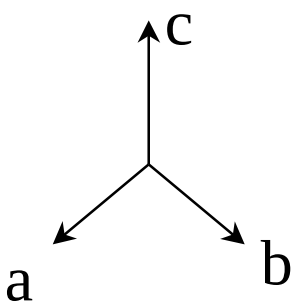
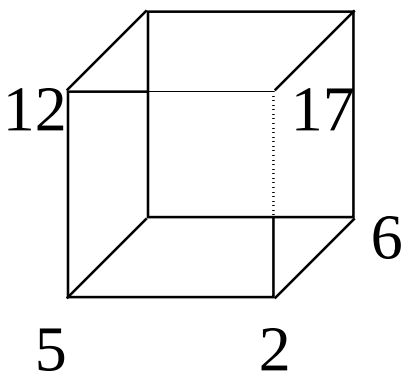
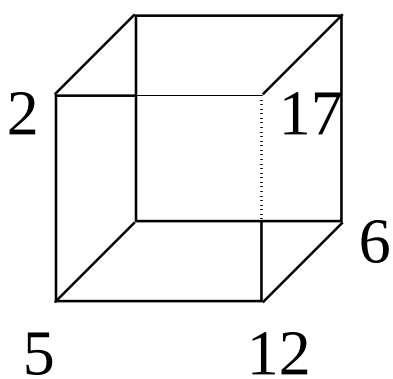
State Assignment

Hybercube Embedding



state groups :
{2,5,12,17}
{2,6,17}

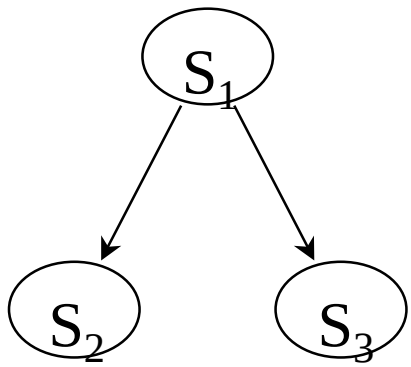




wrong!

Fanin-Oriented (exam next state pair)

- The same present state causes transition to next state pair.

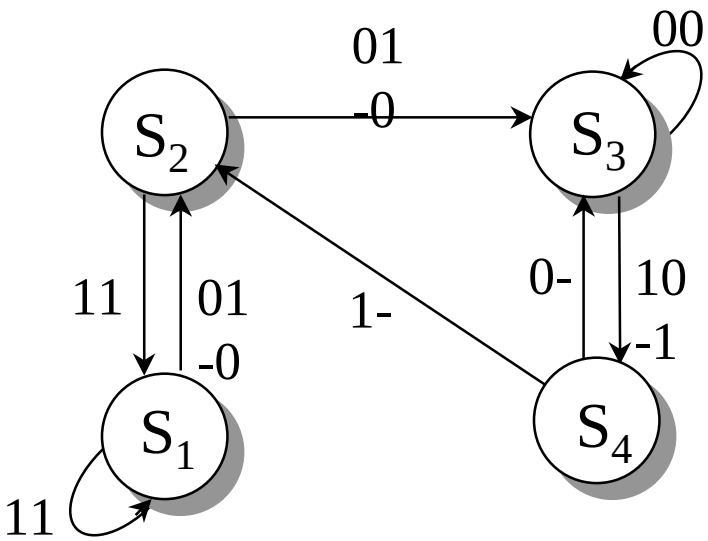


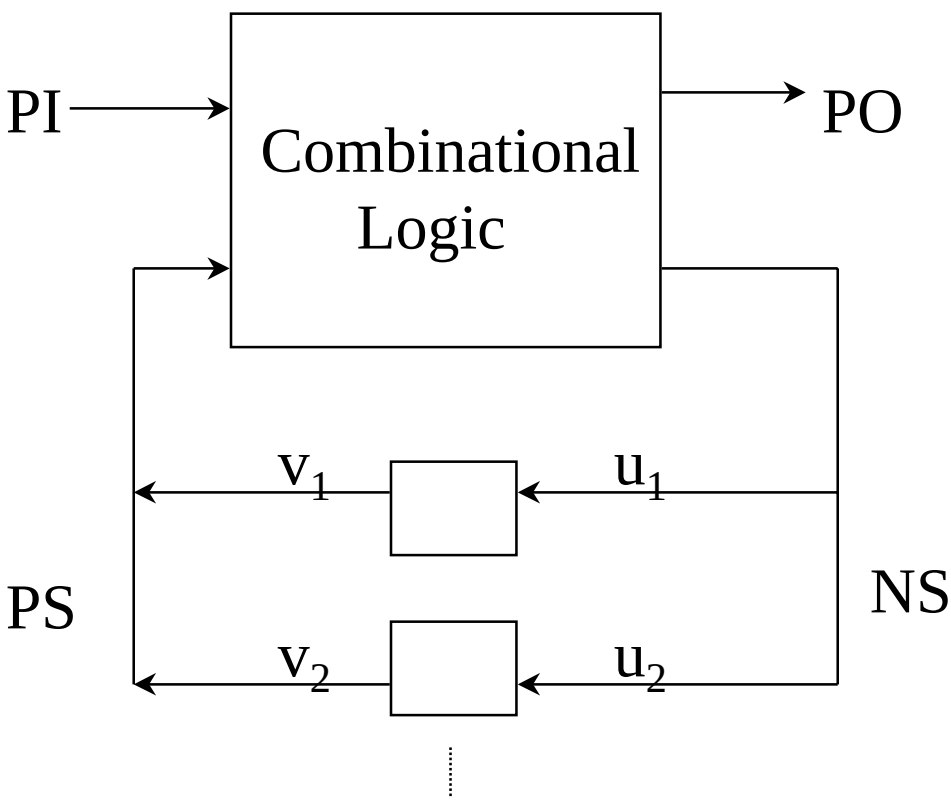
$\$\$\$ \quad S_1 \quad S_2 \quad \$\$\$\$$

$\$\$\$ \quad S_1 \quad S_4 \quad \$\$\$\$$

Add n/2 to $w(S_2,S_4)$ because of S_1

FSM Optimization





Hypercube Embedding Technique

- Observation : one -hot encoding is the easiest to decode

  Am I in state 2,5,12 or 17?

  binary : $x_4'x_3'x_2'x_1x_0'$(00010) +

  $x_4'x_3'x_2x_1'x_0$ (00101) +

  $x_4'x_3x_2x_1'x_0'$(01100) +

  $x_4x_3'x_2'$x1'x0 (10001)

  one hot : $x_2+x_5+x_{12}+x_{17}$

  But one hot uses too many flip flops.

- Exploit this observation

  1. two-level minimization after one hot encoding identifies useful state group for decoding
  2. assigning the states in each group to a single face of the hypercube allows a single product term to decode the group to states.

State Assignment

- Assign unique code to each state to produce logic-level description
  - utilize unassigned codes effectively as don't cares
- Choice for S state machine
  - minimum-bit encoding

    $\log S$
  - maximum-bit encoding
    - one-hot encoding
    - using one bit per state
    - something in between
- Modern techniques
  - hypercube embedding of face constraint derived for collections of states (Kiss,Nova)
  - adjacency embedding guided by weights derived between state pairs (Mustang)

Example

Ex: state machine

input	current-state	next state	output
0	start	S6	00
0	S2	S5	00
0	S3	S5	00
0	S4	S6	00
0	S5	start	10
0	S6	start	01
0	S7	S5	00
1	start	S4	01
1	S2	S3	10
1	S3	S7	10
1	S4	S6	10
1	S5	S2	00
1	S6	S2	00
1	S7	S6	00

Symbolic Implicant : represent a transition from one or more state to a next state under some input condition.

State Assignment

Symbolic cover representation is related to a multiple-valued logic.

Positional cube notation : a p multiple-valued logic is represented as P bits

$(V_1, V_2, ..., V_p)$

Ex: V = 5 for 5-value logic

(00010)

represent a set of values by one string

V = 2 or V = 4

(01010)

State Assignment

Find a minimum multiple-valued-input cover

- espresso

Ex: A minimal multiple-valued-input cover

0 0110001	0000100 00
0 1001000	0000010 00
1 0001001	0000010 10

State Assignment

Consider the first symbolic implicant

0 011001 | 0000100 00

- This implicant shows that input “0” means “state-2” or “state-3” or “state-7” into “state-5” and assert output “00”
- This example shows the effect of symbolic logic minimization is to group together the states that are mapped by some input into the same next-state and assert the same output.
- We call it “state group” if we give encodings to the states in the state group in adjacent binary logic and no other states in the group face, then the states group can be implemented as a cube.

State Assignment

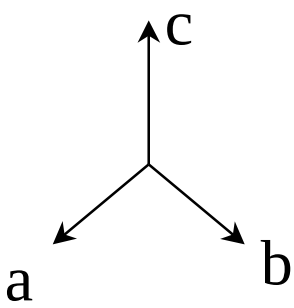
- group face : the minimal dimension subspace containing the encoding assigned to that group.

  Ex: 0010 0**0 group face

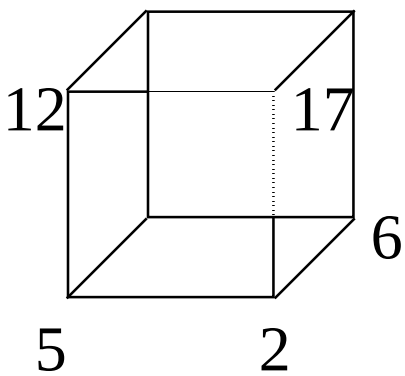
  0100

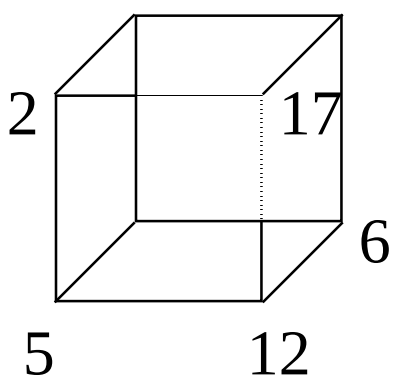
  0110

Hybercube Embedding



state groups :
{2,5,12,17}
{2,6,17}





wrong!

Hybercube Embedding

- Advantage :
  - use two-level logic minimizer to identify good state group
  - almost all of the advantage of one-hot encoding, but fewer state-bit

Adjacency-Based State Assignment

Basic algorithm:

(1) Assign weight w(s,t) to each pair of states
- weight reflects desire of placing states adjacent on the hybercube

(2) Define cost function for assignment of codes to the states
- penalize weights for the distance between the state code

  eg. w(s,t) * distance(enc(s),enc(t))

(3) Find assignment of codes which minimize this cost function summed over all pairs of states.
- heuristic to find initial solution
- pairwise interchange (simulated annealing) to improve solution

Adjacency-Based State Assignment

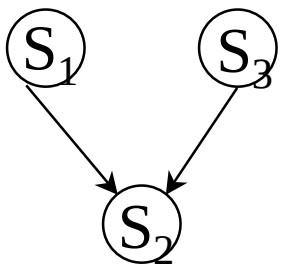
- Mustang : weight assignment technique based on loosely maximizing common cube factors

How to Assign Weight to State Pair

- Assign weights to state pairs based on ability to extract a common-cube factor if these two states are adjacent on the hybercube.

Fan-Out-Oriented
(examine present-state pairs)

- Present state pair transition to the same next state

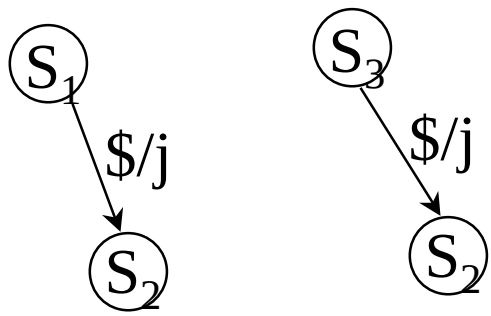


$\$\$ \quad S_1 \; S_2 \; \$\$\$\$$
$\$\$ \quad S_3 \; S_2 \; \$\$\$\$$

Add n to $w(S_1,S_3)$ because of S_2

Fan-Out-Oriented

- present states pair asserts the same output

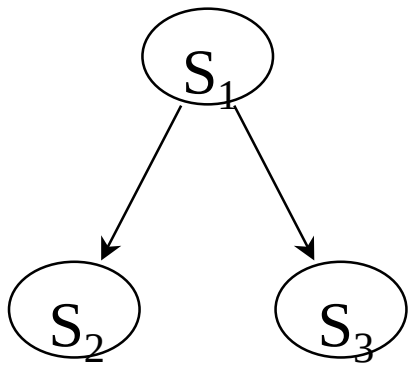


$\$\$\$ \quad S_1 \quad S_2 \quad \$\$\$1\$$
$\$\$\$ \quad S_3 \quad S_4 \quad \$\$\$1\$$

Add 1 to w(S_1 , S3) because of output j

Fanin-Oriented (exam next state pair)

- The same present state causes transition to next state pair.



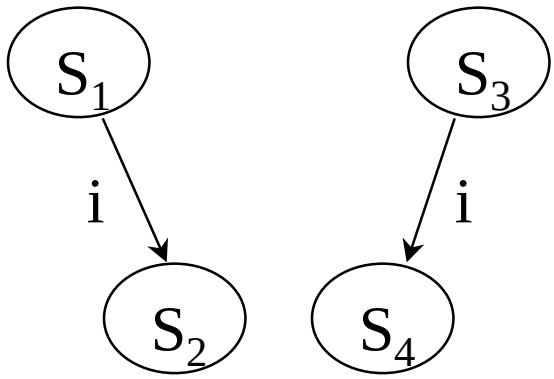
$\$\$\$ \quad S_1 \quad S_2 \quad \$\$\$\$$

$\$\$\$ \quad S_1 \quad S_4 \quad \$\$\$\$$

Add n/2 to $w(S_2,S_4)$ because of S_1

Fanin-Oriented (exam next state pair)

- The same input causes transition to next state pair.



0 S_1 S_2 $$$$

0 S_3 S_4 $$$$

Add 1 to w(S_2,S_4) because of input 2

Which Method Is Better?

- Which is better?

  FSMs have no useful two-level

  face constraints => adjacency-embedding

  FSMs have many two-level

  face constraints => face-embedding