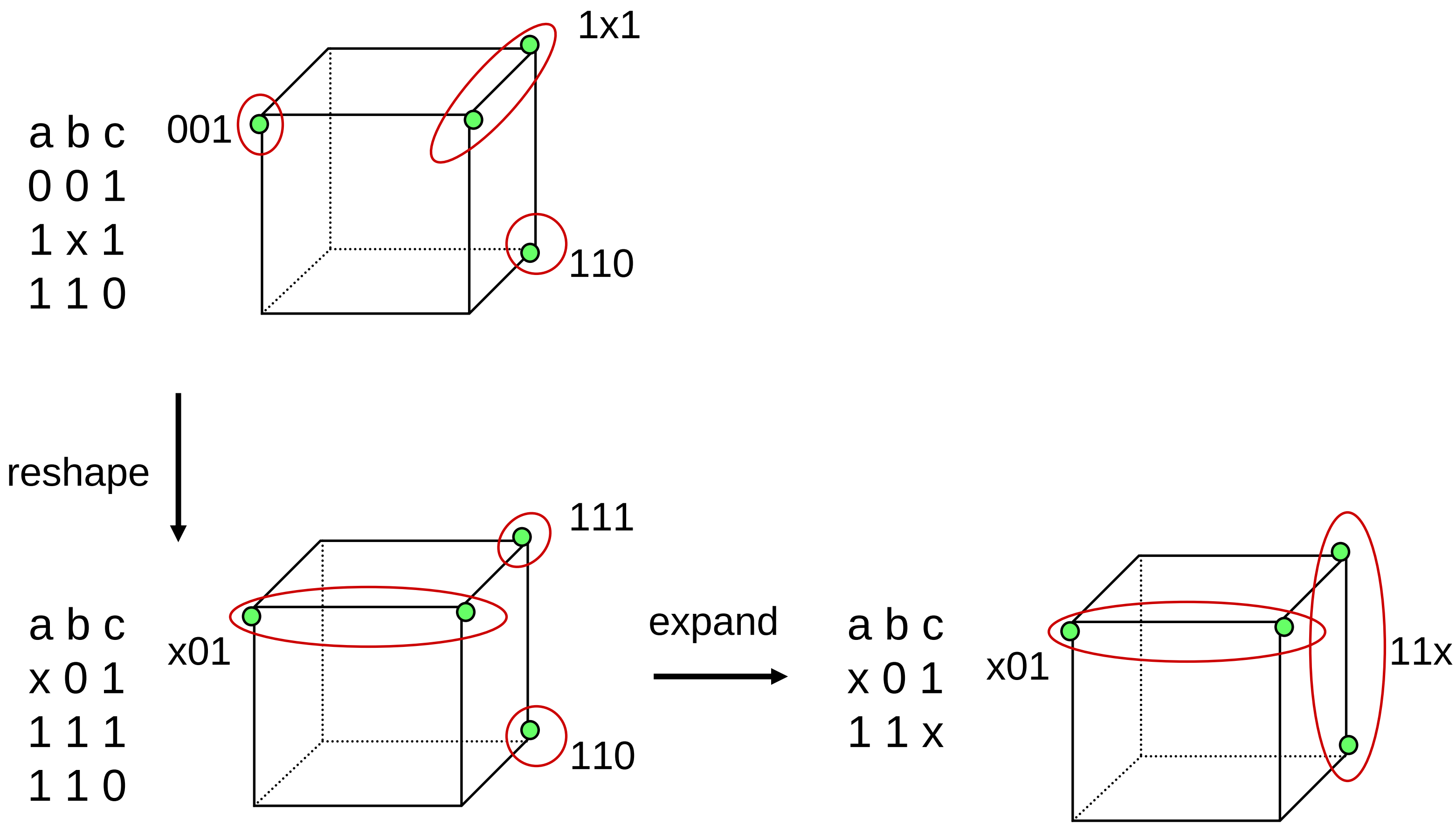
A, B : disjoint
A and B are different in exactly two parts
One different part of A covers the corresponding part of B

$A=\pi_1\ \pi_2\ \ldots\ \pi_i\ \ldots\ \pi_j\ \ldots\ \pi_p$
$B=\pi_1\ \pi_2\ \ldots\ \mu_i\ \ldots\ \mu_j\ \ldots\ \pi_p$
π_j covers μ_j

$A'=\pi_1\ \pi_2\ \ldots\ \pi_i\ \ldots\ (\pi_j \cap \mu_j')\ \ldots\ \pi_p$
$B'=\pi_1\ \pi_2\ \ldots\ (\pi_i \cup \mu_i)\ \ldots\ \mu_j\ \ldots\ \pi_p$



– **Alternate with expansion and reduce.**
– **Example:**



Espresso II

- R.K.Brayton, G.D.Hachtel, C.T.McMullen, and A.L.Sangiovanni-Vincentelli, *Logic Minimization Algorithms for VLSI Synthesis*, Kluwer Academic Publishers, 1984.
- Results are often global minimum.
- Very fast

- **Sequence of operations**

1. Complement
   - **Compute the off-set (complement of X^{ON} U X^{DC})**
2. **Expand**
   - **Expand each implicant into a prime and remove covered implicants**
3. **Essential primes**
   - **Extract essential primes and put them in the don't care set**
4. **Irredundant cover**
   - **Find a minimal irredundant cover**
5. **Reduce**
   - **Reduce each implicant to a minimum essential implicant**
6. **Iterate 2, 4, and 5 until no improvement**
7. **Lastgasp**
   - **Try reduce, expand, and irredundant cover using a different strategy**
   - **If successful, continue the iteration**
8. **Makesparse**
   - **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

- **Complementation**
  - **Recursive computation**

$$
\begin{aligned}
F' &= (x_j F_{x_j} + x_j' F_{x_j'})' \\
&= (x_j F_{x_j})' (x_j' F_{x_j'})' \\
&= (x_j' + (F_{x_j})')(x_j + (F_{x_j'})') \\
&= x_j'(F_{x_j'})' + x_j(F_{x_j})' + (F_{x_j})'(F_{x_j'})' \\
&= x_j'(F_{x_j'})' + x_j(F_{x_j})' + x_j'(F_{x_j})'(F_{x_j'})' + x_j(F_{x_j})'(F_{x_j'})' \\
&= x_j'(F_{x_j'})' + x_j(F_{x_j})'
\end{aligned}
$$

  - **Computation of $(F_{x_j'})'$ and $(F_{x_j})'$:**
    - **$|F_{x_j'}| =< |F|$ (cubes with xj are removed)**
    - **$|F_{x_j}| =< |F|$ (cubes with xj′ are removed)**
    - **One less variable (xj is removed)**
  - **If the cubes have variables xj only for j=1,...,k,**

    then $F_{x_1 x_2 \ldots x_k}$ is tautology and the complement is empty.
  - **Choice of variables:**
    - **Choose variables of the largest cube (with least # of literals) --> terminate the recursion fast**
    - **In the cube, choose first the variable that appears most often in the other cubes of F --> remove as many literals as possible**

- **Sequence of operations**
  1. **Complement**
     - **Compute the off-set (complement of X^{ON} U X^{DC})**
  2. Expand
     - **Expand each implicant into a prime and remove covered implicants**
  3. **Essential primes**
     - **Extract essential primes and put them in the don't care set**
  4. **Irredundant cover**
     - **Find a minimal irredundant cover**
  5. **Reduce**
     - **Reduce each implicant to a minimum essential implicant**
  6. **Iterate 2, 4, and 5 until no improvement**
  7. **Lastgasp**
     - **Try reduce, expand, and irredundant cover using a different strategy**
     - **If successful, continue the iteration**
  8. **Makesparse**
     - **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

- **Expand**
  - **Ordering cubes in the cover:**
    - **Arrange cubes in the order of decreasing size.**

      --> Larger cubes are more likely to cover other cubes and less likely to be covered by other cubes.
  - **Selecting columns in a cube (c):**

    1. If a cube (d) in the off-set has only one column (j) that satisfies the following condition,

    (c_j=1 and d_j=0) or (c_j=0 and d_j=1)

    column j cannot be expanded.

    ex) c=01*, d=111

    --> Column 1 of c cannot be expanded.

    Reduce problem size by eliminating the excluded column, d, and cubes that cannot be covered.

2. Select columns that can be expanded to cover as many cubes in the on-set as possible.

   If there are no more covered cubes, select a column with maximum conflicts between c and other cubes in the on-set.

   The corresponding columns and covered cubes are eliminated.

3. If all cubes in the off-set have no conflict on column j, select the column for expansion. The corresponding column and covered cubes are eliminated.

4. Repeat step 1, 2, and 3 until

   (1) all columns are eliminated

   --> done

   (2) all cubes in the off-set are eliminated

   --> select all remaining columns for expansion

   (3) all cubes in the on-set are covered

   --> select for expansion as many columns as possible, to reduce # of literals

   --> find the minimum column cover of the Blocking Matrix

   --> NP-complete

   --> use heuristics

- **Blocking Matrix**

  on-set: c=10101

  off-set: r1=11*00

  r2=10111

  r3=01101

  r4=11000

$$\begin{array}{c} r1 \\ r2 \\ r3 \\ r4 \end{array} \begin{bmatrix} 01001 \\ 00010 \\ 11000 \\ 01101 \end{bmatrix}$$

1 when $c_j \not\subseteq ri_j$

If c is not expanded for these columns
--> $\{c\} \cap R=\varnothing$

- **Sequence of operations**

1. Complement

- **Compute the off-set (complement of X^{ON} U X^{DC})**

2. Expand

- **Expand each implicant into a prime and remove covered implicants**

3. Essential primes

- **Extract essential primes and put them in the don't care set**

4. Irredundant cover

- **Find a minimal irredundant cover**

5. Reduce

- **Reduce each implicant to a minimum essential implicant**

6. Iterate 2, 4, and 5 until no improvement

7. Lastgasp

- **Try reduce, expand, and irredundant cover using a different strategy**
- **If successful, continue the iteration**

8. Makesparse

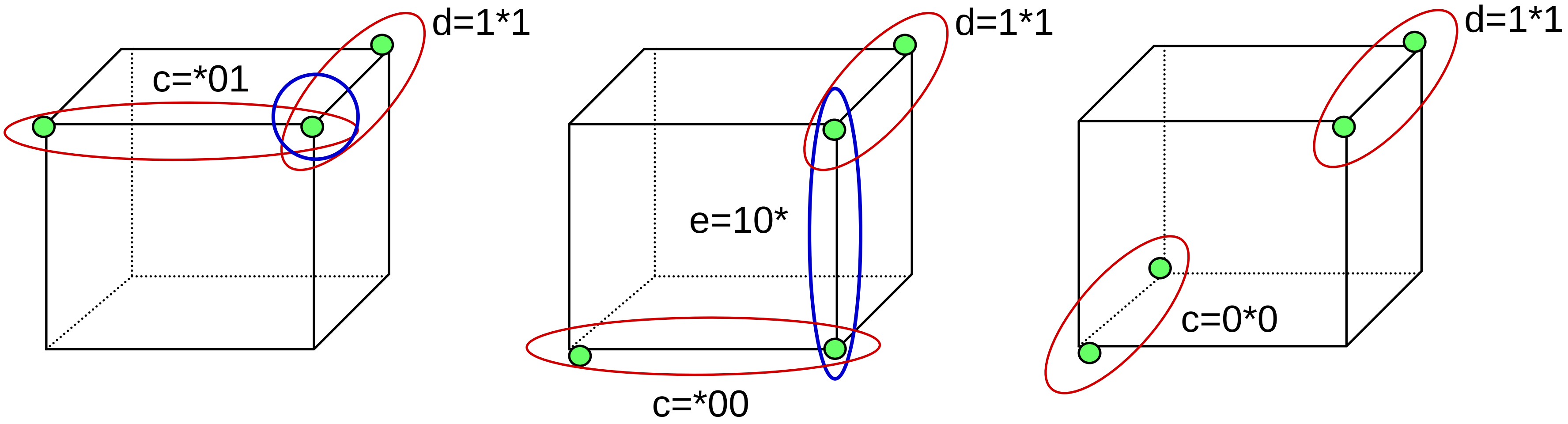
- **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

- **Essential primes**
  - **Consensus e of two cubes c and d is a cube such that:**

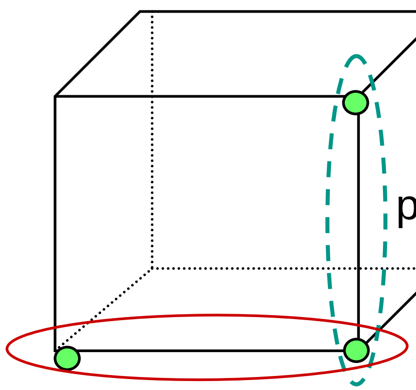
    If d(c, d) = 0, then $e = c \cap d$

    if d(c, d) = 1, then $e_i = \begin{cases} c_i d_i, & c_i d_i \neq \varnothing \\ *, & \text{otherwise} \end{cases}$

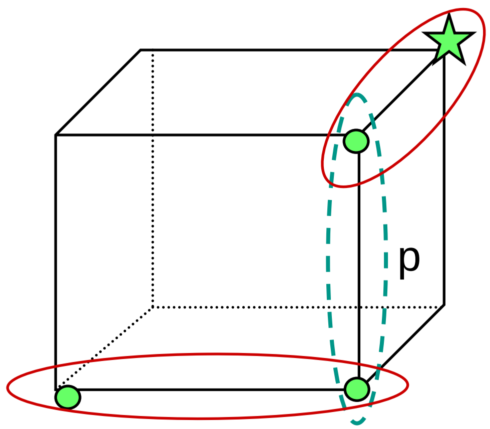
    if d(c, d) ≥ 2, then $e = \varnothing$



– **For a prime p, iff the consensus of ((on-set ∪ dc-set)-{p}) with p completely covers p, p is not essential.**

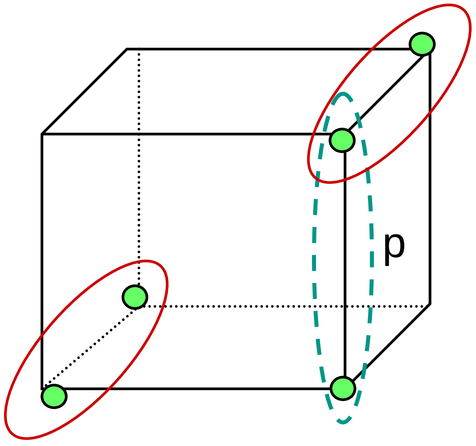


essential



not essential

– **Why take consensus?**

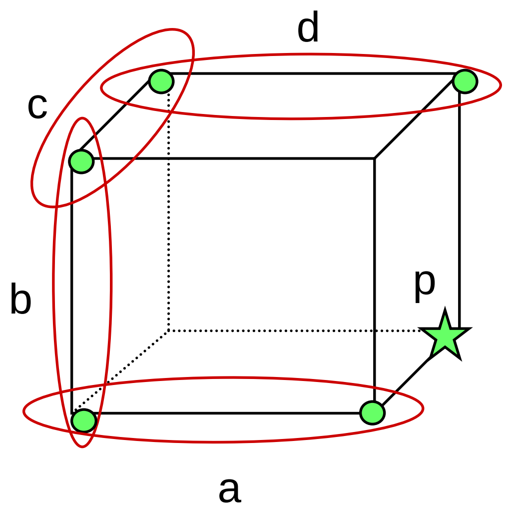


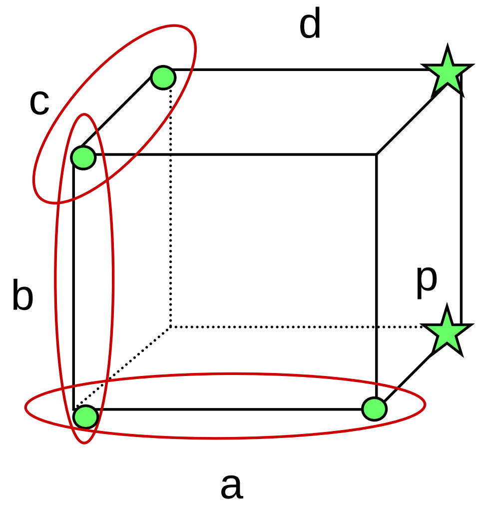
((on-set ∪ dc-set)-{p}) does not cover p
even though p is not essential

- **Sequence of operations**
  1. **Complement**
     - **Compute the off-set (complement of X^{ON} U X^{DC})**
  2. **Expand**
     - **Expand each implicant into a prime and remove covered implicants**
  3. **Essential primes**
     - **Extract essential primes and put them in the don't care set**
  4. **Irredundant cover**
     - **Find a minimal irredundant cover**
  5. **Reduce**
     - **Reduce each implicant to a minimum essential implicant**
  6. **Iterate 2, 4, and 5 until no improvement**
  7. **Lastgasp**
     - **Try reduce, expand, and irredundant cover using a different strategy**
     - **If successful, continue the iteration**
  8. **Makesparse**
     - **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

- **Irredundant cover**
  - **Partition the prime cover into two sets:**
    - **set E of relatively essential cubes**
    - **set R of redundant cubes.**
  - **For a cube c in the (on-set), if ((on-set $\cup$ dc-set) - {c}) covers c, then c is a redundant cube (c $\in$ R), else c is a relatively essential cube (c $\in$ E).**
  - **A redundant cube r is partially redundant if (dc-set $\cup$ E) does not cover r.**
  - **Remaining cubes in R are totally redundant.**
  - **Totally redundant cubes are removed.**
  - **From the set R_p of partially redundant cubes, extract a minimal set R_c such that E $\cup$ R_c is still a cover.**

    --> minimum column cover





a, d : relatively essential
b, c : partially redundant

a, c : relatively essential
b : totally redundant

- **Sequence of operations**
  1. **Complement**
     - **Compute the off-set (complement of X^{ON} U X^{DC})**
  2. **Expand**
     - **Expand each implicant into a prime and remove covered implicants**
  3. **Essential primes**
     - **Extract essential primes and put them in the don't care set**
  4. **Irredundant cover**
     - **Find a minimal irredundant cover**
  5. **Reduce**
     - **Reduce each implicant to a minimum essential implicant**
  6. **Iterate 2, 4, and 5 until no improvement**
  7. **Lastgasp**
     - **Try reduce, expand, and irredundant cover using a different strategy**
     - **If successful, continue the iteration**
  8. **Makesparse**
     - **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

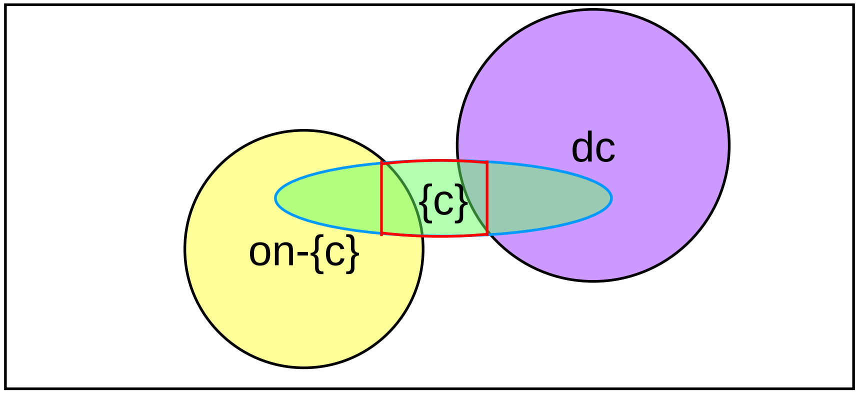
- **Reduction**
  - **Ordering cubes for reduction:**
    - **Select the largest cube**

      --> Largest cubes can be reduced most easily.
    - **Order the remaining cubes in increasing pseudo-distance (number of mismatches)**

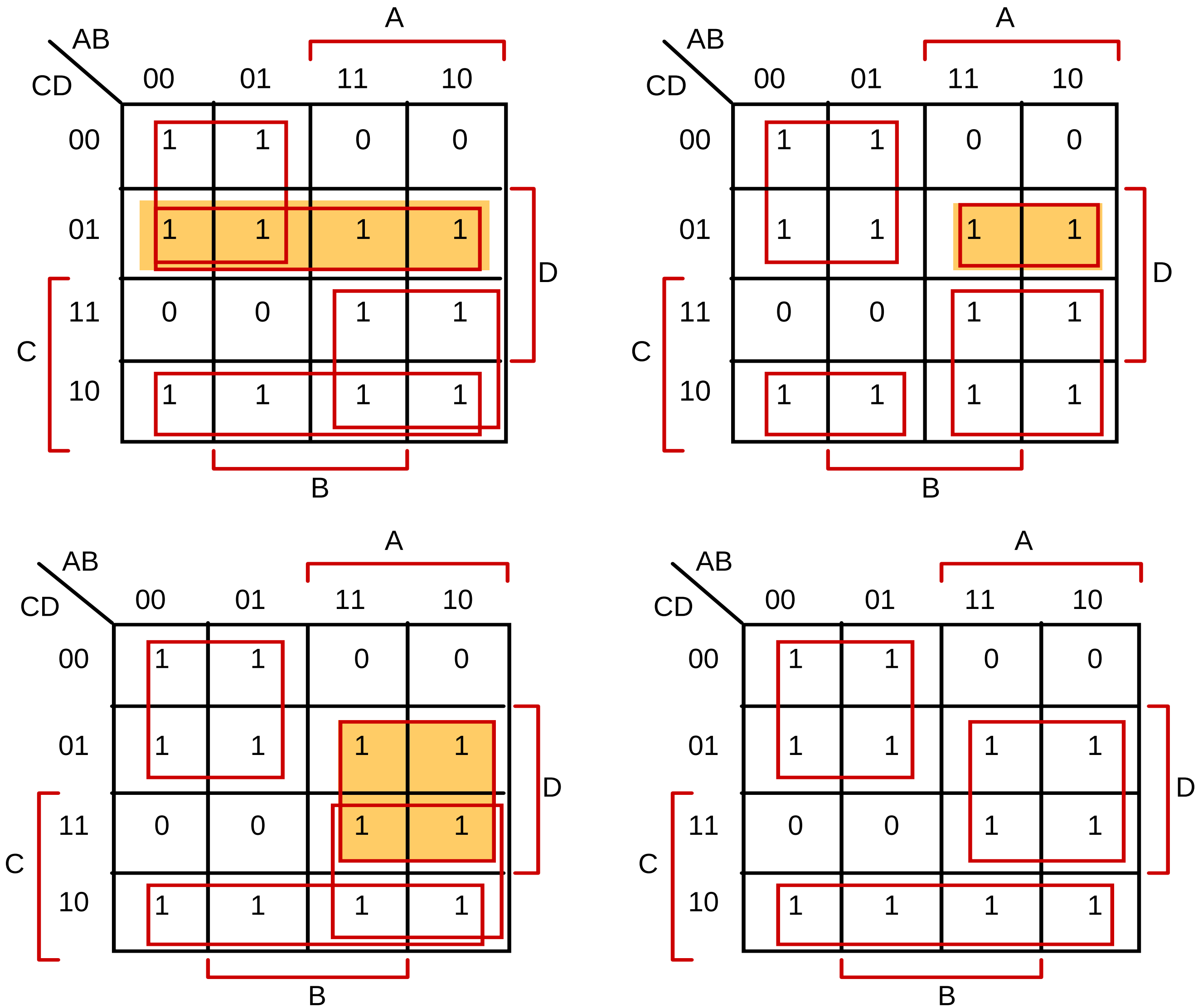
      ex) pd(01*1, 0*11) = 2

      --> Later expansion easily covers its neighbors.
  - **For a cube c in the cover, compute the smallest cube s containing**

    c ((on-set - {c}) ∪ dc-set)′ = c (F(c))′



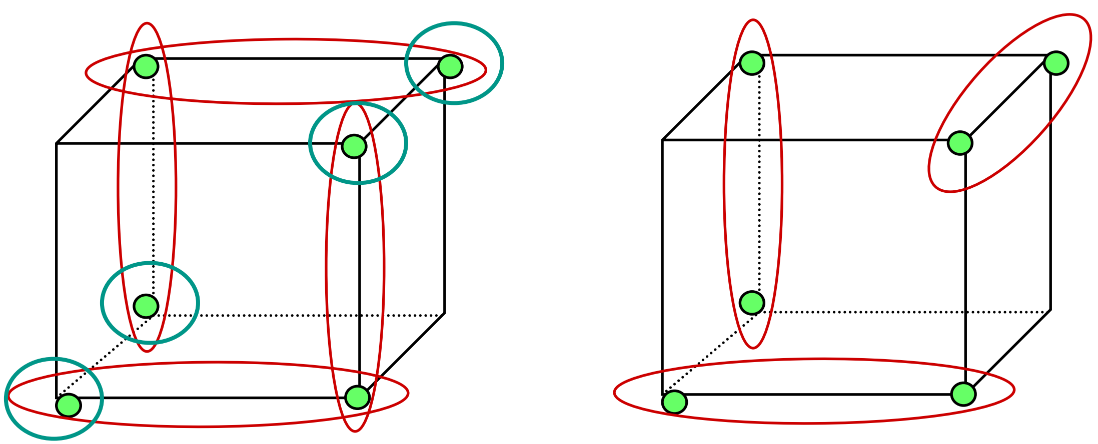
- **Example**


A
AB
CD
00 01 11 10
00 1 1 0 0
01 1 1 1 1
11 0 0 1 1
10 1 1 1 1
C
D
B
A
AB
CD
00 01 11 10
00 1 1 0 0
01 1 1 1 1
11 0 0 1 1
10 1 1 1 1
C
D
B
A
AB
CD
00 01 11 10
00 1 1 0 0
01 1 1 1 1
11 0 0 1 1
10 1 1 1 1
C
D
B
A
AB
CD
00 01 11 10
00 1 1 0 0
01 1 1 1 1
11 0 0 1 1
10 1 1 1 1
C
D
B

- **Sequence of operations**
  1. **Complement**
     - **Compute the off-set (complement of X^{ON} U X^{DC})**
  2. **Expand**
     - **Expand each implicant into a prime and remove covered implicants**
  3. **Essential primes**
     - **Extract essential primes and put them in the don't care set**
  4. **Irredundant cover**
     - **Find a minimal irredundant cover**
  5. **Reduce**
     - **Reduce each implicant to a minimum essential implicant**
  6. **Iterate 2, 4, and 5 until no improvement**
  7. **Lastgasp**
     - **Try reduce, expand, and irredundant cover using a different strategy**
     - **If successful, continue the iteration**
  8. **Makesparse**
     - **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

- **Lastgasp**
  - **Reduce cubes independently (order independent)**

    The set of reduced cubes {c1, c2, ... cp} is not necessarily a cover.
  - **Expand the reduced cubes to generate a set of new primes (NEW_PR)**
  - **Run IRREDUNDANT_COVER with NEW_PR ∪ OLD_PR, where OLD_PR is the set of old primes**



- **Sequence of operations**
  1. **Complement**
     - **Compute the off-set (complement of X^{ON} U X^{DC})**
  2. **Expand**
     - **Expand each implicant into a prime and remove covered implicants**
  3. **Essential primes**
     - **Extract essential primes and put them in the don't care set**
  4. **Irredundant cover**
     - **Find a minimal irredundant cover**
  5. **Reduce**
     - **Reduce each implicant to a minimum essential implicant**
  6. **Iterate 2, 4, and 5 until no improvement**
  7. **Lastgasp**
     - **Try reduce, expand, and irredundant cover using a different strategy**
     - **If successful, continue the iteration**
  8. **Makesparse**
     - **Include the essential primes back into the cover and make the PLA structure as sparse as possible**

- **Makesparse**
  - **Reduce number of literals in each cube**

    --> Make PLA matrix as sparse as possible

    --> Enhance ability to be folded and improve electrical properties
  - **Irredundant cover for multiple output can have redundant cubes for single output.**

    (ex)

input	output
01*	010
011	110
001	101

    --> 2nd cube is redundant for 2nd output.
  - **Lower output part:**
    - **make output plane sparse**
    - **compute irredundant cover for single output**

    (ex)

input	output
01*	010
011	110 → 100
001	101

- Raise input part
  - Make input plane sparse
  - For each cube that has been changed, perform expand operation on input part

(ex) input output

01* 010
011 100
001 101

→

01* 010
0*1 100
001 101

→

01* 010
0*1 100
001 001