COL215 Digital Logic and System Design

Logic and FSM Minimization – Algorithmic Approach

03 November 2017

Algorithmic approach – tabular method

Quine McCluskey method

Two phases :

- Generation of all prime implicants
- Finding a minimum cover

Iterative steps for PI generation

In i^{th} iteration, implicants with 2.i minterms are generated by combining two implicants with i minterms. Let the number of variables be denoted by n.

What is the maximum number of iterations that may be required?

Grouping of implicants

In each iteration, the implicants are grouped according to the number of 1’s in the iimplicants.

What is the maximum number of groups that may be formed in i^{th} iteration?

Groups from which implicants combine

In i^{th} iteration, implicants from jth group and kth group can combine.

How are j and k related?

Finding minimum cover

- What are essential prime implicants?
- What role is played by essential prime implicants?
- Which rows are removed – dominating or dominated?
- Which columns are removed – dominating or dominated?
- What is branching and when is it used?

Algorithmic approach – cubical technique

Starting point need not be the minterm set

Two operations over cubes:

- star (*) operation
  - used to generate new cubes
- sharp (#) operation
  - used to check coverage

Star (*) operation

$A_i \backslash B_i$	0	1	x
0	0	ø	0
1	ø	1	1
x	0	1	x

Possible results of operation

C = A * B on cubes A and B

- if A and B are far apart, no new cube is generated
- if A, B overlap, the result is intersection of A and B
- if A, B differ in exactly one literal, new cube generated is union of a part of A and a part of B

If $A_i * B_i = ø$ for more than one i,
then C = ø
Otherwise,
if $A_i * B_i \neq ø$ then
$C_i = A_i * B_i$
else
$C_i = x$

Sharp (#) operation

$A_i \backslash B_i$	0	1	x
0	ε	ø	ε
1	ø	ε	ε
x	1	0	ε

Possible results of operation

C = A # B on cubes A and B

- if A and B are disjoint, the result is A
- if A is contained in B, the result is empty
- if B partly overlaps with A, the result is non-overlapping part of A

If A_i # B_i = ø for some i,
then C = A

If A_i # B_i = ε for all i,
then C = ø

Finite State Machine Model

Quintuple $(X, Y, S, \delta, \lambda)$

- X = set of primary input patterns
- Y = set of primary output patterns
- S = set of states {includes an initial state}
- δ = state transition function $S \times X \rightarrow S$
- λ = output function
  - $S \times X \rightarrow Y$ for Mealy model or
  - $S \rightarrow Y$ for Moore model

Assumptions

- Synchronous circuits
- Single clock
- Single phase
- Edge triggered D type flip-flops/registers

State minimization

- Deriving an FSM with similar behavior and minimum number of states
- State minimization may lead to reduction in the register bits and logic gates (because of reduced no. of transitions)

Completely specified FSMs

Definition of state equivalence:

equiv (s_i , s_j) iff

output_seq (s_i, input_seq) =

output_seq (s_j, input_seq)

for all input_seq

Completely specified FSMs

Easier way to check:

equiv (s_i , s_j) iff

$(s_i = s_j)$ — s_i, s_j are identical

or

$\lambda (s_i, x) = \lambda (s_j, x)$ and — produce same output for x

equiv $(\delta (s_i, x), \delta (s_j, x))$, — their x-successor are equivalent

for all x in X

Computing equiv relation

Iterative definition:

s_i , s_j are *k-equivalent*

iff they can't be distinguished by input sequences of length upto k

=>

s_i , s_j are (k+1)-equivalent

iff they are k-equivalent and

their x-successors are k-equivalent, for all x

Computing equiv relation

Another iterative definition:

s_i , s_j are *k-distinguishable*

iff they can be distinguished by input sequences of length upto k

=>

s_i , s_j are (k+1)-distinguishable

iff they are k-distinguishable or

their x-successors is k-distinguishable, for some x

Partitioning states into sets of equivalent states

- Required partition is obtained as a series $\pi_1, \pi_2, \pi_3, \ldots, \pi_n$. Max value of $n = |S|$
- In π_k , s_i and s_j are in the same block iff they are k-equivalent
- π_{k+1} is obtained from π_k by checking if successors of state pairs in same block are k-equivalent or not
- The process terminates when π_{k+1} is found to be same as π_k

QUESTIONS?