4 Reduction of Finite Automata

Multiple model has been defined for finite automata so far. It is possible to summarize the defined models as follows:

1. Recognizing model: This model, which is the basic model of finite automata and shortly called FA, has deterministic (DFA) and non-deterministic (NFA) types.

2. Output generating automaton model: There are also two types of this model, called the Mealy and Moore model (or machine).

- In both the FA model and the output generating finite automata model, the complexity of the defined finite automata (machine) is directly proportional to the number of states.
- Therefore, given a finite automata, it may be important to reduce this finite automata. The reduction or simplification of a finite automaton is to find the finite automata with the smallest finite number of states equivalent to that finite automata.
- The reduction of finite automata will be seen for DFA, Moore and Mealy models. For this, particular definitions must be made first.

4.1 Definitions of Succesor, Predecessor, Equivalent and Distinguishable State

- The largest of the DFA, Moore and Mealy models is the Mealy model.
- Because the recognizer (FA) model can be seen as a subtype of the Moore model with the output alphabet {accept, rejection}.
- The Moore model, on the other hand, can be seen as a subtype of the Mealy model, whose output function is independent of the input alphabet. For this reason, the Mealy model will be taken as a basis while the definitions are given, and the M_5 machine, whose definition is given below, will be used as an example.

Example 4.1 The input and output alphabets of the Mealy type M_{11} machine are {0,1} alphabet. The transition and output functions of the machine with initial state A are defined by the states table below.

States	States, z	
	x=0	x=1
→A	A,0	D,1
B	C,0	E,1
C	G,0	E,1
D	G,0	F,1
E	E,1	C,0
F	B,0	D,1
G	B,0	E,1

Figure 4.1

4.1.1 Succesor

If moving from the S_1 state of the M machine to the S_2 state with the symbol x:

$$S_1 \xrightarrow{x} S_2$$

S_2 is the x-successor of S_1.

- Where w is a string of input symbols, if state S_1 is passed to state S_2 with the input string w, the w-successor of S_1 is S_2. For example, on the M_{11} machine, the 1st successor of state A is D. The 011- successor of case B of the same machine is C.

- With x an input symbol and w a string of input symbols, in deterministic models, the x and w successor of a state is always a single state. In nondeterministic models, the successors x and w are subsets of states. The x-successor of a state can be easily found because it is in the transition table. To find the successor $w = x_1 x_2 \ldots x_k$ of a state, find the successor consecutive using the transition table; then x_2-successor of x_i-successor, etc. is obtained.

$M_2 =< Q, \Sigma, \delta, q_0, F >$

$Q = \{q_0, q_1, q_2, q_3\}$

$\Sigma = \{0,1\}$

$F = \{q_3\}$

$\delta(q_0, 0) = \{q_0, q_1\}$ $\quad$ $\delta(q_2, 0) = \phi$

$\delta(q_0, 1) = \{q_0, q_2\}$ $\quad$ $\delta(q_2, 1) = \{q_3\}$

$\delta(q_1, 0) = \{q_3\}$ $\quad$ $\delta(q_3, 0) = \{q_3\}$

$\delta(q_1, 1) = \{\} = \phi$ $\quad$ $\delta(q_3, 1) = \{q_3\}$

- For example, in the M_2 machine, the 0-successor of the q_0 state is $\{q_0, q_1\}$. $\{q_0, q_1\}$ can be replaced with q_0q_1. Accordingly, the 0-successor of the q_0 case is q_0q_1 , the 1-successor is q_0q_2, and the 100-successor is $q_0q_1q_3$. The 1-successor of the q_1 state of the same machine and the 01-successor of the q_2 state do not exist (it is the empty set).

4.1.2 Predecessor

- If the state S_j of machine M is passed from the states S_1, S_2, ..., and S_i with the input symbol x, the x-predecessor of state S_j is $\{S_1, S_2, ..., S_i\}$.

- For example, for machine M_{11}, 0-predecessor of state B is FG and 001-predecessor is AEF. State D has no 0 or 110-predecessors for the same machine. To find the x-predecessors of a machine, a table called predecessor chart can be created.

Statets	Predecessor State x=0	x=1
A	A	-
B	FG	-
C	B	E
D	-	AF
E	E	BCG
F	-	D
G	CD	-

States	States, z x=0	x=1
A	A,0	D,1
B	C,0	E,1
C	G,0	E,1
D	G,0	F,1
E	E,1	C,0
F	B,0	D,1
G	B,0	E,1

4.1.3 Equivalent States

- When the machine M is in any of the S_1 or S_2 states, if the machine always produces the same output symbol, regardless of which input symbol is applied, these states are called 1-equivalent states. For example, for M_{11} machine A and B is 1-equivalent states. Against this, E and G states is not 1-equivalent states.

- When machine M is in either state S_1 or S_2, these states are called n-equivalent states if the machine always produces the same output string no matter which input string of length n or less is applied. For example, A and D states of machine M_{11} are 2-equivalent. Cases B and C are 3-equivalent. B and C states are also 4-equivalent, 5-equivalent, ...etc. In contrast, states A and D are not 3-equivalent.

- When machine M is in either state S_1 or S_2, these states are called equivalent states if the machine always produces the same output string, regardless of its length, no matter what input string is applied. For example, states B and C of machine M_{11} are equivalent states. States D and F are also equivalent. However, the 2-equivalent states A and D are not.

- In practice, states that are n-equivalent for all n are called equivalent states. Two cases that are k-equivalent for all k-values less than n (k ≤ n) are called n-equivalent. For example, the A and F states of the M_{11} machine are 2-equivalent. Qualifying these cases as 2-equivalent means they are not 3-equivalent. However, 2-equivalents are naturally 1-equivalent. However, state equivalence is always specified over the largest n value.

4.1.4 Distinguishable State

- If an input string of at least n length is required to distinguish the S_1 and S_2 states of the M machine, these states are called n-distinguishable states. If states S_1 and S_2 are n-distinguishable, these two states are (n-1)-equivalent. For example, states A and E of machine M_{11} are 1-distinguishable. States B and D are 2-distinguishable. In contrast, states A and F are 3- distinguishable. It is not possible to distinguish D and F states with any input string. Equivalent D and F states are indistinguishable.

- $A \xrightarrow[Z=0]{X=0} A$ E $\xrightarrow[Z=1]{X=0} E$ A and E are 1- distinguishable
- B $\xrightarrow[Z=11]{X=10} E$ D $\xrightarrow[Z=10]{X=10} B$ B and D are 2- distinguishable
- $A \xrightarrow[Z=010]{X=010} G$ F $\xrightarrow[Z=011]{X=010} E$ A and F are 3- distinguishable

Machine Equivalance

- If for every state of machine M_1 there is an equivalent state of machine M_2; For each state of machine M_2, if there is an equivalent state in machine M_1, machines M_1 and M_2 are equivalent.

$$\left.\begin{array}{l} \forall S_{1i} \in M_1 \rightarrow \exists S_{2j} \in M_2 : S_{1i} \equiv S_{2j} \\ \forall S_{2i} \in M_2 \rightarrow \exists S_{1j} \in M_1 : S_{2i} \equiv S_{1j} \end{array}\right\} M_1 \equiv M_2$$

4.2 Reduction of Machine

- Given a machine M, reducing or simplifying it means finding the machine with the smallest number of states equivalent to that machine.
- Equivalence partitions are used to reduce finite automata. For an M machine, the k-equivalence partition, denoted P_k, is a partition in which k-equivalent states are in the same partition. For example, 0-equivalence partitioning for a Mealy machine M_{11}:

$$P_0 = (ABCDEFG)$$

- contains only one section. Because it is not possible to distinguish states of a Mealy machine without any input symbols applied. For M_{11}, the 1-equivalence partitioning is

$$P_1 = (ABCDFG)(E)$$

- contains two sections. Because all states of the machine except E are 1-equivalent

- To find the equivalence partitioning of the machine, find $P_0, P_1, P_2, \ldots$ in order. Derivation of equivalence partitions is continued until the below equation is obtained:

$$P_k = P_{k+1}$$

- When the equivalence partitioning is

$$P = P_k$$

is found to be true and the derivation is interrupted.

- Proposition 4.1 is used to derive P_{m+1} equivalence partitioning from P_m equivalence partitioning.

Proposition 4.1 In order for the S_1 and S_2 states of the machine M to be (m+1)-equivalent, the following two conditions are necessary and sufficient.

a) S_1 and S_2 must be m-equivalent (P_m must be in the same partition in the equivalence partitioning).

b) For all the input symbols x, the x-successors of the states S_1 and S_2 must also be m-equivalent (they must be in the same section in the P_m equivalence partitioning).

4.2.1 Reduction of Mealy Machines

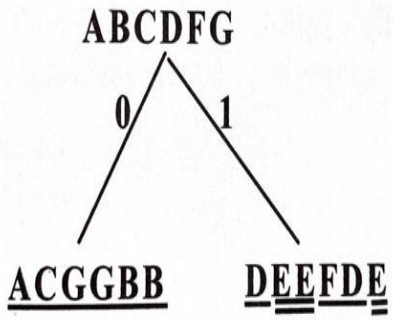
The reduction of Mealy machines will be studied by taking the M_{11} machine as an example. As stated earlier, the states of the Mealy machine cannot be distinguished without applying any input. In other words, all states of the Mealy machine are 0-equivalent.

$$P_0 = (ABCDEFG)$$

It is sufficient to examine the state table to find the equivalence partitioning of P_1. In the state table of M_{11}, it is seen that the output value produced during the x pass is 0 in the first column and 1 in the second column for all cases except E. Accordingly, cases other than E are 1-equivalent.

$$P_1 = (ABCDFG)(E)$$

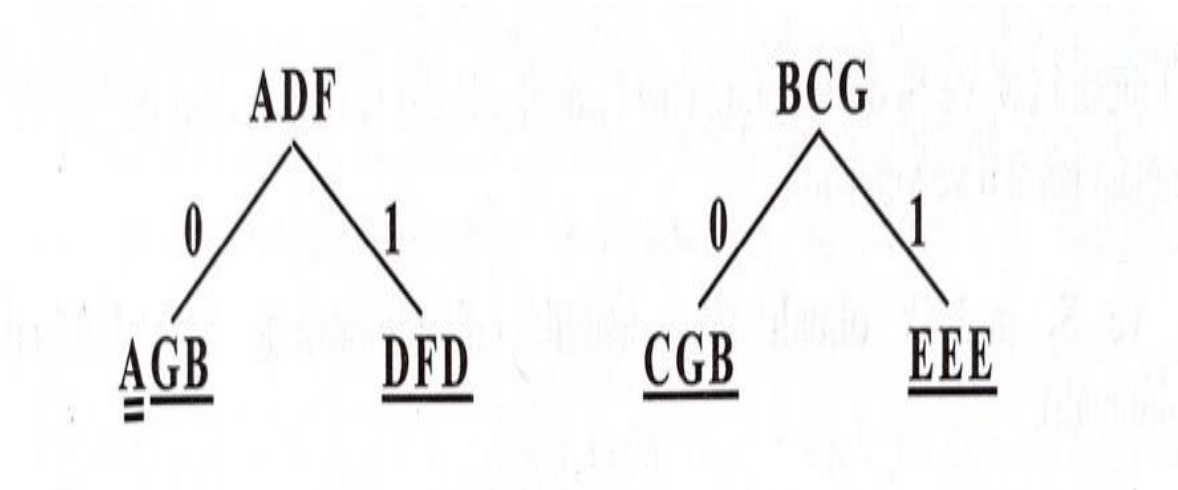
Proposition 4.1 is used to find the P_2 equivalence partition. In order for the two states of the M_{11} machine to be 2-equivalent, both these two states and the 0 and 1-succesors of these two states must be 1-equivalent; for this, it must be in the same section in the P_1 equivalence partitioning.



In the P_1 -partition, the 0-successors of the states ABCDFG, which are located in the same section, are 1-equivalent due to their presence in the same P_1 partition. However, not all 1-successors of these states are 1-equivalent. From this, it is understood that the ADF states are 2-equivalent among themselves and the BCG states are 2-equivalent among themselves, and the following P_2 equivalence partitioning is obtained:

$$P_2 = (ADF)(BCG)(E)$$

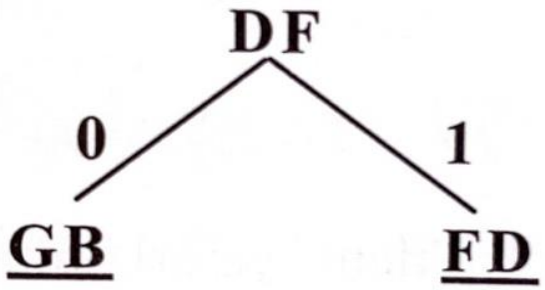
To find the P_3 equivalence partitioning, 0 and 1- successors of ADF and BCG states are examined:



Upon examination, it is seen that 1-successors of ADF states are in the same section in P_2, but 0-successors are not in the same section in P_2. Therefore, it is understood that the DF states are 3-equivalent, but the A state is not 3-equivalent to the D and F states, and in P_3, the A state differs from the DF states. As for the BCG states, both 0 and 1-successors of these cases are in the same partition in P_2. Accordingly, the BCG cases are 3-equivalents and will be in the same partition in the P_3 equivalence partition. As a result, the P_3 equivalence partitioning occurs as follows:

$$P_3 = (A)(DF)(BCG)(E)$$

To find the P_4 equivalence partitioning, it is necessary to examine the 0 and 1 successors of the DF and BCG states. The 0 and 1-successors of the BCG states, previously reviewed and included above, are also included in P_3 in the same section. Therefore, BCG states will be included in P_4 in the same section.



When the successors of the DF cases are examined, it is seen that the 0 and 1-successors of these two cases are in the same section in P_3. Therefore, the DF states will be in the same section in P_4. As a result, the equivalence partition P_4 is equal to P_3.

$$P_4 = P_3 = (A)(DF)(BCG)(E)$$

Thus, the equivalence partitioning of the M_5 machine is obtained as :

$$P = (A)(DF)(BCG)(E)$$

Since there are 4 partitions in equivalence partitioning, the smallest machine equivalent to M_{11} will have 4 states. If the states of the smallest machine are called as below :

for A : S_0

for DF : S_1

for BCG : S_2

for E : S_3

The state table of the smallest machine equivalent to machine M_{11} is found as follows:

States	SD, z	
	x=0	x=1
S_0	S_0,0	S_1,1
S_1	S_2,0	S_1,1
S_2	S_2,0	S_3,1
S_3	S_3,1	S_2,0

4.2.2 Reduction of Moore Machines

- As known, the output function in Moore machines is a mapping from the set of states to the output alphabet. With this mapping, an exit symbol is mapped to each state. When a Moore machine is in a certain state, it produces a specific exit symbol. Therefore, certain states or subsets of states can be distinguished without applying any input symbols. In other words, not all states of the Moore machine are 0-equivalent. The 0-equivalence partitioning of the Moore machine (P_0) has as many partitions as the output symbol.

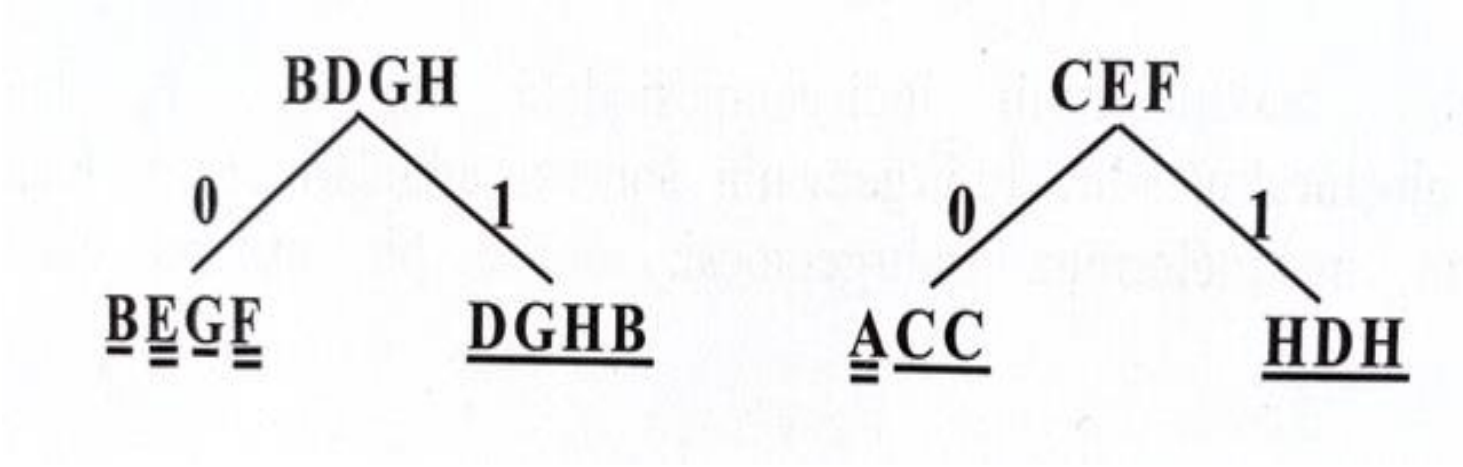
- The only difference in the reduction of Mealy and Moore machines is the establishment of the equivalence partitioning P_0. The next steps of the reduction are carried out similarly. The reduction of Moore machines will be seen on an example.

Example 4.2 The input alphabet of the Moore M_{12} machine is {0,1} and the output alphabet is {0,1,2}. The transition and output functions of the machine with initial status A are described by the status table below:

States	SD		z
	x=0	x=1	
→A	C	B	0
B	B	D	1
C	A	H	2
D	E	G	1
E	C	D	2
F	C	H	2
G	G	H	1
H	F	B	1

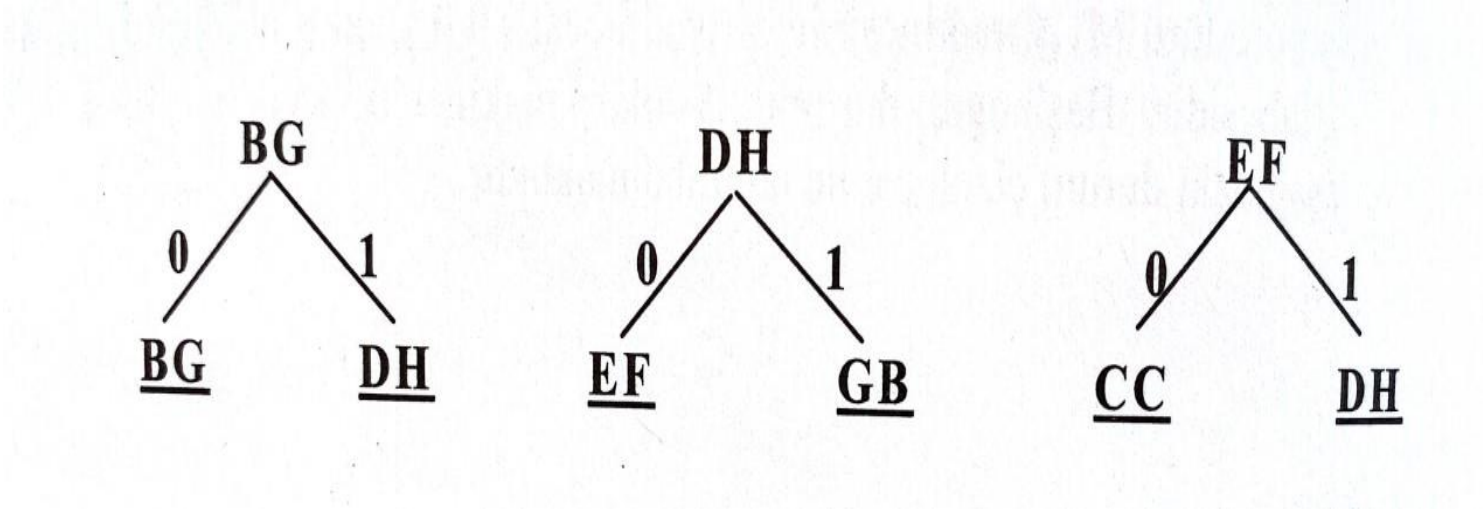
The machine M_{12} is reduced as below:

$$P_0 = (A)(BDGH)(CEF)$$



$$P_1 = (A)(BG)(DH)(C)(EF)$$

$$P_1 = (A)(BG)(DH)(C)(EF)$$



$$P_2 = P_1 = (A)(BG)(DH)(C)(EF)$$
$$P = (A)(BG)(DH)(C)(EF)$$

$$P = (A)(BG)(DH)(C)(EF)$$

Since there are 5 sections in equivalence partitioning, the smallest machine equivalent to M_{12} will have 5 states. If the states renamed as below:

for A, S_0

for BG, S_1

for DH, S_2

for C, S_3

for EF, S_4

State table of the smallest machine equivalent to M_{12} is as below:

States	SD, z		z
	x=0	x=1	
S_0	S_3	S_1	0
S_1	S_1	S_2	1
S_2	S_4	S_1	1
S_3	S_0	S_2	2
S_4	S_3	S_2	2

4.2.3 Reduction of DFA Model

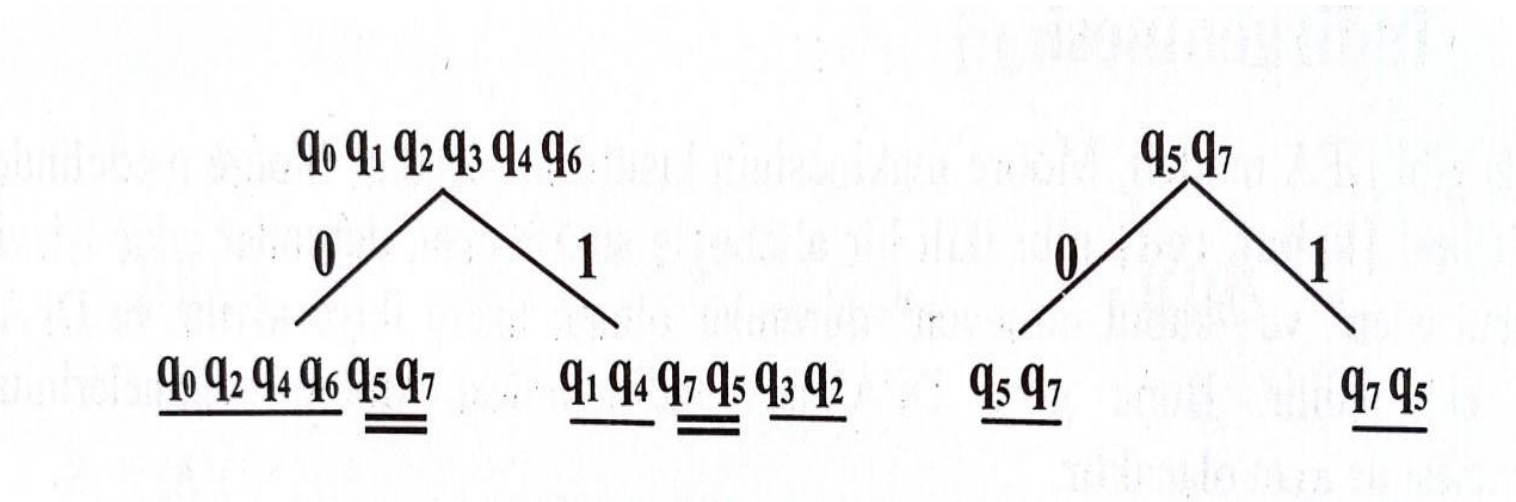
As is known, the DFA model is a constrained variant of the Moore machine. If the output alphabet in the Moore model is limited to a binary alphabet such as {accept, reject}, the states are divided into two as "acceptable" and "not accepting" states with the output function and the DFA model is obtained. Accordingly, the reduction of DFAs will be the same as the reduction of Moore's machines.

Example 4.3 M_{13}, which is a deterministic automata (DFA), is described by the state table below:

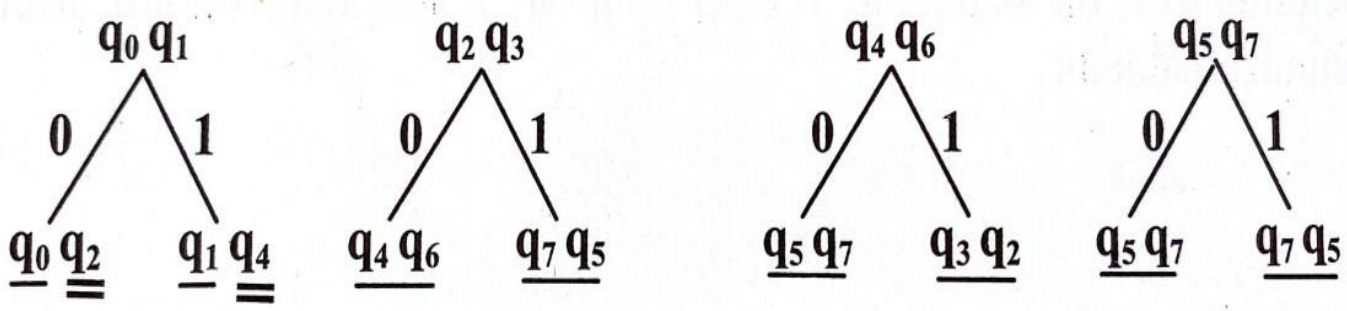
States	Next States	
	x=0	x=1
$\rightarrow q_0$	q_0	q_1
q_1	q_2	q_4
q_2	q_4	q_7
q_3	q_6	q_5
q_4	q_5	q_3
(q_5)	q_5	q_7
q_6	q_7	q_2
(q_7)	q_7	q_5

The machine M_{13} is reduced as below:

$$P_0 = (q_0q_1q_2q_3q_4q_6)(q_5q_7)$$



$$P_1 = (q_0q_1)(q_2q_3)(q_4q_6)(q_5q_7)$$



$$P_2 = (q_0)(q_1)(q_2q_3)(q_4q_6)(q_5q_7)$$
$$P_3 = P_2$$
$$P = (q_0)(q_1)(q_2q_3)(q_4q_6)(q_5q_7)$$

Since there are 5 sections in equivalence partitioning, the smallest machine equivalent to M_7 will have 5 states. If the states renamed as below:

for q_0, S_0

for q_1, S_1

for $q_2 q_3, S_2$

for $q_4 q_6, S_3$

for $q_5 q_7, S_4$

State table of the smallest machine equivalent to M_{13} is as below:

States	SD, z	
	x=0	x=1
S_0	S_0	S_1
S_1	S_2	S_3
S_2	S_3	S_4
S_3	S_4	S_2
(S_4)	S_4	S_4