Module 1:
Introduction to Languages and Automata

Dr. A K Yadav

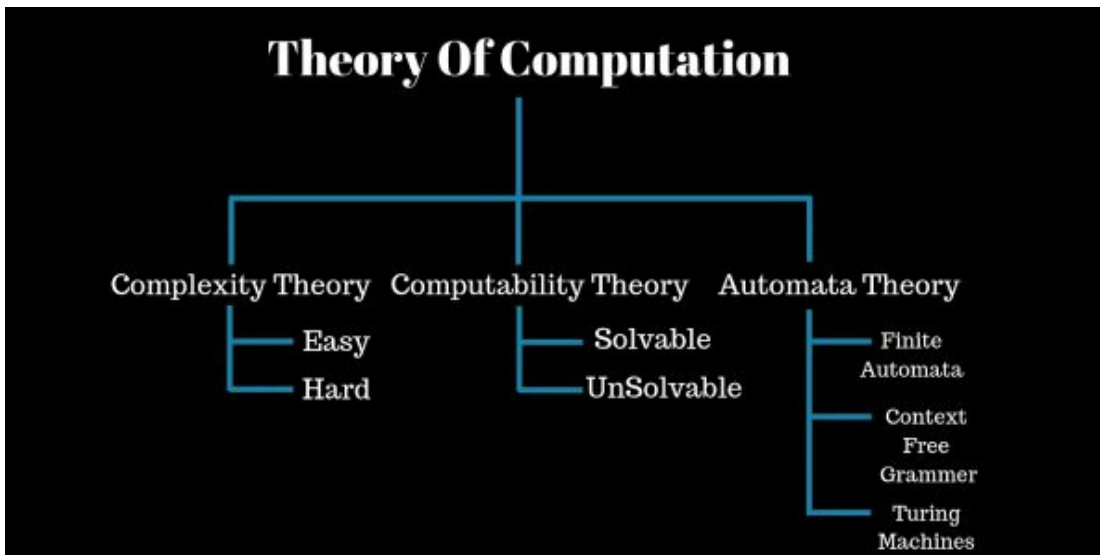
April 28, 2022

Outline

Outline

Theory of Computation: Introduction

- Theory of Computation is the branch of Computer Science which deals with how efficiently problems can be solved on model of computation using an algorithm
- The domain is further classified into 3 sub-domains:
  - Automata theory and languages
  - Computability theory
  - Complexity theory



Preliminaries

- Propositions (or Statements)

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)
  - NOT (Negation)$\neg P$
  - AND (Conjunction) $P \wedge Q$
  - OR (Disjunction) $P \vee Q$
  - If.. Then... (Implication)
  - If and only If

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)
  - NOT (Negation)$\neg P$
  - AND (Conjunction) $P \wedge Q$
  - OR (Disjunction) $P \vee Q$
  - If.. Then... (Implication)
  - If and only If
- Tautology- A tautology or a universally true formula is a well defined formula whose truth value is T for all possible assignments of truth values to the propositional variables.

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)
  - NOT (Negation)$\neg P$
  - AND (Conjunction) $P \wedge Q$
  - OR (Disjunction) $P \vee Q$
  - If.. Then... (Implication)
  - If and only If
- Tautology- A tautology or a universally true formula is a well defined formula whose truth value is T for all possible assignments of truth values to the propositional variables. **Example-**$P \vee \neg P$

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)
  - NOT (Negation)$\neg P$
  - AND (Conjunction) $P \wedge Q$
  - OR (Disjunction) $P \vee Q$
  - If.. Then... (Implication)
  - If and only If
- Tautology- A tautology or a universally true formula is a well defined formula whose truth value is T for all possible assignments of truth values to the propositional variables. **Example-**$P \vee \neg P$
- Contradiction- A contradiction (or absurdity) is well formed formula whose truth value is F for all possible assignments of truth values to propostion variables.

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)
  - NOT (Negation)$\neg P$
  - AND (Conjunction) $P \wedge Q$
  - OR (Disjunction) $P \vee Q$
  - If.. Then... (Implication)
  - If and only If
- Tautology- A tautology or a universally true formula is a well defined formula whose truth value is T for all possible assignments of truth values to the propositional variables.
  Example-$P \vee \neg P$
- Contradiction- A contradiction (or absurdity) is well formed formula whose truth value is F for all possible assignments of truth values to propostion variables.
  Example-$P \wedge \neg P$

Preliminaries

- Propositions (or Statements)
- Connectives (Propositional connectives or Logical connectives)
  - NOT (Negation)$\neg P$
  - AND (Conjunction) $P \wedge Q$
  - OR (Disjunction) $P \vee Q$
  - If.. Then... (Implication)
  - If and only If
- Tautology- A tautology or a universally true formula is a well defined formula whose truth value is T for all possible assignments of truth values to the propositional variables.
  Example-$P \vee \neg P$
- Contradiction- A contradiction (or absurdity) is well formed formula whose truth value is F for all possible assignments of truth values to propostion variables.
  Example-$P \wedge \neg P$
- Equivalence

Preliminaries

- Equivalence- Two well formed α and β in propositional variables $P_1, P_2, ...P_n$ are equivalent (or logically equivalent) if the formula $\alpha \leftrightarrow \beta$ is a tautology.

Preliminaries

- Equivalence- Two well formed α and β in propositional variables $P_1, P_2, ...P_n$ are equivalent (or logically equivalent) if the formula $\alpha \leftrightarrow \beta$ is a tautology.

Example

$(P \implies (Q \vee R)) \equiv ((P \implies Q) \vee (P \implies R))$

Preliminaries

- Logical Identities

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$
  - Associative laws- $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$
    $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$
  - Associative laws- $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$
    $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$
  - Distributive laws- $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$
    $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$
  - Associative laws- $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$
    $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$
  - Distributive laws- $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$
    $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
  - Absorption laws- $P \wedge (P \vee Q) \equiv P$
    $P \vee (P \wedge Q) \equiv P$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$
  - Associative laws- $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$
    $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$
  - Distributive laws- $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$
    $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
  - Absorption laws- $P \wedge (P \vee Q) \equiv P$
    $P \vee (P \wedge Q) \equiv P$
  - De-morgan's Laws- $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
    $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$
  - Associative laws- $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$
    $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$
  - Distributive laws- $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$
    $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
  - Absorption laws- $P \wedge (P \vee Q) \equiv P$
    $P \vee (P \wedge Q) \equiv P$
  - De-morgan's Laws- $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
    $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
  - Contrapositive- $P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P$
    $P \Rightarrow Q \equiv \neg P \vee Q$

Preliminaries

- Logical Identities
  - Idempotent laws- $P \vee P \equiv P$, $P \wedge P \equiv P$
  - Commutative laws- $P \vee Q \equiv Q \vee P$, $P \wedge Q \equiv Q \wedge P$
  - Associative laws- $P \vee (Q \vee R) \equiv (P \vee Q) \vee R$
    $P \wedge (Q \wedge R) \equiv (P \wedge Q) \wedge R$
  - Distributive laws- $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$
    $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
  - Absorption laws- $P \wedge (P \vee Q) \equiv P$
    $P \vee (P \wedge Q) \equiv P$
  - De-morgan's Laws- $\neg(P \vee Q) \equiv \neg P \wedge \neg Q$
    $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
  - Contrapositive- $P \Rightarrow Q \equiv \neg Q \Rightarrow \neg P$
    $P \Rightarrow Q \equiv \neg P \vee Q$
  - Double negation- $P \equiv \neg(\neg P)$

Questions

- Show that:
  $(P \wedge Q) \vee (P \wedge \neg Q) \equiv P$

Questions

- Show that:
  $(P \wedge Q) \vee (P \wedge \neg Q) \equiv P$
- Show that:
  $(P \implies Q) \wedge (R \implies Q) \equiv (P \vee R) \implies Q$

Mathematical Preliminaries

Set:

- A **set** is well defined collection of objects.
  Example-Set of all students in ASET,
  Collection of all books in library.
- Individual objects are called **Members or Elements** of the set.
- Capital letter usually represent Set such as $A, B, C, \ldots$
- Small letters represent Elements of any set such as $a, b, c, \ldots$.
- If a is an element of set $A \implies a \in A$

Mathematical Preliminaries

Set:

- A **set** is well defined collection of objects.
  Example-Set of all students in ASET,
  Collection of all books in library.
- Individual objects are called **Members or Elements** of the set.
- Capital letter usually represent Set such as $A,B,C,...$
- Small letters represent Elements of any set such as $a,b,c,....$
- If a is an element of set $A \implies a \in A$
- Ways of describing set
  - Listing its element with no repetition
    Example: $\{15, 30, 45, 60, 75, 90\}$

Mathematical Preliminaries

Set:

- A **set** is well defined collection of objects.
  Example-Set of all students in ASET,
  Collection of all books in library.
- Individual objects are called **Members or Elements** of the set.
- Capital letter usually represent Set such as $A, B, C, \ldots$
- Small letters represent Elements of any set such as $a, b, c, \ldots$.
- If a is an element of set $A \implies a \in A$
- Ways of describing set
  - Listing its element with no repetition
    Example: $\{15, 30, 45, 60, 75, 90\}$
  - Describing properties of elements of set
    Example: $\{n \mid n is a positive integer divisible by 15 and less than 100\}$

Mathematical Preliminaries

Set:

- A **set** is well defined collection of objects.
  Example-Set of all students in ASET,
  Collection of all books in library.
- Individual objects are called **Members or Elements** of the set.
- Capital letter usually represent Set such as $A, B, C, \ldots$
- Small letters represent Elements of any set such as $a, b, c, \ldots$.
- If a is an element of set $A \implies a \in A$
- Ways of describing set
  - Listing its element with no repetition
    Example: $\{15, 30, 45, 60, 75, 90\}$
  - Describing properties of elements of set
    Example: $\{n \mid n is a positive integer divisible by 15 and less than 100\}$
  - By recursion
    Example: Set of all natural numers leaving remainder 1 when divided by 3 can be written as
    $\{a_n \mid a_0 = 1, a_{n+1} = a_n + 3\}$

Mathematical Preliminaries

Subsets and Operations on Sets

- A set A is said to be subset of B i.e. $(A \subseteq B)$, if every element of A is also an element of B.
- If two sets A and B are *equal* i.e. $(A = B)$
  $\implies A \subseteq B$ and $B \subseteq A$.

Mathematical Preliminaries

Subsets and Operations on Sets

- A set A is said to be subset of B i.e. $(A \subseteq B)$, if every element of A is also an element of B.
- If two sets A and B are *equal* i.e. $(A = B)$ $\implies A \subseteq B$ and $B \subseteq A$.
- Empty set: A set with no elements.

Mathematical Preliminaries

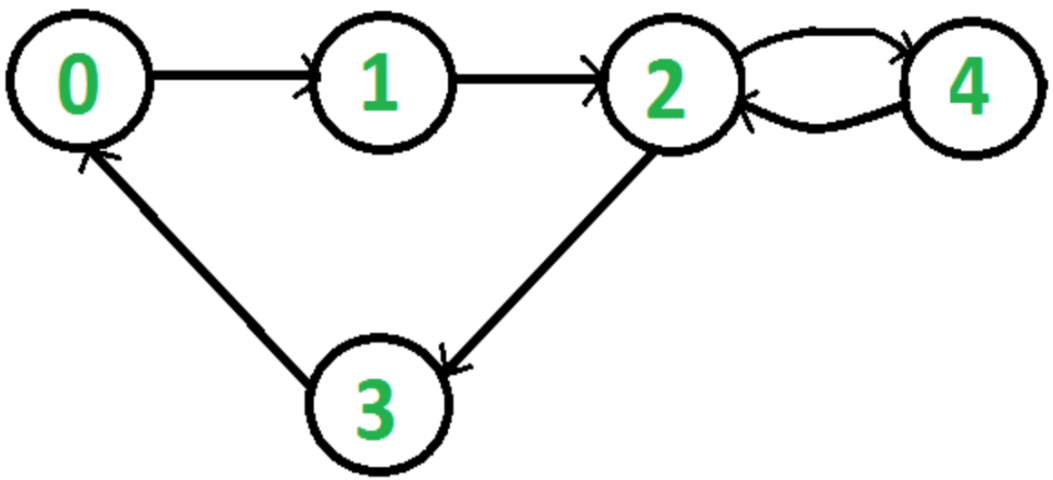
Subsets and Operations on Sets

- A set A is said to be subset of B i.e. $(A \subseteq B)$, if every element of A is also an element of B.
- If two sets A and B are *equal* i.e. $(A = B)$
  $\implies A \subseteq B$ and $B \subseteq A$.
- Empty set: A set with no elements.
- **Operations on sets**:
  - $A \cup B$: $\{x \mid x \in A \text{ or } x \in B\}$ called union of A and B.
  - $A \cap B$: $\{x \mid x \in A \text{ and } x \in B\}$ called intersection of A and B.
  - $A - B$: $\{x \mid x \in A \text{ and } x \notin B\}$ called complement of B in A.

Mathematical Preliminaries

Graph

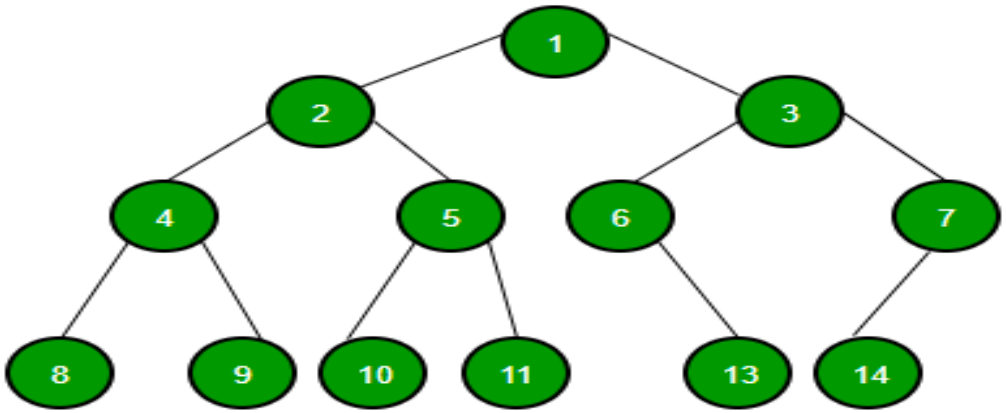
- A graph (or undirected graph) consists of:
  - a non-empty set V of **vertices**
  - a set E called set of **edges**.
  - a **map** ϕ which assigns to every edge a unique unordered pair of vertices.
- A directed graph or (digraph) consists of
  - a non-empty set V of **vertices**
  - a set E called set of **edges**
  - a **map** ϕ which assigns to every edge a unique ordered pair of vertices.



Mathematical Preliminaries

Tree

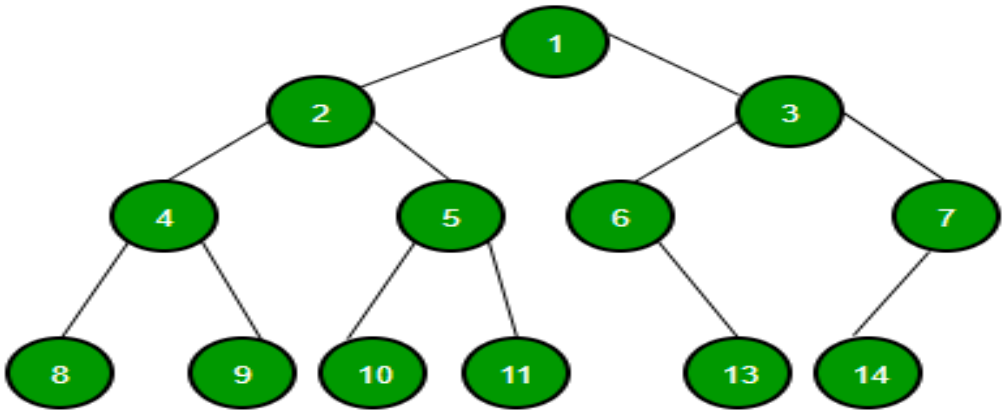
- A graph is called a tree if it is connected and has no circuits



Mathematical Preliminaries

Tree

- A graph is called a tree if it is connected and has no circuits

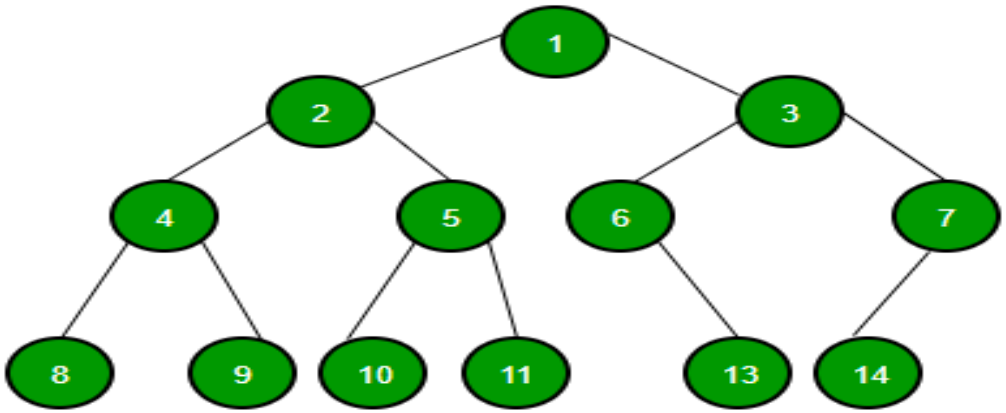


- Properties of tree:
  - A tree is connected **graph with no circuits** or loops

Mathematical Preliminaries

Tree

- A graph is called a tree if it is connected and has no circuits

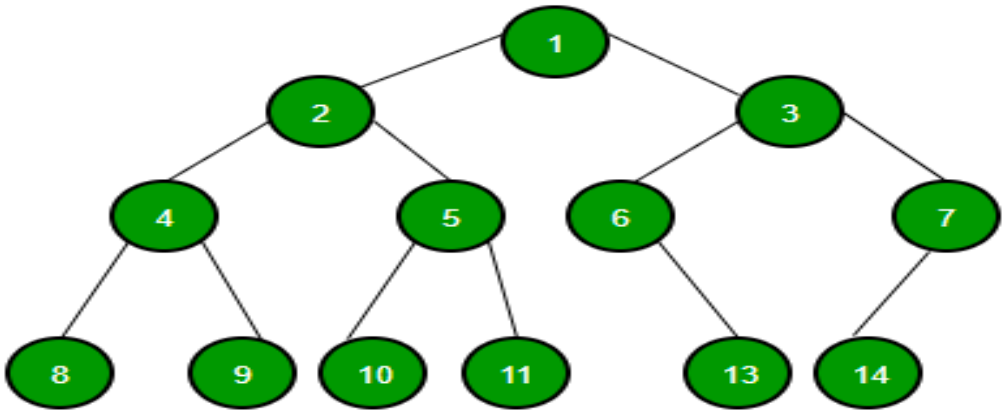


- Properties of tree:
  - A tree is connected **graph with no circuits** or loops
  - there is **one and only one path** between every pair of vertices.

Mathematical Preliminaries

Tree

- A graph is called a tree if it is connected and has no circuits



- Properties of tree:
  - A tree is connected **graph with no circuits** or loops
  - there is **one and only one path** between every pair of vertices.
  - if a connected graph has n **vertices** $\Longrightarrow$ **has** $n-1$ **edges**, implies a tree

Automata

- Automata is defined as a system where energy, material and information are transformed, transmitted and used for performing some function without direct human participation.
- Example: Automatic machine tools, automatic packing machines, automatic photoprinting machines

Figure 1: Automatic machine tools

Figure 2: Automatic photoprinting machine

Discrete Automata

- Model of discrete automaton:

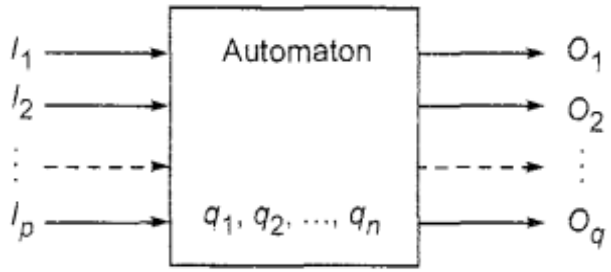


Figure 3: Model of a discrete automaton

- Characteristics of Automata
  - Input-At a discrete instant of **time** $t_1, t_2, \ldots . t_m$, **input values** $I_1, I_2 \ldots . I_p$ take finite number of fixed values from **input alphabet** Σ are applied as input to model.

Discrete Automata

- Model of discrete automaton:

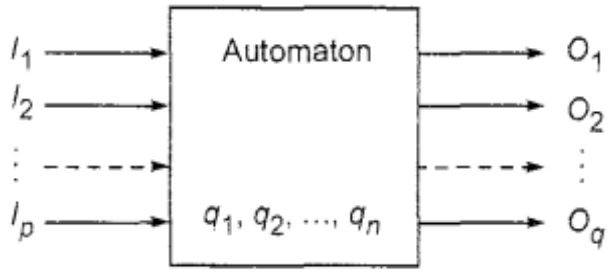


Figure 3: Model of a discrete automaton

- Characteristics of Automata
  - Input-At a discrete instant of **time** $t_1, t_2, ...t_m$, **input values** $I_1, I_2....I_p$ take finite number of fixed values from **input alphabet** Σ are applied as input to model.
  - Output- $O_1, O_2, ...O_q$ are output of model, each of which can take finite number of fixed values from an Output.

Discrete Automata

- Model of discrete automaton:

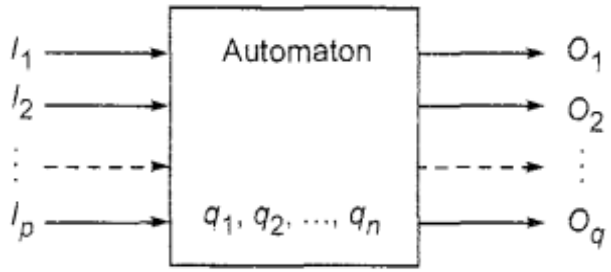


Figure 3: Model of a discrete automaton

- Characteristics of Automata
  - Input-At a discrete instant of **time** $t_1, t_2, ...t_m$, **input values** $I_1, I_2....I_p$ take finite number of fixed values from **input alphabet** Σ are applied as input to model.
  - Output- $O_1, O_2, ...O_q$ are output of model, each of which can take finite number of fixed values from an Output.
  - States- At any instant of time, the automaton can be in one of the states $q_1, q_2, ...q_n$

Discrete Automata

- Model of discrete automaton:

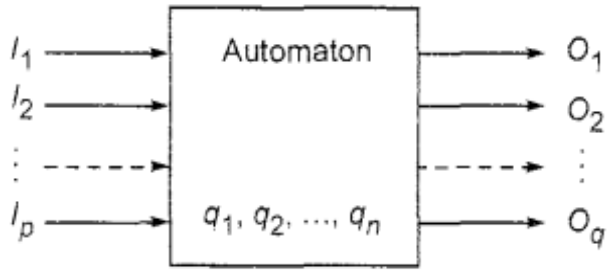


Figure 3: Model of a discrete automaton

- Characteristics of Automata
  - Input-At a discrete instant of **time** $t_1, t_2, ...t_m$, **input values** $I_1, I_2....I_p$ take finite number of fixed values from **input alphabet** Σ are applied as input to model.
  - Output- $O_1, O_2, ...O_q$ are output of model, each of which can take finite number of fixed values from an Output.
  - States- At any instant of time, the automaton can be in one of the states $q_1, q_2, ...q_n$
  - State relation- The next state of an automaton at any instant of time is determined by present state and present input.

Discrete Automata

- Model of discrete automaton:

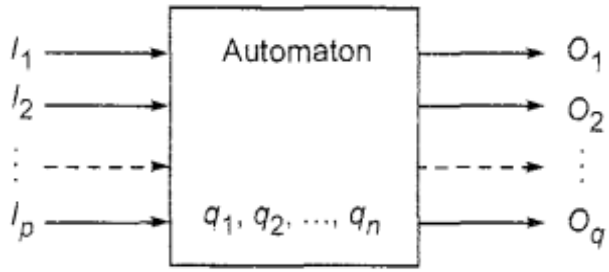


Figure 3: Model of a discrete automaton

- Characteristics of Automata
  - Input-At a discrete instant of **time** $t_1, t_2, ...t_m$, **input values** $I_1, I_2....I_p$ take finite number of fixed values from **input alphabet** Σ are applied as input to model.
  - Output- $O_1, O_2, ...O_q$ are output of model, each of which can take finite number of fixed values from an Output.
  - States- At any instant of time, the automaton can be in one of the states $q_1, q_2, ...q_n$
  - State relation- The next state of an automaton at any instant of time is determined by present state and present input.
  - Output relation- On **reading** an input symbol, automaton moves to next state which is given by state relation.

Discrete Automata

- Automaton without Memory:An automata in which the output depends only on input.
- Automaton with finite Memory: An automaton in which the output depends on states as well as input.

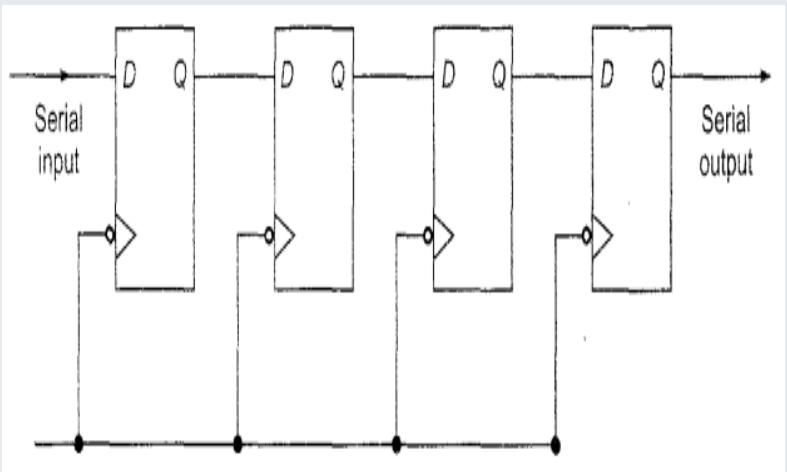
Discrete Automata

- Automaton without Memory:An automata in which the output depends only on input.
- Automaton with finite Memory: An automaton in which the output depends on states as well as input.
  - Moore Machine: An automaton in which the output depends only on states of machine.
  - Mealy Machine: An automaton in which output depends on the state as well as on the input at any instant of time.

Discrete Automata

Any sequential machine behaviour can be represented by an automaton.

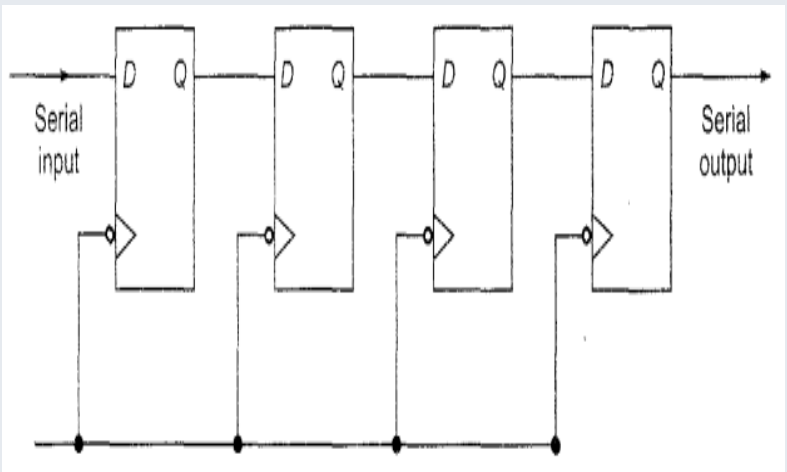
Example: Consider a 4-bit serial shift register as a finite state machine.



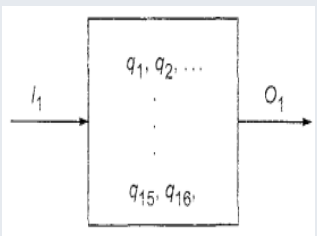
Discrete Automata

Any sequential machine behaviour can be represented by an automaton.

Example: Consider a 4-bit serial shift register as a finite state machine.



- $2^4 = 16$ states (0000, 0001, ..., 1111)
- 1 serial Input and 1 serial output
- Input alphabet, $\Sigma = \{0,1\}$
- Can be represented as



- Here, output depends on both Input and state $\therefore$ **Mealy machine**.

Finite Automaton

- A finite automaton can be represented by a **finite 5-tuple** $(Q, \Sigma, \delta, q_0, F)$, where:
  - Q is a finite non-empty set of **states**.
  - Σ is a finite non-empty set of inputs called **Input** alphabet.
  - δ is a function which maps $Q \times \Sigma \rightarrow Q$ called **Direct transition function**
    It describes change of states during transition.
    It is represented by transition table \ diagram.
  - $q_0 \in Q$ is **Initial state**
  - $F \subseteq Q$ is the set of **Final states**
- The transition function which maps $Q \times \Sigma^*$ into Q is called **Indirect transition function**.

Finite Automata

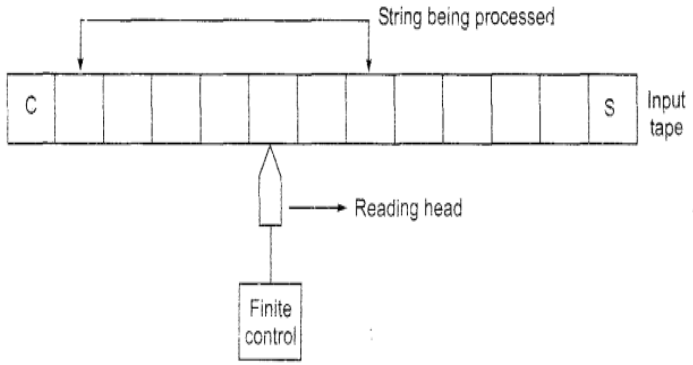


Figure 4: Block diagram of Finite Automata

Finite Automata

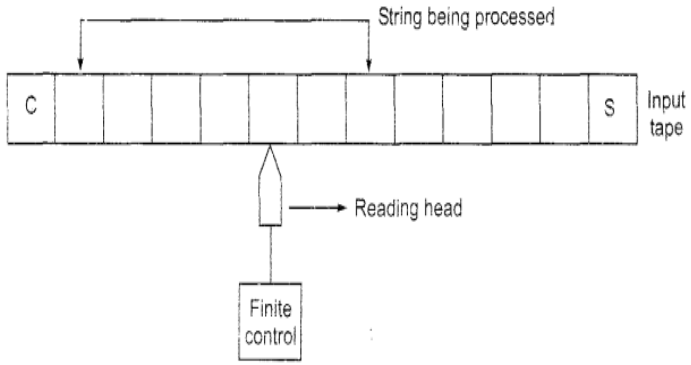


Figure 4: Block diagram of Finite Automata

- Input tape: Each square contains a single symbol from input alphabet Σ.
  - C and $ are end markers of Tape
  - Absence of end markers indicates that the tape is of infinite length

Finite Automata

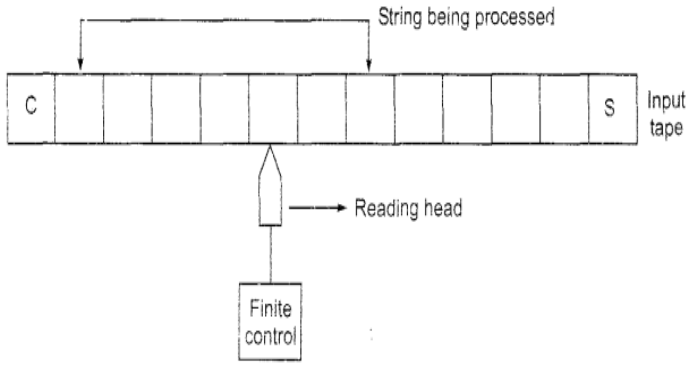


Figure 4: Block diagram of Finite Automata

- Input tape: Each square contains a single symbol from input alphabet Σ.
  - ¢ and $ are end markers of Tape
  - Absence of end markers indicates that the tape is of infinite length
- Reading Head: Examines only 1 $\square$ at a time
  - can move from Left $\rightarrow$ Right *or* Right $\rightarrow$ Left

Finite Automata

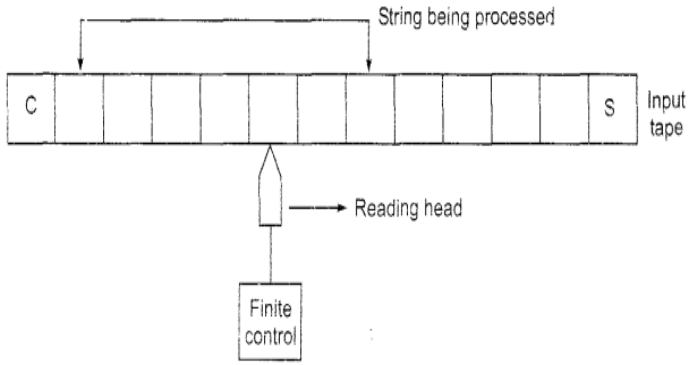
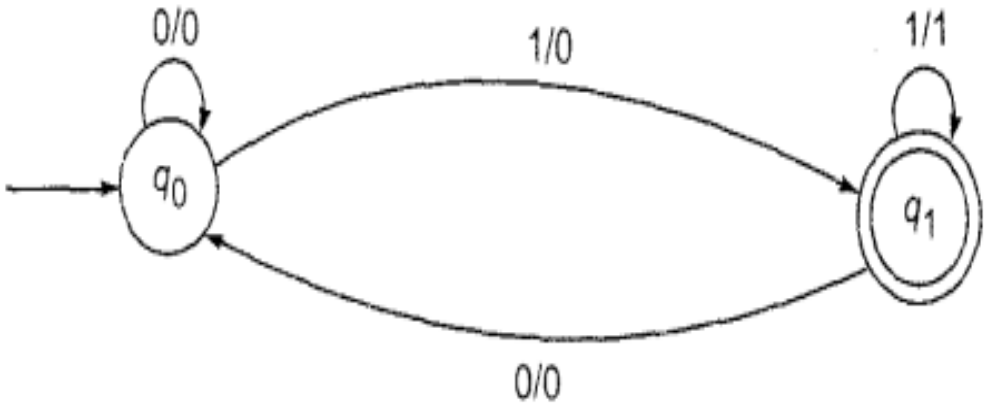


Figure 4: Block diagram of Finite Automata

- Input tape: Each square contains a single symbol from input alphabet Σ.
  - C and $ are end markers of Tape
  - Absence of end markers indicates that the tape is of infinite length
- Reading Head: Examines only 1 $\square$ at a time
  - can move from Left $\rightarrow$ Right *or* Right $\rightarrow$ Left
- Finite Control: Input to a finite control will usually be a symbol under read head.
  Following Outputs:
  - A motion to R-head along the tape on next $\square$
  - Next state of the finite state machine given by $\delta(q, a)$

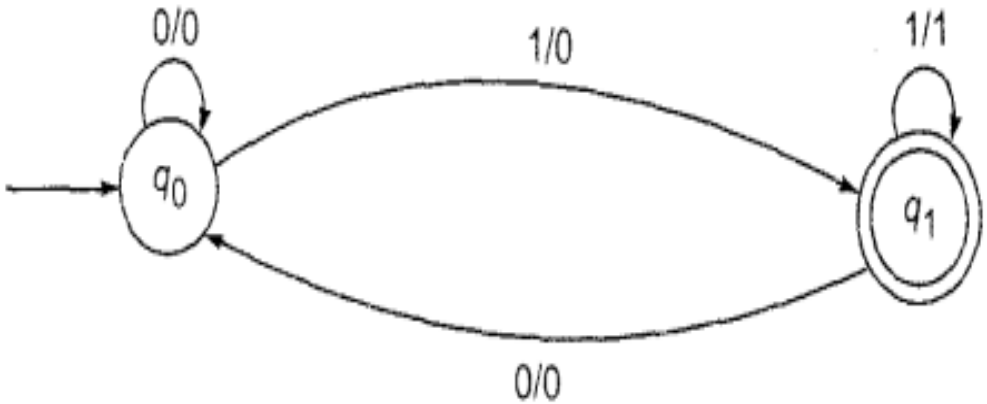
Transition Systems

- A transition system or transition graph is finite directed labelled graph in which each vertex (node) is represented by a state and edges are labelled with input/output.



Transition Systems

- A transition system or transition graph is finite directed labelled graph in which each vertex (node) is represented by a state and edges are labelled with input/output.



- A transition system is a 5-tuple $(Q, \Sigma, \delta, Q_0, F)$ where:
  - Q, Σ, F are finite non-empty set of states, input alphabet and set of final states respectively.
  - $Q_0 \subseteq Q$ and is non-empty
  - δ is a finite subset of $Q \times \Sigma^* \times Q$

Transition system

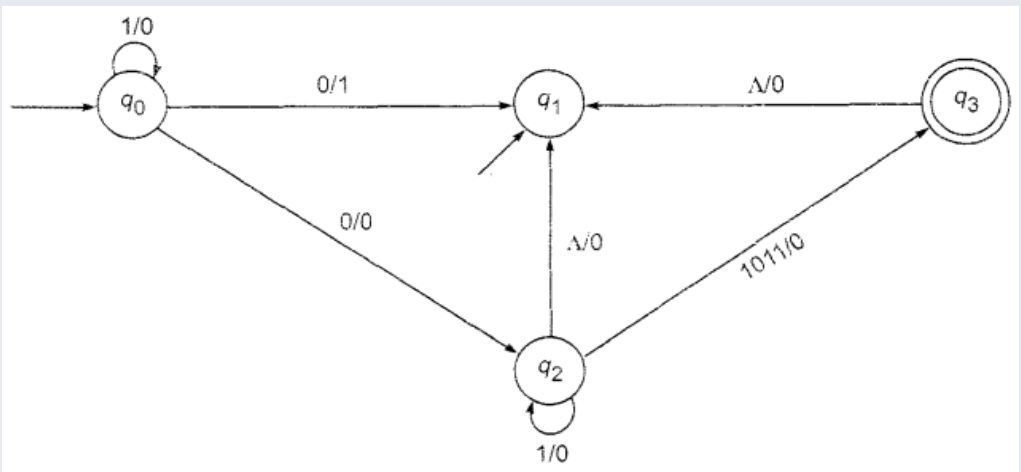
- A transition system accepts a string w in Σ^* if:
  - There exists a path which originates from some initial state, goes along the arrows and terminates at some final state, and
  - The path value obtained by concatenation of all edge-labels of the path is equal to w

Transition system

- A transition system accepts a string w in Σ^* if:
  - There exists a path which originates from some initial state, goes along the arrows and terminates at some final state, and
  - The path value obtained by concatenation of all edge-labels of the path is equal to w

Example

Consider the given transition system:



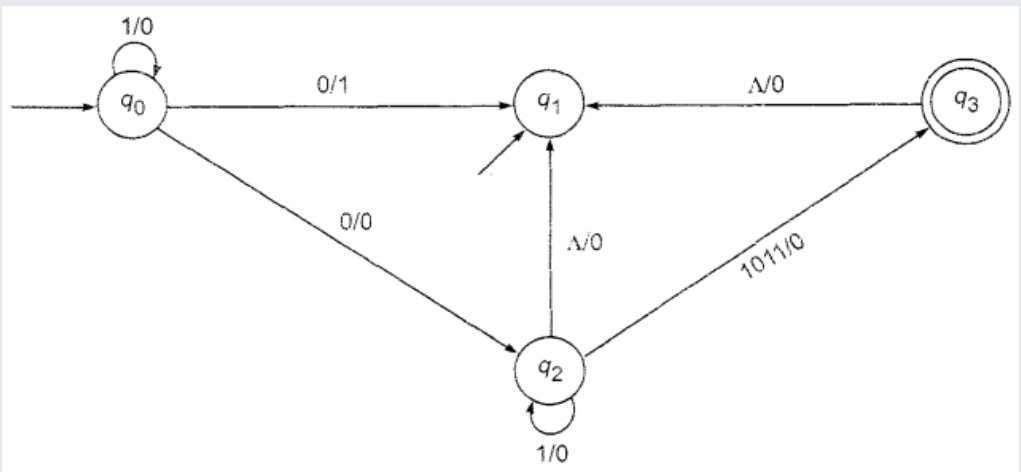
Determine the initial states, final states and acceptability of 101011, 111010.

Transition system

- A transition system accepts a string w in Σ^* if:
  - There exists a path which originates from some initial state, goes along the arrows and terminates at some final state, and
  - The path value obtained by concatenation of all edge-labels of the path is equal to w

Example

Consider the given transition system:



Determine the initial states, final states and acceptability of 101011, 111010.
Initial states: q_0 and q_1; Final State: q_3
Path value $q_0q_0q_2q_3$ for 101011 $\Longrightarrow$ **accepted by system**
But, 111010 not accepted

Transition function

- Every finite automaton (Q, Σ, δ, q_0, F) can be viewd as a transition system (Q, Σ, δ',Q_0,F) if we take $Q_0 = \{q_0\}$ and $\delta' = \{(q,w,\delta(q,w)) | \ q \in Q, \ w \in \Sigma^*\}$
- But, a transition system need not be a finite automaton.
- Example: *A transition system may contain more than one initial state*.

Transition function

- Every finite automaton (Q, Σ, δ, q_0, F) can be viewd as a transition system (Q, Σ, δ',Q_0,F) if we take $Q_0 = \{q_0\}$ and $\delta' = \{(q,w,\delta(q,w)) | \ q \in Q, \ w \in \Sigma^*\}$
- But, a transition system need not be a finite automaton.
- Example: *A transition system may contain more than one initial state*.
- **Properties of Transition Functions:**
  1. $\delta(q, \wedge) = q$ is a finite automaton
     $\Longrightarrow$ State of the system can be changed only by an input symbol.
  2. For all strings w and input symbol a:
     $\delta(q, aw) = \delta(\delta(q, a), w)$
     $\delta(q, wa) = \delta(\delta(q, w), a)$

Transition function

- Every finite automaton (Q, Σ, δ, q_0, F) can be viewd as a transition system (Q, Σ, δ',Q_0,F) if we take $Q_0 = \{q_0\}$ and $\delta' = \{(q,w,\delta(q,w)) | \ q \in Q, \ w \in \Sigma^*\}$
- But, a transition system need not be a finite automaton.
- Example: *A transition system may contain more than one initial state.*
- **Properties of Transition Functions:**
  1. $\delta(q, \wedge) = q$ is a finite automaton
     $\Longrightarrow$ State of the system can be changed only by an input symbol.
  2. For all strings w and input symbol a:
     $\delta(q, aw) = \delta(\delta(q, a), w)$
     $\delta(q, wa) = \delta(\delta(q, w), a)$

Exercise:

Prove that for any transition function δ and for any two input string x and y
$\delta(q, xy) = \delta(\delta(q, x), y)$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$
$= \delta(q_2, \mathbf{1}01)$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$
$= \delta(q_2, \mathbf{1}01)$
$= \delta(q_3, \mathbf{0}1)$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$
$= \delta(q_2, \mathbf{1}01)$
$= \delta(q_3, \mathbf{0}1)$
$= \delta(q_1, \mathbf{1})$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$
$= \delta(q_2, \mathbf{1}01)$
$= \delta(q_3, \mathbf{0}1)$
$= \delta(q_1, \mathbf{1})$
$= \delta(q_0, \wedge)$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$
$= \delta(q_2, \mathbf{1}01)$
$= \delta(q_3, \mathbf{0}1)$
$= \delta(q_1, \mathbf{1})$
$= \delta(q_0, \wedge)$
$= q_0$

Acceptability of a String by a finite automaton

- A string 'x' is accepted by a finite automaton M = $(Q, \Sigma, \delta, q_0, F)$ if $\delta(q_0, x) = q$ for some $q \in F$

Example

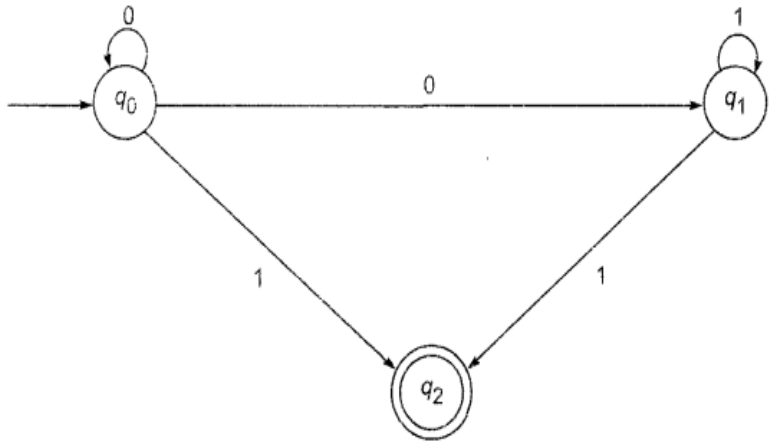
Consider the finite state machine whose transition function δ is given below in form of a transition table. Here, Q=$\{q_0, q_1, q_2, q_3\}$, $\Sigma = \{0, 1\}$, F = $\{q_0\}$.
Give the entire sequence of states for the input string 110101.

	Input	
State	0	1
$\rightarrow \textcircled{q_0}$	q_2	q_1
q_1	q_3	q_0
q_2	q_0	q_3
q_3	q_1	q_2

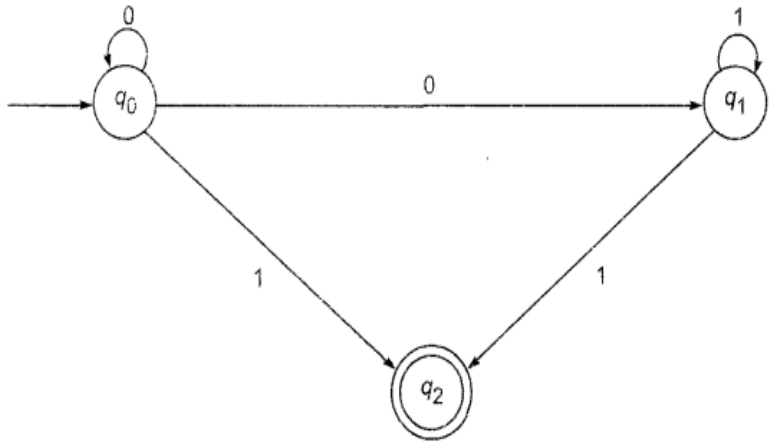
Answer: $\delta(q_0, \mathbf{1}10101) = \delta(q_1, \mathbf{1}0101)$
$= \delta(q_0, \mathbf{0}101)$
$= \delta(q_2, \mathbf{1}01)$
$= \delta(q_3, \mathbf{0}1)$
$= \delta(q_1, \mathbf{1})$
$= \delta(q_0, \wedge)$
$= q_0$

Hence, $q_0 \xrightarrow{1} q_1 \xrightarrow{1} q_0 \xrightarrow{0} q_2 \xrightarrow{1} q_3 \xrightarrow{0} q_1 \xrightarrow{1} q_0$ Accepted

Non-Deterministic Finite State Machines



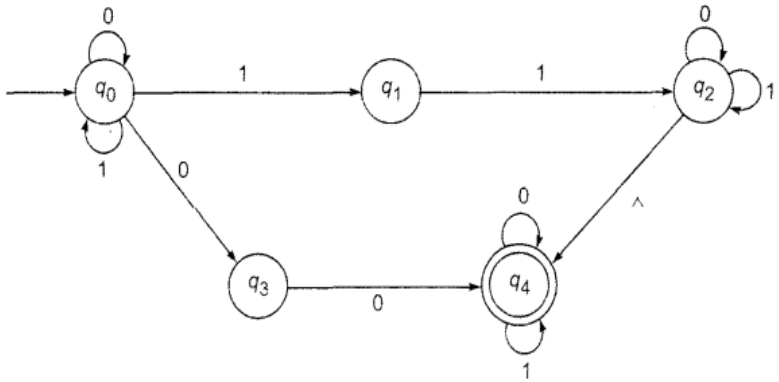
Non-Deterministic Finite State Machines



- A non-deterministic finite automaton (NDFA) is a 5-tuple $(Q, \Sigma, \delta, q_0, F)$ where
  Q is a finite non-empty set of states
  Σ is a finite non-empty set of inputs
  δ is a transition function mapping from $Q \times \Sigma$ into 2^Q which is power set of Q, the set of all subsets of Q
  $q_0 \in Q$ is the initial state
  $F \subseteq Q$ is the set of final states.

Non-Deterministic Finite State Machines

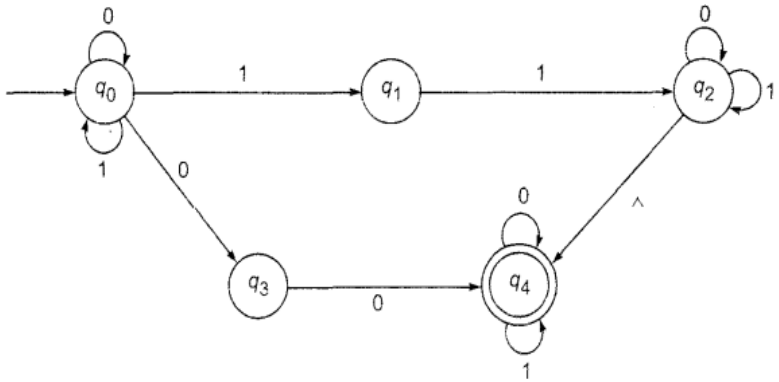
- Consider a Non-deterministic automaton as under:



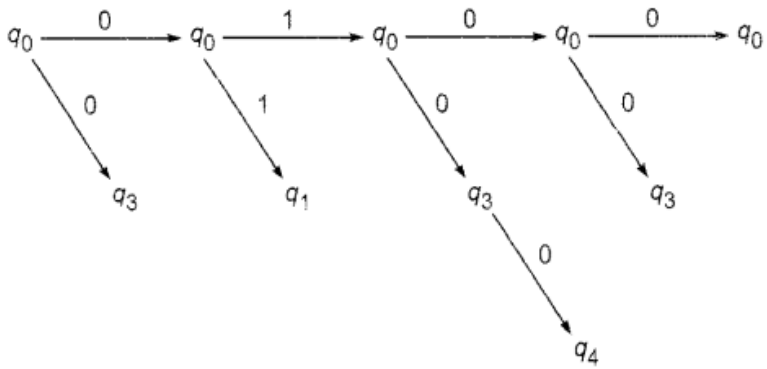
Determine the sequence of states for input string 0100

Non-Deterministic Finite State Machines

- Consider a Non-deterministic automaton as under:

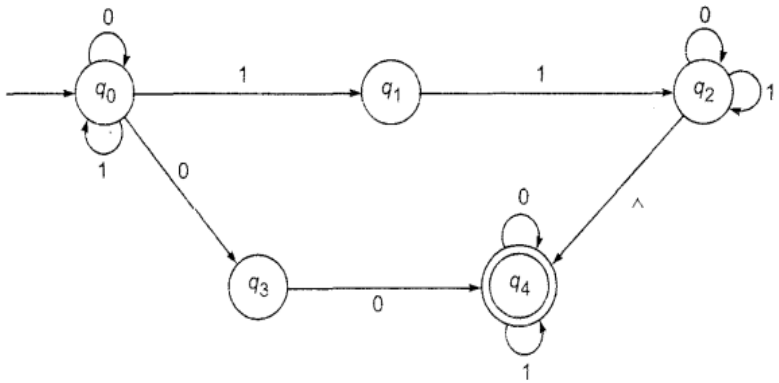


Determine the sequence of states for input string 0100

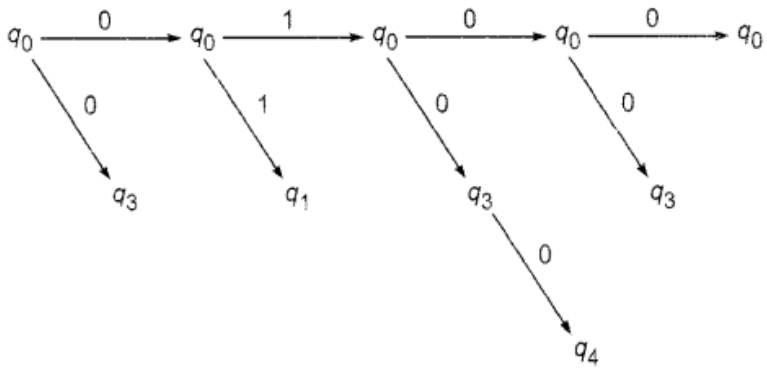


Non-Deterministic Finite State Machines

- Consider a Non-deterministic automaton as under:



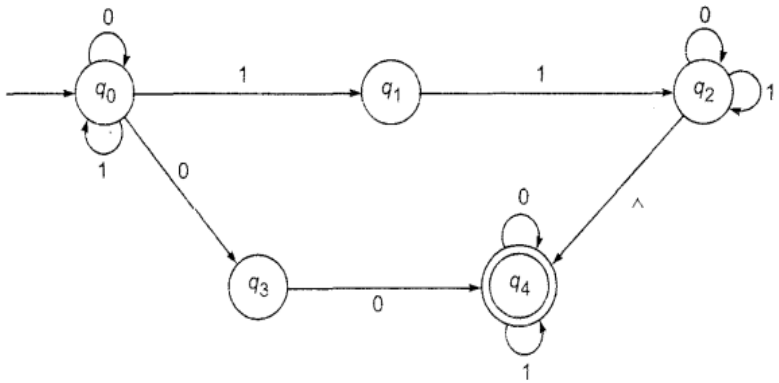
Determine the sequence of states for input string 0100



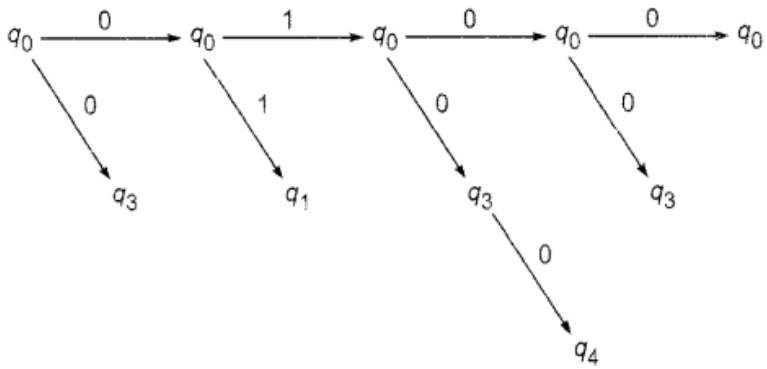
$\delta(q_0, 0100) = \{q_0, q_3, q_4\}$
Since q_4 is the final state. $\therefore$ input string 0100 is accepted by the system.

Non-Deterministic Finite State Machines

- Consider a Non-deterministic automaton as under:



Determine the sequence of states for input string 0100



$\delta(q_0, 0100) = \{q_0, q_3, q_4\}$
Since q_4 is the final state. $\therefore$ input string 0100 is accepted by the system.

- A string $w \in \Sigma^*$ is accepted by NDFA "M". If $\delta(q_0, w)$ contains some final state.

Equivalence of DFA and NDFA

- A DFA can simulate the behaviour of NDFA by increasing the number of states
- DFA (Q, Σ, δ, q_0, F) can be viewed as NDFA (Q, Σ, δ', q_0, F)
- Any NDFA is a more general machine without being more powerful.

  $\Longrightarrow$ For every NDFA, there exists a DFA which simulates the behaviour of NDFA.

Equivalence of DFA and NDFA

- A DFA can simulate the behaviour of NDFA by increasing the number of states
- DFA (Q, Σ, δ, q_0, F) can be viewed as NDFA (Q, Σ, δ', q_0, F)
- Any NDFA is a more general machine without being more powerful.

  $\Longrightarrow$ For every NDFA, there exists a DFA which simulates the behaviour of NDFA. **Alternatively, if L is a set accepted by NDFA, then there exists a DFA which also accepts L.**

Example

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled) q_0	q_0	q_1
q_1	q_1	q_0, q_1

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$		

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled) q_0	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$		

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled) q_0	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$		

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0, q_1]$

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (q0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0, q_1]$
$[q_o, q_1]$		

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (q0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0, q_1]$
$[q_o, q_1]$	$[q_o, q_1]$	

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (q0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0, q_1]$
$[q_o, q_1]$	$[q_o, q_1]$	$[q_o, q_1]$

NDFA $\rightarrow$ DFA

Example 1

Contruct a deterministic automaton equivalent to M=($\{q_0, q_1\}$, $\{0, 1\}$, δ, q_0, $\{q_0\}$) where δ is defined as under:

State	Input	
	0	1
$\rightarrow$ (circled q_0)	q_0	q_1
q_1	q_1	q_0, q_1

Solution: For the deterministic automaton M_1:

- The states are subsets of $\{q_0, q_1\}$
  $\Longrightarrow$ ϕ, $[q_0]$, $[q_1]$, $[q_0, q_1]$
- $[q_0]$ is initial state.
- $[q_0]$ and $[q_0, q_1]$ are final states as these are the only states containing q_0
- δ is defined by state table as under:

State	Input	
	0	1
$[\phi]$	$[\phi]$	$[\phi]$
$[q_0]$	$[q_0]$	$[q_1]$
$[q_1]$	$[q_1]$	$[q_0, q_1]$
$[q_o, q_1]$	$[q_o, q_1]$	$[q_o, q_1]$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$		

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$		

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$		

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$		

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_1, q_2]$

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_1, q_2]$
$[q_1, q_2]$		

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_0]$	

NDFA $\rightarrow$ DFA

Example 2

Find a deterministic acceptor equivalent to:
M= $(\{q_0,q_1,q_2\},\{a,b\}, \delta, q_0, \{q_2\})$
where δ is given by

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_2
q_1	q_0	q_1
$\textcircled{q_2}$	ϕ	q_0, q_1

Solution: The deterministic automaton M_1 equivalent to M is defined as follows:
$M_1=(2^Q,\{a,b\}, \delta, [q_0], F')$
where, $F'= \{[q_2],[q_0, q_2],[q_1, q_2],[q_0, q_1, q_2]\}$

State	Input	
	a	b
$[q_0]$	$[q_0, q_1]$	$[q_2]$
$[q_1]$	$[q_0]$	$[q_1]$
$[q_2]$	ϕ	$[q_0, q_1]$
$[q_0, q_1]$	$[q_0, q_1]$	$[q_1, q_2]$
$[q_1, q_2]$	$[q_0]$	$[q_0, q_1]$

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
((q_3))		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$		

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
(@)q_3		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$		

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
(@)q_3		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$		

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$		

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
(@)q_3		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_2, q_3]$		

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$	

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$

NDFA $\rightarrow$ DFA

Example 3

Construct a deterministic finite automaton equivalent to
M=($\{q_0, q_1, q_2, q_3\}$,{a,b},δ,q_0,$\{q_3\}$)
where δ is as under:

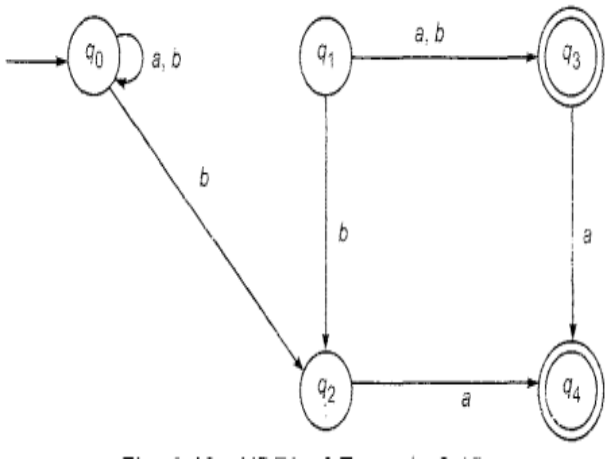
State	Input	
	a	b
$\rightarrow q_0$	q_0, q_1	q_0
q_1	q_2	q_1
q_2	q_3	q_3
$\textcircled{q_3}$		q_2

Solution: Let Q = $\{q_0, q_1, q_2, q_3\}$, then the deterministic automaton M_1 equivalent to M is given by M_1=(2^Q,{a,b},δ,$[q_0]$,F)
where, F consists of:
$\{[q_3],[q_0, q_3],[q_1, q_3],[q_2, q_3],[q_0, q_1, q_3],[q_0, q_2, q_3],[q_1, q_2, q_3],[q_0, q_1, q_2, q_3]\}$
and δ is defined by state table as under:

State	Input	
	a	b
$[q_0]$	$[q_o, q_1]$	$[q_0]$
$[q_0, q_1]$	$[q_0, q_1, q_2]$	$[q_0, q_1]$
$[q_0, q_1, q_2]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_3]$
$[q_0, q_1, q_3]$	$[q_0, q_1, q_2]$	$[q_0, q_1, q_2]$
$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$	$[q_0, q_1, q_2, q_3]$

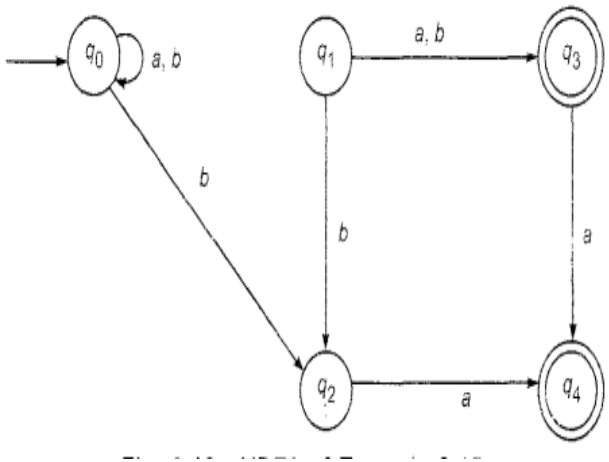
NDFA $\rightarrow$ DFA

1. Construct a DFA equivalent to NDFA 'M' whose transition diagram is given as:



NDFA $\rightarrow$ DFA

1. Construct a DFA equivalent to NDFA 'M' whose transition diagram is given as:



2. Construct a DFA equivalent to NDFA with initial state q_0 whose transition table is defined as

State	a	b
q_0	q_1, q_3	q_2, q_3
q_1	q_1	q_3
q_2	q_3	q_2
(q_3)	–	–

Constructing required DFA

Example 1

Construct a DFA accepting all strings 'w' over $\{0,1\}$ such that the number of 1's in 'w' is 3mod4.

Constructing required DFA

Example 1

Construct a DFA accepting all strings 'w' over $\{0,1\}$ such that the number of 1's in 'w' is 3mod4.

Solution: Let the required DFA, as the condition on strings of T(M) doesn't at all involve 0,

$\Longrightarrow$ M doesnot change the state on input 0.

Constructing required DFA

Example 1

Construct a DFA accepting all strings 'w' over $\{0,1\}$ such that the number of 1's in 'w' is 3mod4.

Solution: Let the required DFA, as the condition on strings of T(M) doesn't at all involve 0,

$\Longrightarrow$ M doesnot change the state on input 0.

If 1 appears in w (4k+3) times, M can come back to initial state, after reading 4 1's and to a final state after reading 3 1's.

Constructing required DFA

Example 1

Construct a DFA accepting all strings 'w' over $\{0,1\}$ such that the number of 1's in 'w' is 3mod4.

Solution: Let the required DFA, as the condition on strings of T(M) doesn't at all involve 0,

$\Longrightarrow$ M doesnot change the state on input 0.

If 1 appears in w (4k+3) times, M can come back to initial state, after reading 4 1's and to a final state after reading 3 1's.

The required DFA:

Constructing required DFA

Example 1

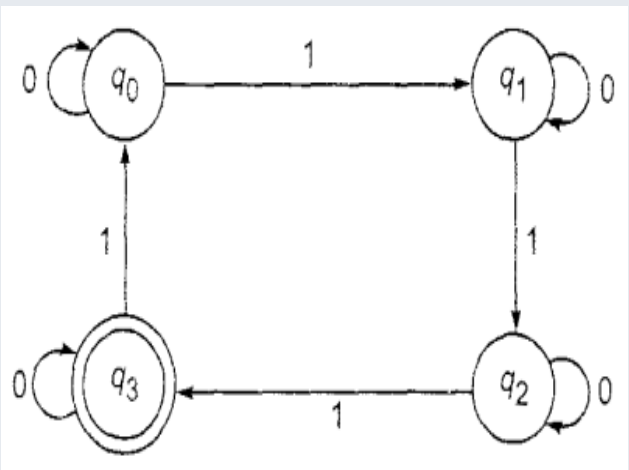
Construct a DFA accepting all strings 'w' over $\{0,1\}$ such that the number of 1's in 'w' is 3mod4.

Solution: Let the required DFA, as the condition on strings of T(M) doesn't at all involve 0,

$\Longrightarrow$ M doesnot change the state on input 0.

If 1 appears in w (4k+3) times, M can come back to initial state, after reading 4 1's and to a final state after reading 3 1's.

The required DFA:

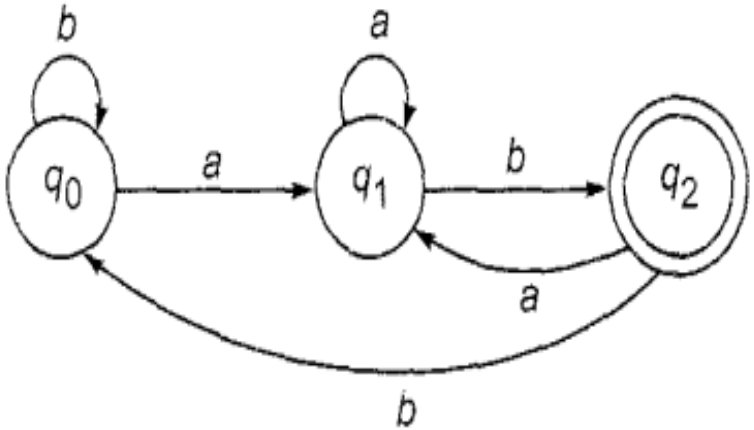


Constructing required DFA

1. Construct a DFA accepting all strings over {a,b} ending in *ab*.

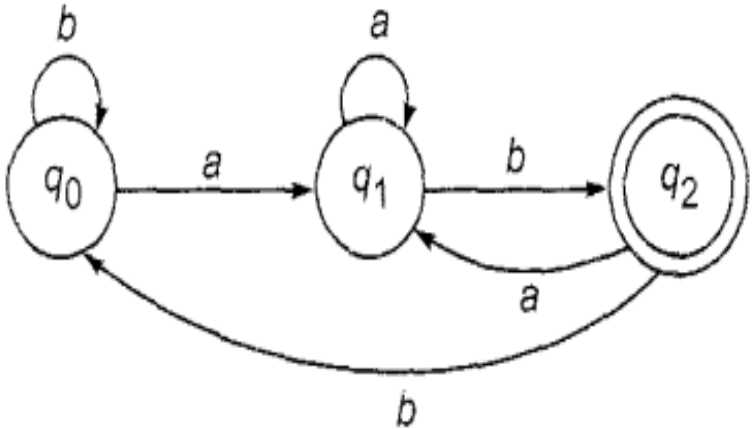
Constructing required DFA

1. Construct a DFA accepting all strings over {a,b} ending in *ab*.



Constructing required DFA

1. Construct a DFA accepting all strings over {a,b} ending in *ab*.



2. Construct a DFA equivalent to NDFA for:

State	Input		
	0	1	$\wedge$
$\rightarrow q_0$	q_0, q_3	q_0, q_1	
q_1	q_2		
q_2	q_2	q_2	q_4
q_3	q_4		
$\textcircled{q_4}$	q_4	q_4	

Constructing required DFA

1. M=($\{q_1, q_2, q_3\}$,{0,1}, δ,q_1,$\{q_3\}$) is a NDFA where δ is given by:
   $\delta(q_1, 0)=\{q_2, q_3\}$
   $\delta(q_1, 1)=\{q_1\}$
   $\delta(q_2, 0)=\{q_1, q_2\}$
   $\delta(q_2, 1)=\phi$
   $\delta(q_3, 0)=\{q_2\}$
   $\delta(q_3, 1)=\{q_1, q_2\}$
   Construct equivalent DFA.

Finite Automata with Outputs

- Moore Machine is a 6-tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
  where Q is a finite set of states
  - Σ is the input alphabet
  - Δ is the output alphabet
  - δ is the transition function $Q \times \Sigma$ **into Q**
  - λ is the output function **Q into** Δ
  - q_0 is the initial state

Finite Automata with Outputs

- Moore Machine is a 6-tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
  where Q is a finite set of states
  Σ is the input alphabet
  Δ is the output alphabet
  δ is the transition function $Q \times \Sigma$ **into Q**
  λ is the output function **Q into** Δ
  q_0 is the initial state

Example:

Initial state q_0 is marked with an arrow. The table defines δ and λ:

Present State	Next State a=0	Next State a=1	Output λ
$\rightarrow$ q_0	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

Determine transition states and output string for input string 0111.

Finite Automata with Outputs

- Moore Machine is a 6-tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
  where Q is a finite set of states
  Σ is the input alphabet
  Δ is the output alphabet
  δ is the transition function $Q \times \Sigma$ **into Q**
  λ is the output function **Q into** Δ
  q_0 is the initial state

Example:

Initial state q_0 is marked with an arrow. The table defines δ and λ:

Present State	Next State a=0	Next State a=1	Output λ
$\rightarrow$ $\textcircled{q_0}$	q_3	q_1	0
q_1	q_1	q_2	1
q_2	q_2	q_3	0
q_3	q_3	q_0	0

Determine transition states and output string for input string 0111.
Solution: Transition states:
$q_0 \xrightarrow{0\backslash 0} q_3 \xrightarrow{1\backslash 0} q_0 \xrightarrow{1\backslash 0} q_1 \xrightarrow{1\backslash 1} q_2$ 0
OutputString: 00010

Finite Automata with Outputs

- Mealy Machine is a 6-tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
  where Q is a finite set of states
  - Σ is the input alphabet
  - Δ is the output alphabet
  - δ is the transition function $Q \times \Sigma$ **into Q**
  - λ is the output function mapping **Q** $\times \Sigma$ **into** Δ
  - q_0 is the initial state

Finite Automata with Outputs

- Mealy Machine is a 6-tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
  where Q is a finite set of states
  Σ is the input alphabet
  Δ is the output alphabet
  δ is the transition function $Q \times \Sigma$ **into Q**
  λ is the output function mapping **Q** $\times\Sigma$ **into** Δ
  q_0 is the initial state

Example:

Consider a mealy machine for q_1 as initial state.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

Determine the transition of states and corresponding output string for input string 0011.

Finite Automata with Outputs

- Mealy Machine is a 6-tuple $(Q, \Sigma, \Delta, \delta, \lambda, q_0)$
  where Q is a finite set of states
  Σ is the input alphabet
  Δ is the output alphabet
  δ is the transition function $Q \times \Sigma$ **into Q**
  λ is the output function mapping **Q** $\times\Sigma$ **into** Δ
  q_0 is the initial state

Example:

Consider a mealy machine for q_1 as initial state.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

Determine the transition of states and corresponding output string for input string 0011.

Solution: $q_1 \xrightarrow{0\backslash 0} q_3 \xrightarrow{0\backslash 1} q_2 \xrightarrow{1\backslash 0} q_4 \xrightarrow{1\backslash 0} q_3$
Output String: 0100

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

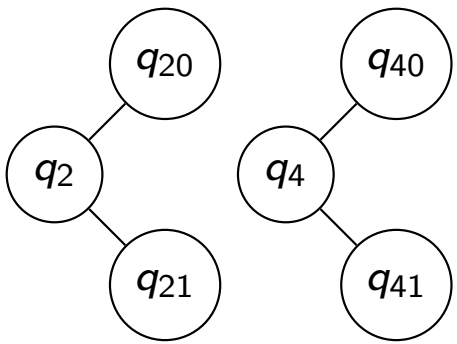
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**

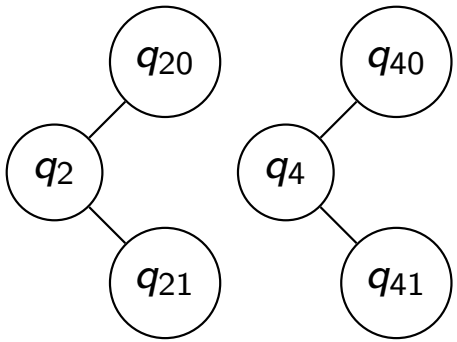


Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



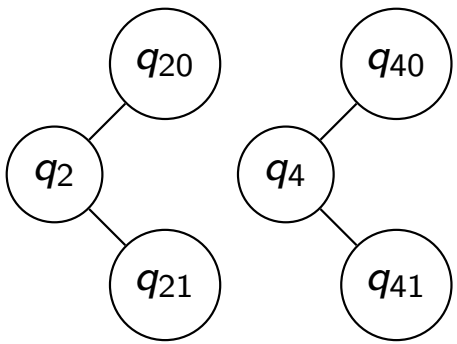
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$				
q_{20}				
q_{21}				
q_3				
q_{40}				
q_{41}				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



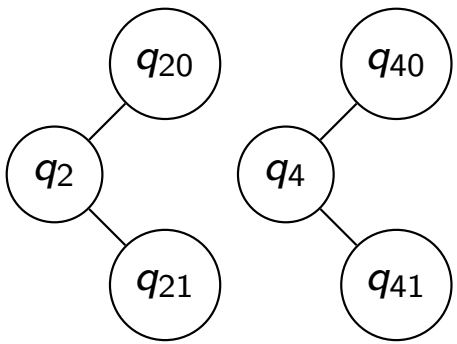
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



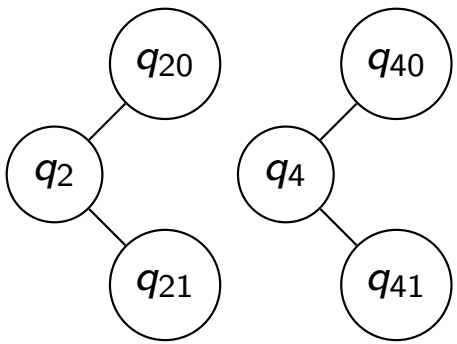
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3			

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



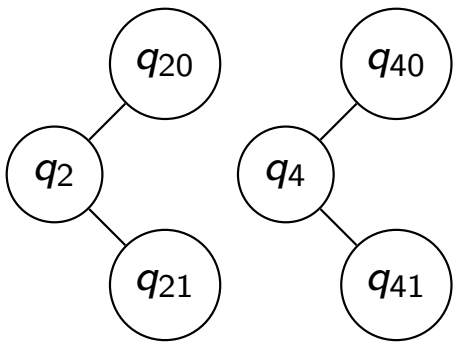
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0		

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



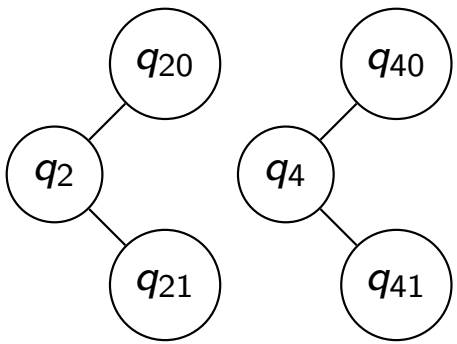
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



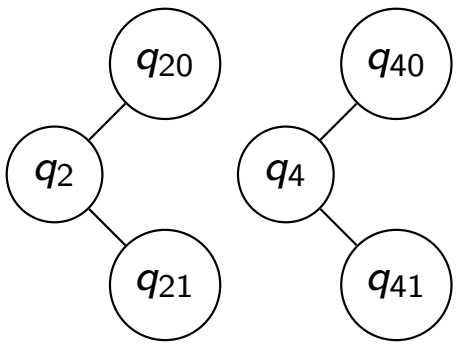
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



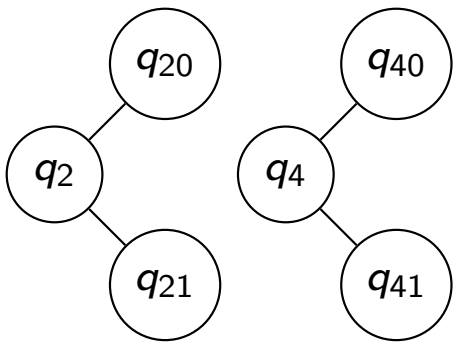
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1			

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



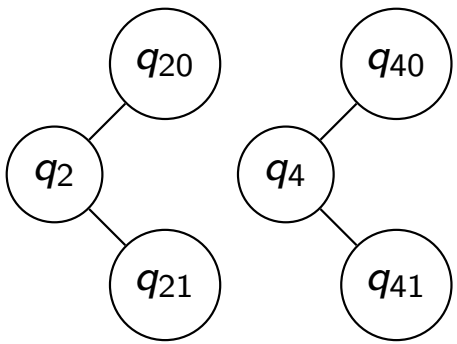
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1		

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



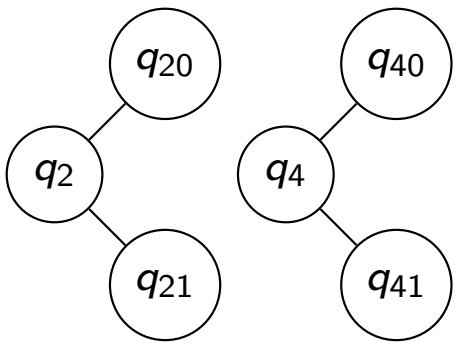
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



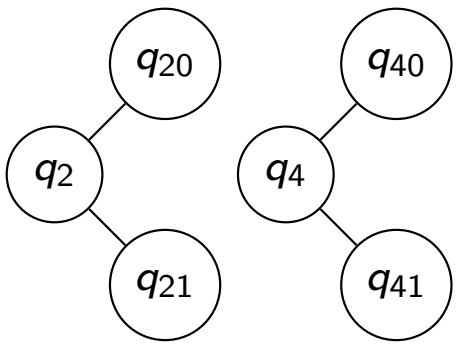
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



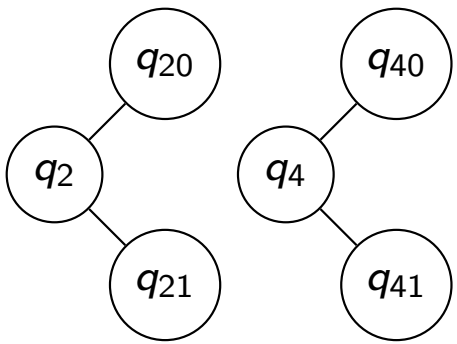
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1			

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



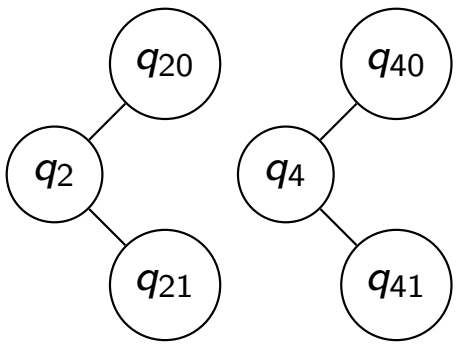
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1		

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



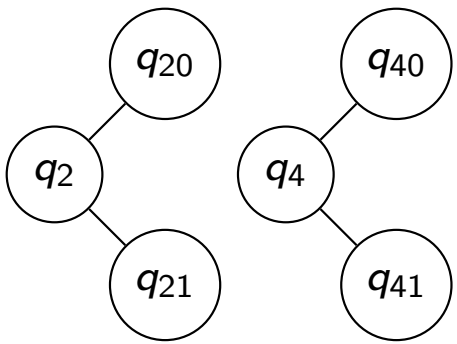
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



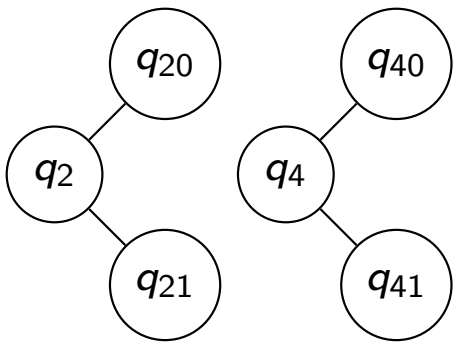
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



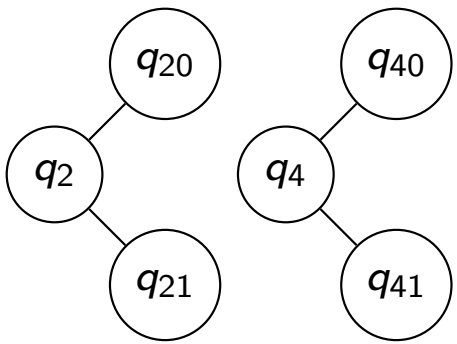
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}			

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



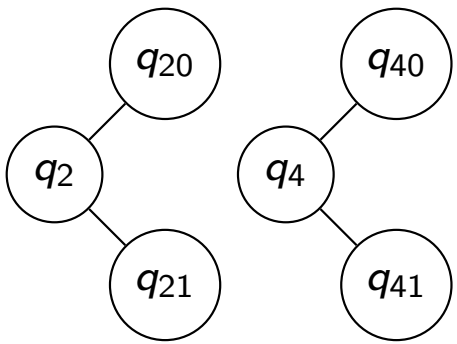
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1		

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



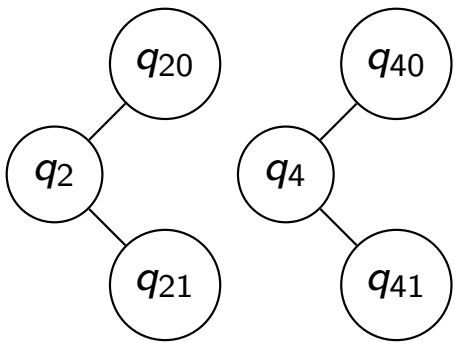
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



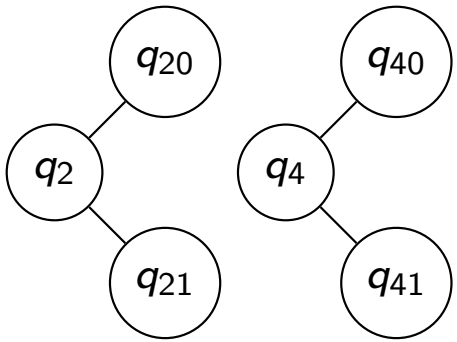
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



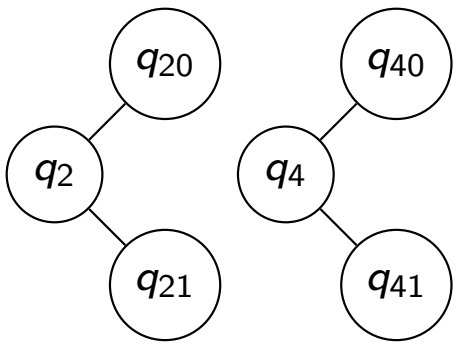
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}			

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



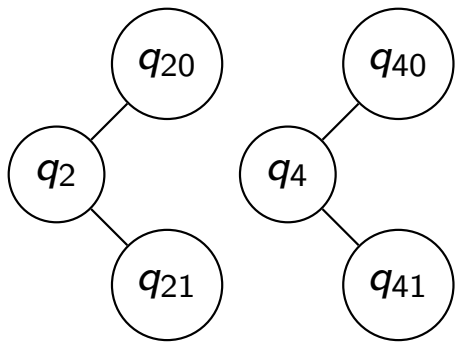
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1		

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**

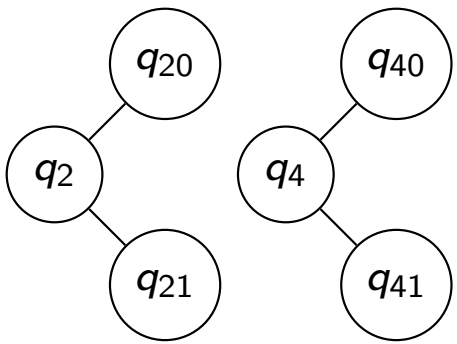


Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



- **Solution:**

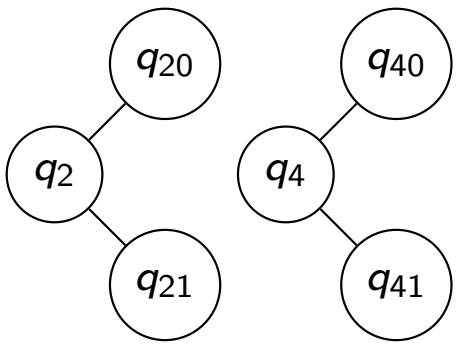
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}				

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**



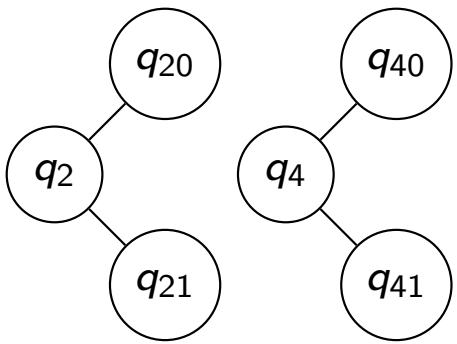
Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}	q_{41}			

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

- **Solution:**

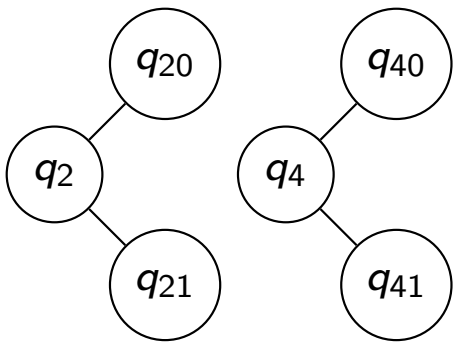


Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}	q_{41}	1		

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



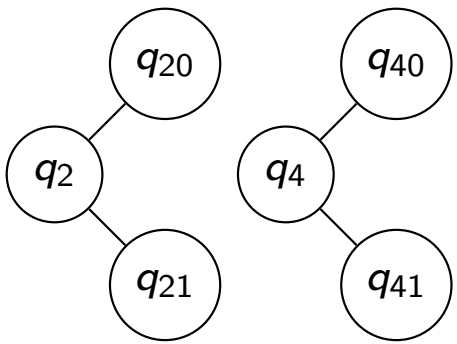
- **Solution:**

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}	q_{41}	1	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



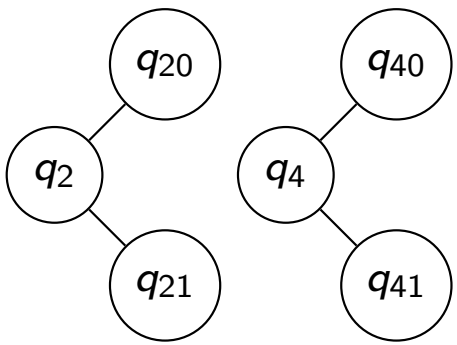
- **Solution:**

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_{20}	0
q_{20}	q_1	1	q_{40}	0
q_{21}	q_1	1	q_{40}	0
q_3	q_{21}	1	q_1	1
q_{40}	q_{41}	1	q_3	0
q_{41}	q_{41}	1	q_3	0

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



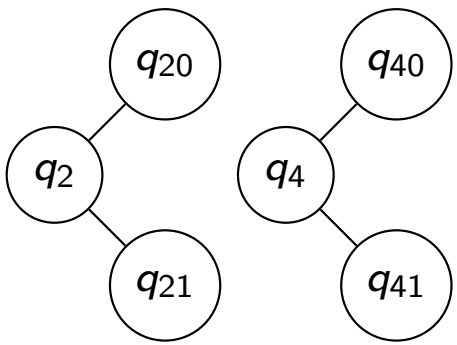
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	
q_{20}	q_1	q_{40}	
q_{21}	q_1	q_{40}	
q_3	q_{21}	q_1	
q_{40}	q_{41}	q_3	
q_{41}	q_{41}	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



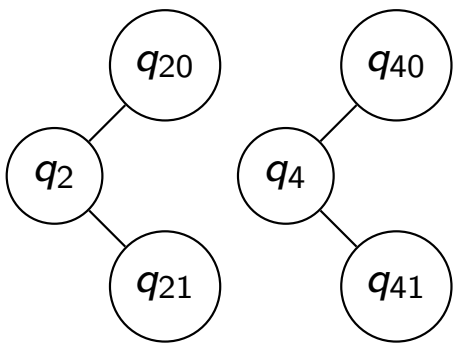
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	
q_{21}	q_1	q_{40}	
q_3	q_{21}	q_1	
q_{40}	q_{41}	q_3	
q_{41}	q_{41}	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



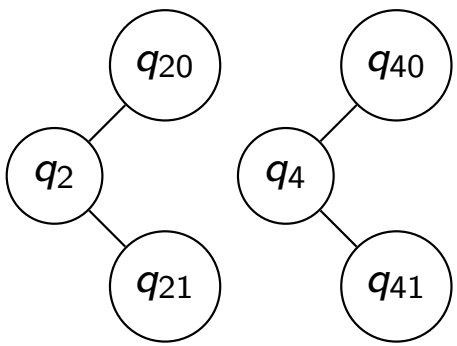
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	
q_3	q_{21}	q_1	
q_{40}	q_{41}	q_3	
q_{41}	q_{41}	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



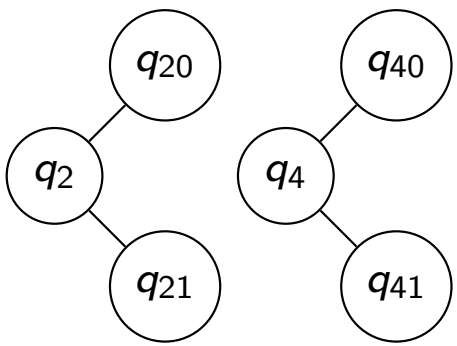
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	
q_{40}	q_{41}	q_3	
q_{41}	q_{41}	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



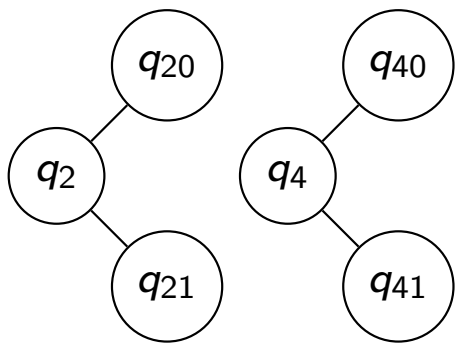
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	
q_{41}	q_{41}	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



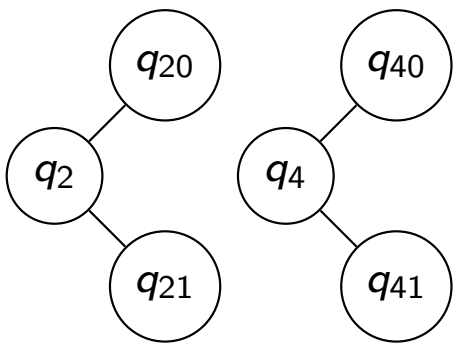
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0



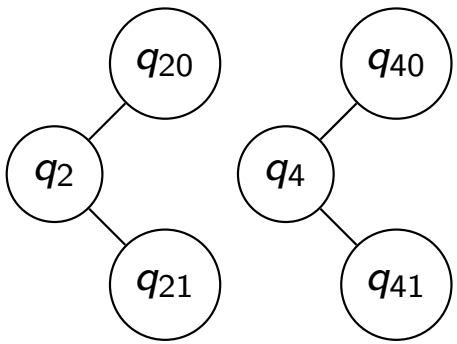
- **Solution:**

Present	a=0	a=1	Output
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	1

Procedure of transforming Mealy machine into Moore machine

- Consider the mealy machie described by given transition table. Construct a moore machine which is equivalent to given mealy machine.

Present State	a=0		a=1	
	State	Output	State	Output
$\rightarrow q_1$	q_3	0	q_2	0
q_2	q_1	1	q_4	0
q_3	q_2	1	q_1	1
q_4	q_4	1	q_3	0

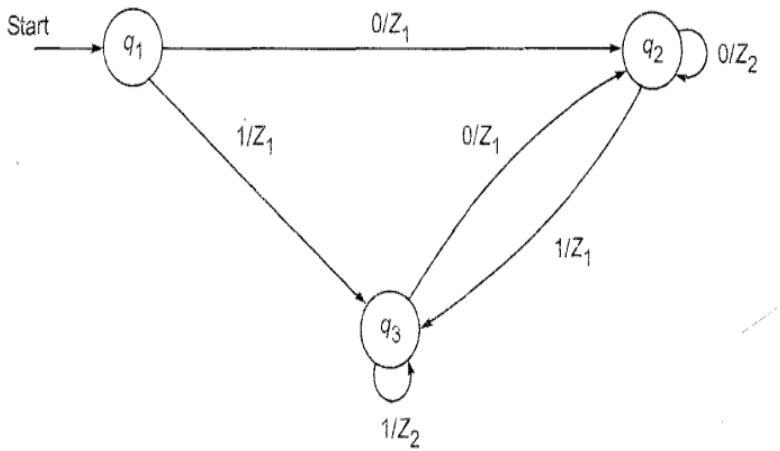


- **Solution:**

Present	a=0	a=1	Output
$\rightarrow q_0$	q_3	q_{20}	0
q_1	q_3	q_{20}	1
q_{20}	q_1	q_{40}	0
q_{21}	q_1	q_{40}	1
q_3	q_{21}	q_1	0
q_{40}	q_{41}	q_3	0
q_{41}	q_{41}	q_3	1

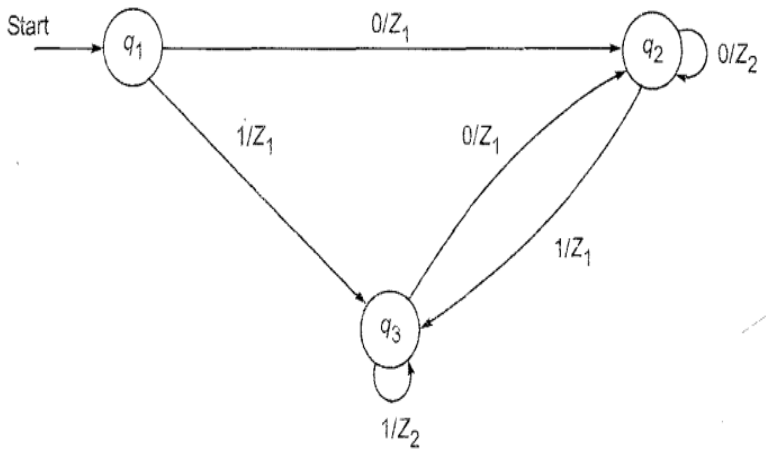
Procedure of transforming Mealy machine into Moore machine

Convert the given mealy machine into equivalent moore machine



Procedure of transforming Mealy machine into Moore machine

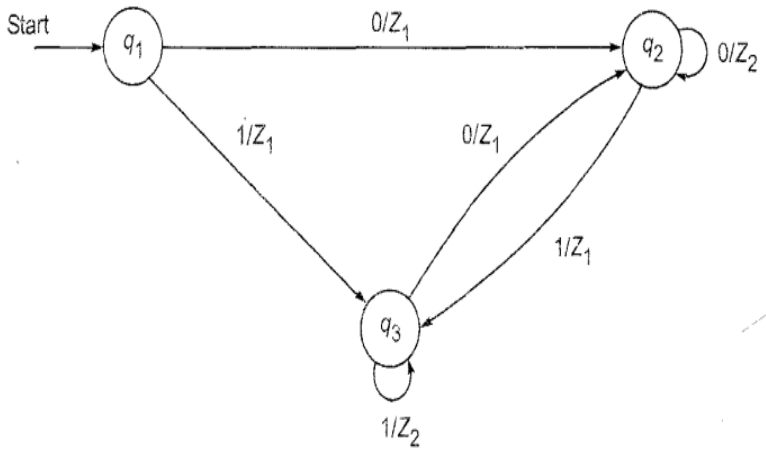
Convert the given mealy machine into equivalent moore machine



Present state	Next state			
	a = 0		a = 1	
	state	output	state	output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2

Procedure of transforming Mealy machine into Moore machine

Convert the given mealy machine into equivalent moore machine

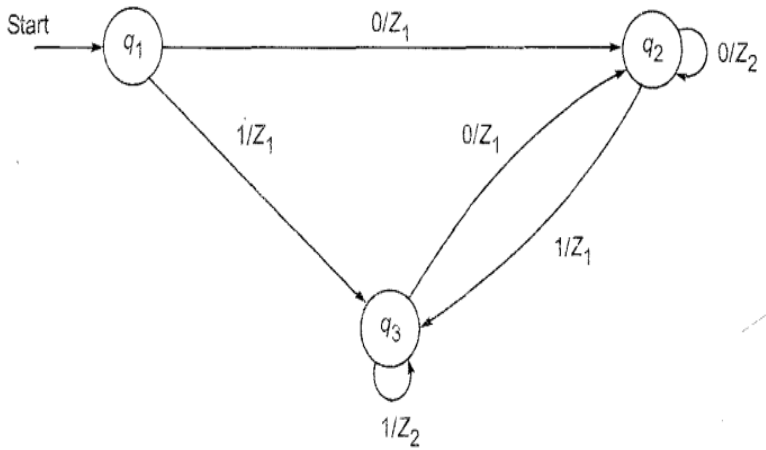


Present state	Next state			
	a = 0		a = 1	
	state	output	state	output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2

Present state	Next state		Output
	a = 0	a = 1	
$\rightarrow q_1$	q_{21}	q_{31}	
q_{21}	q_{22}	q_{31}	Z_1
q_{22}	q_{22}	q_{31}	Z_2
q_{31}	q_{21}	q_{32}	Z_1
q_{32}	q_{21}	q_{32}	Z_2

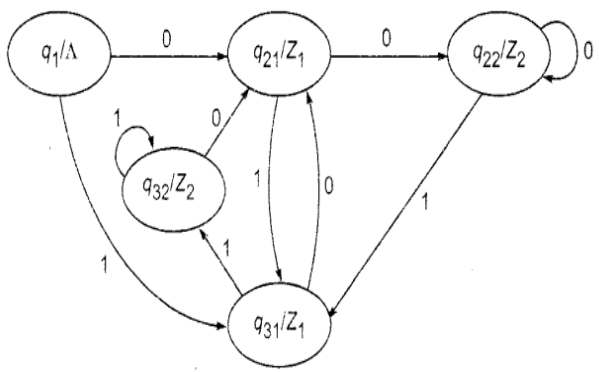
Procedure of transforming Mealy machine into Moore machine

Convert the given mealy machine into equivalent moore machine

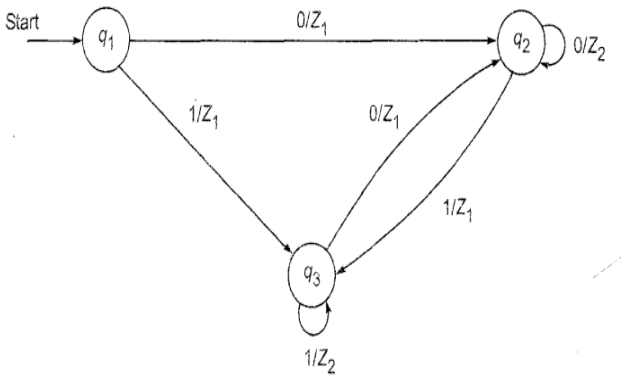


Present state	Next state			
	a = 0		a = 1	
	state	output	state	output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2

Present state	Next state		Output
	a = 0	a = 1	
$\rightarrow q_1$	q_{21}	q_{31}	
q_{21}	q_{22}	q_{31}	Z_1
q_{22}	q_{22}	q_{31}	Z_2
q_{31}	q_{21}	q_{32}	Z_1
q_{32}	q_{21}	q_{32}	Z_2

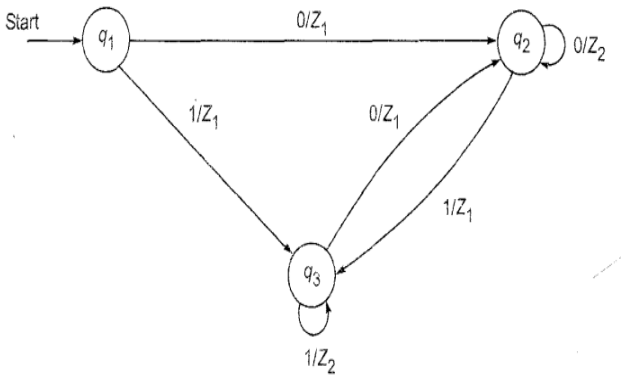


Procedure of transforming Mealy machine into Moore machine

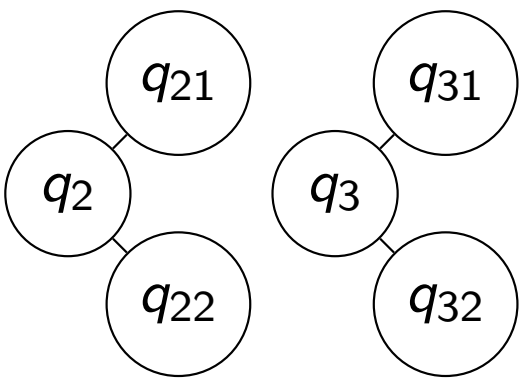


Present State	a=0		a=1	
	state	Output	State	Output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2

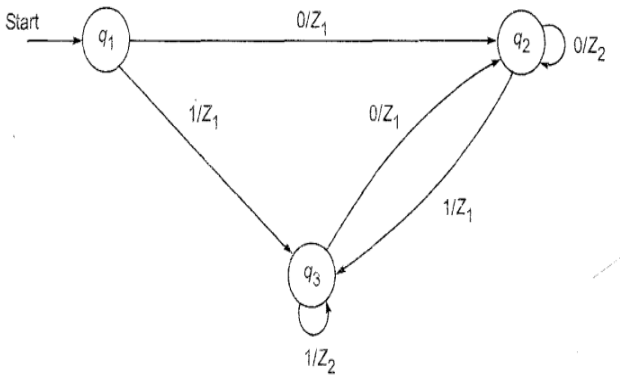
Procedure of transforming Mealy machine into Moore machine



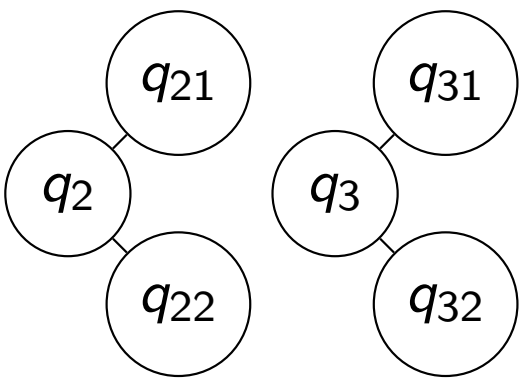
Present State	a=0		a=1	
	state	Output	State	Output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2



Procedure of transforming Mealy machine into Moore machine



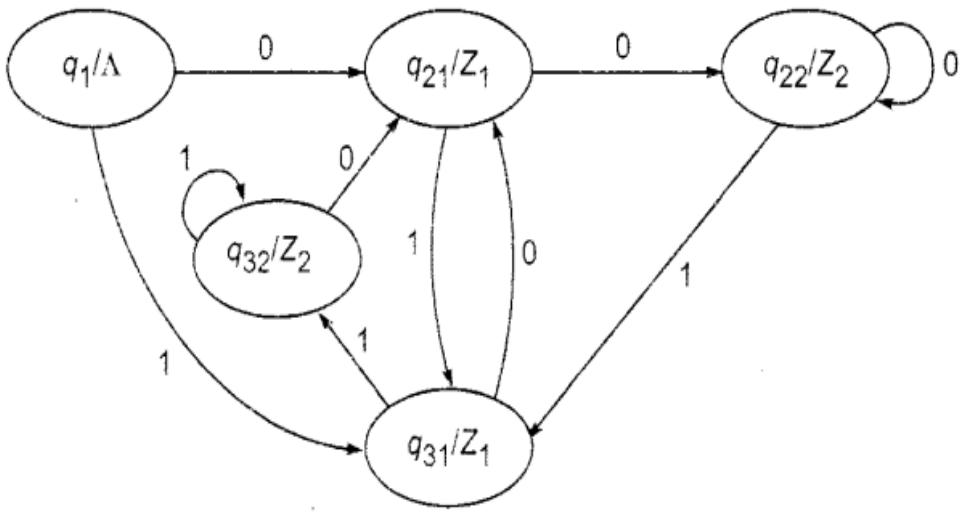
Present State	a=0		a=1	
	state	Output	State	Output
$\rightarrow q_1$	q_2	Z_1	q_3	Z_1
q_2	q_2	Z_2	q_3	Z_1
q_3	q_2	Z_1	q_3	Z_2



Present State	a=0		a=1	
	state	Output	State	Output
q_1	q_{21}	Z_1	q_{31}	Z_1
q_{21}	q_{22}	Z_2	q_{31}	Z_1
q_{22}	q_{22}	Z_2	q_{31}	Z_1
q_{31}	q_{21}	Z_1	q_{32}	Z_2
q_{32}	q_{21}	Z_1	q_{32}	Z_2

Procedure of transforming Mealy machine into Moore machine

Present State	Next State a=0	Next State a=1	Output
q_1	q_{21}	q_{31}	
q_{21}	q_{22}	q_{31}	Z_1
q_{22}	q_{22}	q_{31}	Z_2
q_{31}	q_{21}	q_{32}	Z_1
q_{32}	q_{21}	q_{32}	Z_2



Procedure of transforming Moore machine into Mealy machine

- Consider the moore machine described by the transition table given:

Present State	Next State a=0	Next State a=1	Output
$\rightarrow q_1$	q_1	q_2	0
q_2	q_1	q_3	0
q_3	q_1	q_3	1

Construct the corresponding mealy machine.

Procedure of transforming Moore machine into Mealy machine

- Consider the moore machine described by the transition table given:

Present State	Next State a=0	Next State a=1	Output
$\rightarrow q_1$	q_1	q_2	0
q_2	q_1	q_3	0
q_3	q_1	q_3	1

Construct the corresponding mealy machine.

- **Solution:**

Present State	a=0		a=1	
	state	Output	State	Output
$\rightarrow q_1$	q_1	0	q_2	0
q_2	q_1	0	q_3	1
q_3	q_1	0	q_3	1

Procedure of transforming Moore machine into Mealy machine

- Consider the moore machine described by the transition table given:

Present State	Next State a=0	Next State a=1	Output
$\rightarrow q_1$	q_1	q_2	0
q_2	q_1	q_3	0
q_3	q_1	q_3	1

Construct the corresponding mealy machine.

- **Solution:**

Present State	a=0		a=1	
	state	Output	State	Output
$\rightarrow q_1$	q_1	0	q_2	0
q_2	q_1	0	q_3	1
q_3	q_1	0	q_3	1

Now, Find identical rows and remove one of them

Procedure of transforming Moore machine into Mealy machine

- Consider the moore machine described by the transition table given:

Present State	Next State a=0	Next State a=1	Output
$\rightarrow q_1$	q_1	q_2	0
q_2	q_1	q_3	0
q_3	q_1	q_3	1

Construct the corresponding mealy machine.

- **Solution:**

Present State	a=0 state	a=0 Output	a=1 State	a=1 Output
$\rightarrow q_1$	q_1	0	q_2	0
q_2	q_1	0	q_3	1
q_3	q_1	0	q_3	1

Now, Find identical rows and remove one of them

Present State	a=0 state	a=0 Output	a=1 State	a=1 Output
$\rightarrow q_1$	q_1	0	q_2	0
q_2	q_1	0	q_2	1

Minimization of Finite Automata

- Equivalence: Two states q_1 and q_2 are equivalent(denoted by $q_1 \equiv q_2$), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states or both of them are non-final states for all x $\in \Sigma^*$.

Minimization of Finite Automata

- Equivalence: Two states q_1 and q_2 are equivalent(denoted by $q_1 \equiv q_2$), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states or both of them are non-final states for all x $\in \Sigma^*$.
- Precisely, Two states q_1 and q_2 are k-equivalent (k ≥ 0), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states *or* both non-final states for all string x of length k or less.

Minimization of Finite Automata

- Equivalence: Two states q_1 and q_2 are equivalent(denoted by $q_1 \equiv q_2$), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states or both of them are non-final states for all x $\in \Sigma^*$.
- Precisely, Two states q_1 and q_2 are k-equivalent (k ≥ 0), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states *or* both non-final states for all string x of length k or less.
- If $\delta(q_1, w)$ and $\delta(q_2, w)$ are equivalent then
  - for |w |= 0, the states are 0-equivalent.
  - for |w |= 1, the states are 1-equivalent.
  - for |w |= 2, the states are 2-equivalent.

    .

    ..
  - for |w |= n, the states are n-equivalent.

Minimization of Finite Automata

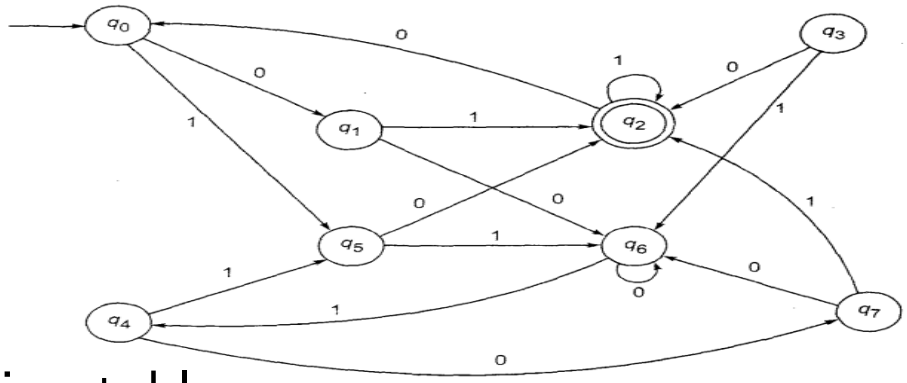
- Equivalence: Two states q_1 and q_2 are equivalent(denoted by $q_1 \equiv q_2$), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states or both of them are non-final states for all x $\in \Sigma^*$.
- Precisely, Two states q_1 and q_2 are k-equivalent (k ≥ 0), if both $\delta(q_1, x)$ and $\delta(q_2, x)$ are final states *or* both non-final states for all string x of length k or less.
- If $\delta(q_1, w)$ and $\delta(q_2, w)$ are equivalent then
  - for |w |= 0, the states are 0-equivalent.
  - for |w |= 1, the states are 1-equivalent.
  - for |w |= 2, the states are 2-equivalent.

    .

    ..
  - for |w |= n, the states are n-equivalent.
- Properties of Equivalence relations:
  - If a relation is equivalence or k-equivalence, then they are reflexive, symmetric and transitive.
  - If q_1 and q_2 are k-equivalent for all k ≥ 0, then they are equivalent.
  - If q_1 and q_2 are (k+1)-equivalent, then they are k-equivalent.
  - $\pi_n = \pi_{n+1}$ for some n

Minimization of Finite Automata

- Construct a minimum state automaton equivalent to the given finite automaton



Solution:

1. Draw transition table

State $\backslash \Sigma$	0	1
$\rightarrow q_0$	q_1	q_5
q_1	q_6	q_2
(q_2)	q_0	q_2
q_3	q_2	q_6
q_4	q_7	q_5
q_5	q_2	q_6
q_6	q_6	q_4
q_7	q_6	q_2

Minimization of Finite Automata

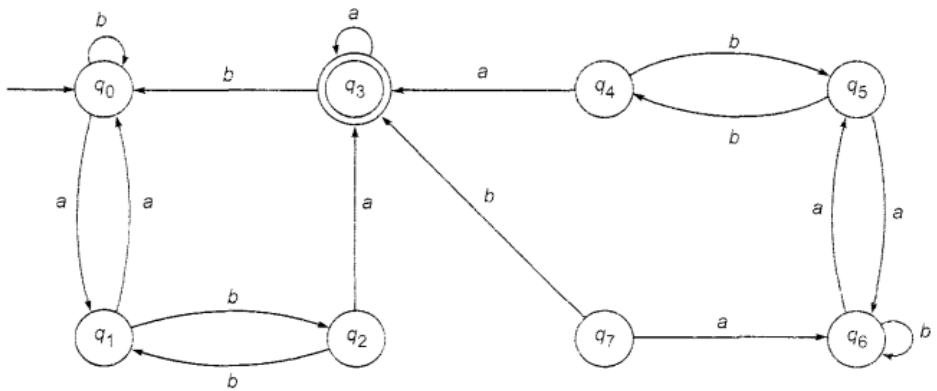
2. Find 0-equivalent set
   $\pi_0 = [q_0, q_1, q_3, q_4, q_5, q_6, q_7][q_2]$
3. Find 1-equivalent set
   $\pi_1 = [q_0, q_6, q_4][q_1, q_7][q_5, q_3][q_2]$
4. find 2-equivalent set
   $\pi_2 = [q_0, q_4][q_6][q_2][q_1, q_7][q_3, q_5]$
5. find 3-equivalent set
   $\pi_3 = [q_0, q_4][q_6][q_2][q_1, q_7][q_3, q_5]$

Therefore, M'=(Q',{ 0,1 } ,δ,q_0',F')
where Q'=$\{[q_2],[q_0, q_4],[q_6],[q_1, q_7],[q_3, q_5]\}$
$q_0' = [q_0, q_4]$, F'=$[q_2]$

State \ Σ	0	1
$[q_0, q_4]$	$[q_1, q_7]$	$[q_3, q_5]$
$[q_1, q_7]$	$[q_6]$	$[q_2]$
$[q_2]$	$[q_0, q_4]$	$[q_2]$
$[q_3, q_5]$	$[q_2]$	$[q_6]$
$[q_6]$	$[q_6]$	$[q_0, q_4]$

Minimization of Finite Automata

- Construct a minimum state automaton equivalent to the given finite automaton



- **Solution:**

State $\backslash \Sigma$	a	b
$\rightarrow q_0$	q_1	q_0
q_1	q_0	q_2
q_2	q_3	q_1
(q_3)	q_3	q_0
q_4	q_3	q_5
q_5	q_6	q_4
q_6	q_5	q_6
q_7	q_6	q_3

Minimization of Finite Automata

- π_0=$\{\{q_3\}\{q_0, q_1, q_2, q_4, q_5, q_6, q_7\}\}$
- π_1=$\{\{q_3\}\{q_0, q_1, q_5, q_6\}\{q_2, q_4\},\{q_7\}\}$
- π_2=$\{\{q_3\}\{q_0, q_6\}\{q_1, q_5\}\{q_2, q_4\}\{q_7\}\}$
- π_3=$\{\{q_3\}\{q_0, q_6\}\{q_1, q_5\}\{q_2, q_4\}\{q_7\}\}$

$\therefore$ Q'= $\{\{q_3\}\{q_0, q_6\}\{q_1, q_5\}\{q_2, q_4\}\{q_7\}\}$

q_0'=$\{q_0, q_6\}$

F'=$\{q_3\}$

Now, make δ' and transition diagram.

Minimization of Finite Automata

- Construct minimum state automaton equivalent to given automata M:

State $\backslash \Sigma$	a	b
$\rightarrow q_0$	q_0	q_3
q_1	q_2	q_5
q_2	q_3	q_4
q_3	q_0	q_5
q_4	q_0	q_6
q_5	q_1	q_4
(q_6)	q_1	q_3

Language: Introduction

- Language

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String A concatenation of finite symbols from the alphabet is called a string.

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String A concatenation of finite symbols from the alphabet is called a string.
    Example: If Σ=$\{$a,b$\}$ then a, abab, aaabb, abababababaaaaaaabaab, etc.

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String A concatenation of finite symbols from the alphabet is called a string.
    Example: If Σ={a,b} then a, abab, aaabb, abababababaaaaaaabaab, etc.
  - Empty String or Null String

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String A concatenation of finite symbols from the alphabet is called a string.
    Example: If Σ={a,b} then a, abab, aaabb, abababababaaaaaaabaab, etc.
  - Empty String or Null String$\wedge$ or Λ or ϕ
    $\Longrightarrow$ A string with no symbols
- Words

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String A concatenation of finite symbols from the alphabet is called a string.
    Example: If Σ={a,b} then a, abab, aaabb, abababababaaaaaaaabaab, etc.
  - Empty String or Null String$\wedge$ or Λ or ϕ
    $\Longrightarrow$ A string with no symbols
- Words $\Longrightarrow$ strings belonging to some language
  Example: If Σ={x} then a language L can be defined as
  L={x^n:n=1,2,3,....} or
  L={x,xx,xxx,....}
  Here, x, xx, xxx ,.... are the words of L.

Language: Introduction

- Language
  - Formal (Syntactic Languages)
  - Informal (Semantic Languages)
- Alphabet
  - String A concatenation of finite symbols from the alphabet is called a string.
    Example: If Σ={a,b} then a, abab, aaabb, abababababaaaaaaabaab, etc.
  - Empty String or Null String$\wedge$ or Λ or ϕ
    $\Longrightarrow$ A string with no symbols
- Words $\Longrightarrow$ strings belonging to some language
  Example: If Σ={x} then a language L can be defined as
  L={x^n:n=1,2,3,....} or
  L={x,xx,xxx,....}
  Here, x, xx, xxx ,.... are the words of L.
- All words are strings but not all strings are words.

Language: Introduction

- Length of String: $|S|$ $\Longrightarrow$ number of letters in the string.
  Example: Σ={a,b}
  If S= ababa, then $|S|$=5
- Reverse of String: S^r $\Longrightarrow$ Obtained by writing letters of 'S' in reverse order.

Language: Introduction

- Length of String: $|S| \implies$ number of letters in the string.
  Example: Σ={a,b}
  If S= ababa, then $|S|$=5
- Reverse of String: $S^r \implies$ Obtained by writing letters of 'S' in reverse order.
  Example:
  - If s=abc over Σ={a,b,c}

Language: Introduction

- Length of String: $|S| \implies$ number of letters in the string.
  Example: Σ={a,b}
  If S= ababa, then $|S|$=5
- Reverse of String: $S^r \implies$ Obtained by writing letters of 'S' in reverse order.
  Example:
  - If s=abc over Σ={a,b,c}
    Then, Rev(s) or s^r=cba

Language: Introduction

- Length of String: $|S| \implies$ number of letters in the string.
  Example: Σ={a,b}
  If S= ababa, then $|S|$=5
- Reverse of String: $S^r \implies$ Obtained by writing letters of 'S' in reverse order.
  Example:
  - If s=abc over Σ={a,b,c}
    Then, Rev(s) or s^r=cba
  - Σ={B,aB,bab,d}
    s=BaBbabBd

Language: Introduction

- Length of String: $|S|$ $\Longrightarrow$ number of letters in the string.
  Example: Σ={a,b}
  If S= ababa, then $|S|$=5
- Reverse of String: S^r $\Longrightarrow$ Obtained by writing letters of 'S' in reverse order.
  Example:
  - If s=abc over Σ={a,b,c}
    Then, Rev(s) or s^r=cba
  - Σ={B,aB,bab,d}
    s=BaBbabBd
    s^r=dBbabaBB

Language: Definition

- Descriptive Definition
- Recursive Definition
- Using Regular Expression
- Using Finite Automata, etc.

Language: Definition

- Descriptive Definition
- Recursive Definition
- Using Regular Expression
- Using Finite Automata, etc.

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.

Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.

Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}
4. The language L of strings ending in 0 over Σ={0,1}
   L={0,00,10,000,010,100,...}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}
4. The language L of strings ending in 0 over Σ={0,1}
   L={0,00,10,000,010,100,...}
5. The language L of Strings with number of "a"(s) equal to number of "b"(s) over Σ={a,b}
   L={$\wedge$, ab,aabb,abab,baba,abba,....}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}
4. The language L of strings ending in 0 over Σ={0,1}
   L={0,00,10,000,010,100,...}
5. The language L of Strings with number of "a"(s) equal to number of "b"(s) over Σ={a,b}
   L={$\wedge$, ab,aabb,abab,baba,abba,....}
6. Language **Even-Even** of string with even number of a(s) and even number of b(s) over Σ={a,b}
   L={$\wedge$, aa,bb,aaaa,aabb,abab,...}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}
4. The language L of strings ending in 0 over Σ={0,1}
   L={0,00,10,000,010,100,...}
5. The language L of Strings with number of "a"(s) equal to number of "b"(s) over Σ={a,b}
   L={$\wedge$, ab,aabb,abab,baba,abba,....}
6. Language **Even-Even** of string with even number of a(s) and even number of b(s) over Σ={a,b}
   L={$\wedge$, aa,bb,aaaa,aabb,abab,...}
7. Language Integer of strings over Σ=-,0,1,2,3,4,5,6,7,8,9
   L={....., -2,-1,0,1,2,....}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}
4. The language L of strings ending in 0 over Σ={0,1}
   L={0,00,10,000,010,100,...}
5. The language L of Strings with number of "a"(s) equal to number of "b"(s) over Σ={a,b}
   L={$\wedge$, ab,aabb,abab,baba,abba,....}
6. Language **Even-Even** of string with even number of a(s) and even number of b(s) over Σ={a,b}
   L={$\wedge$, aa,bb,aaaa,aabb,abab,...}
7. Language Integer of strings over Σ=-,0,1,2,3,4,5,6,7,8,9
   L={....., -2,-1,0,1,2,....}
8. Language $\{a^n b^n\}$ over Σ={a,b} or $\{a^n b^n$:n=1,2,3,....}
   L={ab,aabb,aaabbb, ...}

Descriptive definition of Language

The language is defined describing the conditions imposed on its words.
Example:

1. The language L of strings of odd length, defined over Σ={a}
   $\Longrightarrow$ L={a, aaa,aaaaa, ...}
2. The language L of strings that doesnot start with 'a' defined over Σ={a,b,c}
   $\Longrightarrow$ L={b,c,ba, bb,bc,ca,cb,cc}
3. The language L of strings of length 2 over Σ={0,1,2}
   $\Longrightarrow$ L={00,01,02,10,22,12,...}
4. The language L of strings ending in 0 over Σ={0,1}
   L={0,00,10,000,010,100,...}
5. The language L of Strings with number of "a"(s) equal to number of "b"(s) over Σ={a,b}
   L={$\wedge$, ab,aabb,abab,baba,abba,....}
6. Language **Even-Even** of string with even number of a(s) and even number of b(s) over Σ={a,b}
   L={$\wedge$, aa,bb,aaaa,aabb,abab,...}
7. Language Integer of strings over Σ=-,0,1,2,3,4,5,6,7,8,9
   L={....., -2,-1,0,1,2,....}
8. Language $\{a^n b^n\}$ over Σ={a,b} or $\{a^n b^n$:n=1,2,3,....}
   L={ab,aabb,aaabbb, ...}
9. Palindrome over Σ={a,b}
   L={$\wedge$, a,b, aa,bb, aaa,aba,bab,bbb,....}

GRAMMAR

- A grammar is (V_N, Σ, P, S)
  where V_N is a non-empty set whose elements are called variables.
  Σ finite non-empty set whose elements are called terminals.
  S is aspecial symbol called Start Symbol.
  P are set of Production rules.
- $V_N \cap \Sigma = \phi$

GRAMMAR

- A grammar is (V_N, Σ, P, S)
  where V_N is a non-empty set whose elements are called variables.
  Σ finite non-empty set whose elements are called terminals.
  S is aspecial symbol called Start Symbol.
  P are set of Production rules.
- $V_N \cap \Sigma = \phi$

Example

G=$\{V_N, \Sigma, P, S\}$ is a Grammar, where
V_N=$\{$<sentence>, <noun>, <verb>, <adverb>$\}$, Σ=$\{$Ram,Sam,ran,sang,fast$\}$,
S=<sentence>
P consists of following productions:
<sentence>$\rightarrow$<noun><verb>
<sentence>$\rightarrow$<noun><verb><adverb>
<noun>$\rightarrow$ Ram
<noun>$\rightarrow$ Sam
<verb>$\rightarrow$ ran
<verb>$\rightarrow$ sang
<adverb>$\rightarrow$ fast

Grammar

- If G=({S},{0,1},{S→ 0S1,S→ ∧},S). Find L(G).

Grammar

- If G=($\{S\}$,$\{0,1\}$,$\{S\rightarrow 0S1, S\rightarrow \wedge\}$,S). Find L(G).
  Solution: $S\rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111 \ldots . 0^nS1^n$
  $0^n \wedge 1^n = 0^n1^n \in L(G)$ for $n \geq 0$
  $\therefore L(G)=\{0^n1^n \mid n \geq 0\}$

Grammar

- If G=({S},{0,1},{S→ 0S1,S→ ∧},S). Find L(G).
  Solution: S→0S1 → 00S11 → 000S111.....0^nS1^n
  $0^n \wedge 1^n = 0^n1^n \in$ L(G) for n ≥ 0
  ∴ L(G)=$\{0^n1^n \mid n \geq 0\}$
- If G=({S},{a},{S → SS},S), Find the language generated by G.

Grammar

- If G=($\{S\}$,$\{0,1\}$,$\{S\rightarrow 0S1, S\rightarrow \wedge\}$,S). Find L(G).
  Solution: $S\rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111.....0^nS1^n$
  $0^n\wedge 1^n = 0^n1^n \in L(G)$ for $n \geq 0$
  $\therefore L(G)=\{0^n1^n \mid n \geq 0\}$
- If G=($\{S\}$,$\{a\}$,$\{S \rightarrow SS\}$,S), Find the language generated by G.
  Solution: $L(G)=\phi$

Grammar

- If G=($\{S\}$,$\{0,1\}$,$\{S\rightarrow 0S1, S\rightarrow \wedge\}$,S). Find L(G).
  Solution: $S\rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111.....0^nS1^n$
  $0^n\wedge 1^n = 0^n1^n \in L(G)$ for $n \geq 0$
  $\therefore L(G)=\{0^n1^n \mid n \geq 0\}$
- If G=($\{S\}$,$\{a\}$,$\{S \rightarrow SS\}$,S), Find the language generated by G.
  Solution: $L(G)=\phi$
- Let G=($\{S,C\}$,$\{a,b\}$,P,S), where P consists of $S \rightarrow aCa, C \rightarrow aCa \mid b$. Find L(G).

Grammar

- If G=($\{S\},\{0,1\},\{S\rightarrow 0S1, S\rightarrow \wedge\},S$). Find L(G).
  Solution: $S\rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111.....0^nS1^n$
  $0^n\wedge 1^n = 0^n1^n \in L(G)$ for $n \geq 0$
  $\therefore L(G)=\{0^n1^n \mid n \geq 0\}$
- If G=($\{S\},\{a\},\{S \rightarrow SS\},S$), Find the language generated by G.
  Solution: $L(G)=\phi$
- Let G=($\{S,C\},\{a,b\},P,S$), where P consists of $S \rightarrow aCa, C \rightarrow aCa \mid b$. Find L(G).
  Solution: $S \implies aCa \implies aba$. So, $aba \in L(G)$

Grammar

- If G=($\{S\},\{0,1\},\{S\rightarrow 0S1, S\rightarrow \wedge\}$,S). Find L(G).
  Solution: $S\rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111.....0^nS1^n$
  $0^n\wedge 1^n = 0^n1^n \in L(G)$ for $n \geq 0$
  $\therefore L(G)=\{0^n1^n \mid n \geq 0\}$
- If G=($\{S\},\{a\},\{S \rightarrow SS\}$,S), Find the language generated by G.
  Solution: $L(G)=\phi$
- Let G=($\{S,C\},\{a,b\}$,P,S), where P consists of $S \rightarrow aCa, C \rightarrow aCa \mid b$. Find L(G).
  Solution: $S \implies aCa \implies aba$. So, $aba \in L(G)$
  $S \implies aCa$ *(using $S\rightarrow aCa$)*
  $\implies a^nCa^n$ *(using $S\rightarrow aCa$ (n-1) times)*

Grammar

- If G=($\{S\},\{0,1\},\{S\rightarrow 0S1, S\rightarrow \wedge\}$,S). Find L(G).
  Solution: $S\rightarrow 0S1 \rightarrow 00S11 \rightarrow 000S111.....0^nS1^n$
  $0^n \wedge 1^n = 0^n1^n \in L(G)$ for $n \geq 0$
  $\therefore L(G)=\{0^n1^n \mid n \geq 0\}$
- If G=($\{S\},\{a\},\{S \rightarrow SS\}$,S), Find the language generated by G.
  Solution: $L(G)=\phi$
- Let G=($\{S,C\},\{a,b\}$,P,S), where P consists of $S \rightarrow aCa, C \rightarrow aCa \mid b$. Find L(G).
  Solution: $S \implies aCa \implies aba$. So, $aba \in L(G)$
  $S \implies aCa$ *(using $S\rightarrow aCa$)*
  $\implies a^nCa^n$ *(using $S\rightarrow aCa$ (n-1) times)*
  $\implies a^nba^n$ *(using $C \rightarrow b$)*
  Hence, $a^nba^n \in L(G)$, where $n \geq 1$
  $\therefore L(G)=\{a^nba^n \mid n \geq 1\}$

Grammar

Exercise

Construct a grammar G so that $L(G) = \{a^n b a^m \mid n, m \geq 1\}$

Grammar

Exercise

Construct a grammar G so that $L(G) = \{a^n b a^m \mid n, m \geq 1\}$

- If G is $S \rightarrow aS \mid bS \mid a \mid b$, Find L(G).

Grammar

Exercise

Construct a grammar G so that L(G) = $\{a^n b a^m \mid n, m \geq 1\}$

- If G is S $\rightarrow$ aS |bS |a |b, Find L(G).
  Solution: L(G)=$\{a, b\}^+$

Grammar

Exercise

Construct a grammar G so that L(G) = $\{a^n b a^m \mid n, m \geq 1\}$

- If G is S $\rightarrow$ aS |bS |a |b, Find L(G).
  Solution: L(G)=$\{a, b\}^+$

Exercise 1

If G is S $\rightarrow$ aS |a, then show that L(G)=$\{a\}^+$

Grammar

Exercise

Construct a grammar G so that L(G) = $\{a^n b a^m \mid n, m \geq 1\}$

- If G is S $\rightarrow$ aS |bS |a |b, Find L(G).
  Solution: L(G)=$\{a, b\}^+$

Exercise 1

If G is S $\rightarrow$ aS |a, then show that L(G)=$\{a\}^+$

- Let L be the set of all palindromes over {a,b}. Construct a grammar G generating L.

Grammar

Exercise

Construct a grammar G so that L(G) = $\{a^n b a^m \mid n, m \geq 1\}$

- If G is S $\rightarrow$ aS |bS |a |b, Find L(G).
  Solution: L(G)=$\{a, b\}^+$

Exercise 1

If G is S $\rightarrow$ aS |a, then show that L(G)=$\{a\}^+$

- Let L be the set of all palindromes over {a,b}. Construct a grammar G generating L.
  Solution: $\wedge$, a, b, or axa and bxb are palindromes.

Grammar

Exercise

Construct a grammar G so that L(G) = $\{a^n b a^m \mid n, m \geq 1\}$

- If G is S $\rightarrow$ aS |bS |a |b, Find L(G).
  Solution: L(G)=$\{a, b\}^+$

Exercise 1

If G is S $\rightarrow$ aS |a, then show that L(G)=$\{a\}^+$

- Let L be the set of all palindromes over {a,b}. Construct a grammar G generating L.
  Solution: $\wedge$, a, b, or axa and bxb are palindromes.
  $\therefore$ P consists of
  S $\rightarrow \wedge$
  S $\rightarrow$ a, S $\rightarrow$ b
  S $\rightarrow$ aSa, S $\rightarrow$ bSb
  Thus, G=({S},{a,b},P,S)

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a, b\}^*\}$

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a, b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a, b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a, b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a, b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$
  L_2=$\{a^n b^n c^i \mid n \geq 1, i \geq 1\}$

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a,b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$
  L_2=$\{a^n b^n c^i \mid n \geq 1, i \geq 1\}$
  Let "P" be as follows:
  S $\rightarrow$ A
  A $\rightarrow$ ab |aAb

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a, b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$
  L_2=$\{a^n b^n c^i \mid n \geq 1, i \geq 1\}$
  Let "P" be as follows:
  S $\rightarrow$ A
  A $\rightarrow$ ab |aAb
  S $\rightarrow$ Sc

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a,b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$
  L_2=$\{a^n b^n c^i \mid n \geq 1, i \geq 1\}$
  Let "P" be as follows:
  S $\rightarrow$ A
  A $\rightarrow$ ab |aAb
  S $\rightarrow$ Sc
  Let G=({S,A}, {a,b,c}, P,S) for n $\geq$ 1, i $\geq$ 0

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a,b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$
  L_2=$\{a^n b^n c^i \mid n \geq 1, i \geq 1\}$
  Let "P" be as follows:
  S $\rightarrow$ A
  A $\rightarrow$ ab |aAb
  S $\rightarrow$ Sc
  Let G=({S,A}, {a,b,c}, P,S) for n $\geq$ 1, i $\geq$ 0
  $S \overset{*}{\rightarrow} Sc^i \rightarrow Ac^i \rightarrow a^{n-1}Ab^{n-1}c^i \rightarrow a^{n-1}abb^{n-1}c^i = a^n b^n c^i$

Grammar

- Construct a Grammar generating L=$\{wcw^T \mid w \in \{a,b\}^*\}$
  Solution: Let G=({S},{a,b,c},P,S)
  where P is defined as
  S $\rightarrow$ aSa |bSb |c
- Find a grammar generating L=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$
  Solution: L=$L_1 \cup L_2$, L_1=$\{a^n b^n \mid n \geq 1\}$
  L_2=$\{a^n b^n c^i \mid n \geq 1, i \geq 1\}$
  Let "P" be as follows:
  S $\rightarrow$ A
  A $\rightarrow$ ab |aAb
  S $\rightarrow$ Sc
  Let G=({S,A}, {a,b,c}, P,S) for n $\geq$ 1, i $\geq$ 0
  $S \stackrel{*}{\rightarrow} Sc^i \rightarrow Ac^i \rightarrow a^{n-1}Ab^{n-1}c^i \rightarrow a^{n-1}abb^{n-1}c^i = a^n b^n c^i$
  L(G)=$\{a^n b^n c^i \mid n \geq 1, i \geq 0\}$

Grammar

- Find a grammar generating $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  $S \rightarrow aS \mid A$
  $A \rightarrow bAc \mid bc$

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  $S \rightarrow aS \mid A$
  $A \rightarrow bAc \mid bc$
- Let G =({S,A},{0,1,2},P,S) where P consists of
  $S \rightarrow 0SA2 \mid S \rightarrow 012$
  $2A \rightarrow A2$
  $1A \rightarrow 11$. Show that L(G)=$\{0^n 1^n 2^n \mid n \geq 1\}$

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  S $\rightarrow$ aS |A
  A $\rightarrow$ bAc |bc
- Let G =({S,A},{0,1,2},P,S) where P consists of
  S $\rightarrow$ 0SA2 |S $\rightarrow$ 012
  2A $\rightarrow$ A2
  1A $\rightarrow$ 11. Show that L(G)=$\{0^n 1^n 2^n \mid n \geq 1\}$
  Solution: S $\stackrel{*}{\rightarrow}$ $0^{n-1} S (A2)^{n-1}$ by applying S $\rightarrow$ 0SA2 (n-1) times

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  $S \rightarrow aS \mid A$
  $A \rightarrow bAc \mid bc$
- Let G =({S,A},{0,1,2},P,S) where P consists of
  $S \rightarrow 0SA2 \mid S \rightarrow 012$
  $2A \rightarrow A2$
  $1A \rightarrow 11$. Show that $L(G)=\{0^n 1^n 2^n \mid n \geq 1\}$
  Solution: $S \stackrel{*}{\rightarrow} 0^{n-1} S (A2)^{n-1}$ by applying S $\rightarrow$ 0SA2 (n-1) times
  $\rightarrow 0^n 12 (A2)^{n-1}$ by applying S $\rightarrow$ 012

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  $S \rightarrow aS \mid A$
  $A \rightarrow bAc \mid bc$
- Let G =({S,A},{0,1,2},P,S) where P consists of
  $S \rightarrow 0SA2 \mid S \rightarrow 012$
  $2A \rightarrow A2$
  $1A \rightarrow 11$. Show that L(G)=$\{0^n 1^n 2^n \mid n \geq 1\}$
  Solution: $S \stackrel{*}{\rightarrow} 0^{n-1} S (A2)^{n-1}$ by applying S $\rightarrow$ 0SA2 (n-1) times
  $\rightarrow 0^n 12 (A2)^{n-1}$ by applying S $\rightarrow$ 012
  $\stackrel{*}{\rightarrow} 0^n 1 A^{n-1} 2^n$ by applying 2A $\rightarrow$ A2 several times

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  $S \rightarrow aS \mid A$
  $A \rightarrow bAc \mid bc$
- Let G =({S,A},{0,1,2},P,S) where P consists of
  $S \rightarrow 0SA2 \mid S \rightarrow 012$
  $2A \rightarrow A2$
  $1A \rightarrow 11$. Show that L(G)=$\{0^n1^n2^n \mid n \geq 1\}$
  Solution: $S \stackrel{*}{\rightarrow} 0^{n-1}S(A2)^{n-1}$ by applying S $\rightarrow$ 0SA2 (n-1) times
  $\rightarrow 0^n12(A2)^{n-1}$ by applying S $\rightarrow$ 012
  $\stackrel{*}{\rightarrow} 0^n1A^{n-1}2^n$ by applying 2A $\rightarrow$ A2 several times
  $\stackrel{*}{\rightarrow} 0^n1^n2^n$ by applying 1A $\rightarrow$ 11 (n-1 times)

Grammar

- Find a grammar generating
  $\{a^j b^n c^n \mid n \geq 1, j \geq 0\}$
  Solution: Let G=({S,A},{a,b,c},P,S)
  where "P" consists of:
  $S \rightarrow aS \mid A$
  $A \rightarrow bAc \mid bc$
- Let G =({S,A},{0,1,2},P,S) where P consists of
  $S \rightarrow 0SA2 \mid S \rightarrow 012$
  $2A \rightarrow A2$
  $1A \rightarrow 11$. Show that $L(G)=\{0^n1^n2^n \mid n \geq 1\}$
  Solution: $S \stackrel{*}{\rightarrow} 0^{n-1}S(A2)^{n-1}$ by applying $S \rightarrow 0SA2$ (n-1) times
  $\rightarrow 0^n12(A2)^{n-1}$ by applying $S \rightarrow 012$
  $\stackrel{*}{\rightarrow} 0^n1A^{n-1}2^n$ by applying $2A \rightarrow A2$ several times
  $\stackrel{*}{\rightarrow} 0^n1^n2^n$ by applying $1A \rightarrow 11$ (n-1 times)
  $\therefore 0^n1^n2^n \in L(G)$ for all $n \geq 1$

Grammar

- Construct grammar G generating $\{a^n b^n c^n \mid n \geq 1\}$

Grammar

- Construct grammar G generating $\{a^n b^n c^n \mid n \geq 1\}$
  Solution: G=({S,B,C},{a,b,c},P,S)
  where P consists of: S → aSBC |aBC, CB → BC, aB → ab, bB → bb, bC → bc, cC → cc
  S ⟹ aBC ⟹ abC ⟹ abc

Grammar

- Construct grammar G generating $\{a^n b^n c^n \mid n \geq 1\}$
  Solution: G=({S,B,C},{a,b,c},P,S)
  where P consists of: S → aSBC |aBC, CB → BC, aB → ab, bB → bb, bC → bc, cC → cc
  S $\Longrightarrow$ aBC $\Longrightarrow$ abC $\Longrightarrow$ abc
- Construct a grammar G generating $\{xx \mid x \in \{a, b\}^*\}$

Grammar

- Construct grammar G generating $\{a^n b^n c^n \mid n \geq 1\}$
  Solution: G=({S,B,C},{a,b,c},P,S)
  where P consists of: S → aSBC |aBC, CB → BC, aB → ab, bB → bb, bC → bc, cC → cc
  S ⟹ aBC ⟹ abC ⟹ abc
- Construct a grammar G generating $\{xx \mid x \in \{a, b\}^*\}$
  Solution: Let G is as follows: G=({S,D,E,F,A,B},{a,b},P,S)
  where P consists of :
  S → DEF
  DE → aDA, DE → bDB
  AF → EaF, BF → EbF
  Aa → aA, Ab → bA, Ba → aB, Bb→bB
  aE → Ea, bE→ Eb
  DE → ∧, F → ∧

Grammar

- Let G=({S,A,B},{a,b},P,S) where P consists of
  $S \rightarrow aABa$, $A \rightarrow baABb$, $B \rightarrow Aab$, $aA \rightarrow baa$, $bBb \rightarrow abab$.
  Test whether w = baabbabaaabbaba is in L(G).

Grammar

- Let G=({S,A,B},{a,b},P,S) where P consists of
  S → aABa, A → baABb, B → Aab, aA → baa, bBb → abab.
  Test whether w = baabbabaaabbaba is in L(G).
  Solution: S→ $\underline{aA}$Ba
  $\Longrightarrow$ baa$\underline{B}$a
  $\Longrightarrow$ baa$\underline{A}$aba
  $\Longrightarrow$ baab$\underline{aA}$Bbaba
  $\Longrightarrow$ baabbaa$\underline{B}$baba
  $\Longrightarrow$ baabbaa$\underline{aA}$abbaba
  $\Longrightarrow$ baabbabaaabbaba=w
  $\therefore$ w $\in$ L(G)

Grammar

- Let G=({S,A,B},{a,b},P,S) where P consists of
  S → aABa, A → baABb, B → Aab, aA → baa, bBb → abab.
  Test whether w = baabbabaaabbaba is in L(G).
  Solution: S→ $\underline{aA}$Ba
  $\Longrightarrow$ baa$\underline{B}$a
  $\Longrightarrow$ baa$\underline{A}$aba
  $\Longrightarrow$ baab$\underline{aA}$Bbaba
  $\Longrightarrow$ baabbaa$\underline{B}$baba
  $\Longrightarrow$ baabbaa$\underline{aA}$abbaba
  $\Longrightarrow$ baabbabaaabbaba=w
  $\therefore$ w $\in$ L(G)
- If the grammar G is given by the productions S → aSa |bSb |aa |bb |∧, show that:
  - L(G) has no strings of odd length
  - Any string in L(G) is of length 2n, n $\geq$ 0
  - The number of strings of length 2n is 2^n

Chomsky Classification of Languages

- Chomsky classified grammar into 4 types i.e. (type 0-3)
- A type 0 grammar is any phrase structure grammar without any restrictions

  $\Longrightarrow$ All grammar we have considered till now are type 0 grammar
- In a production of form $\phi A \psi \rightarrow \phi \alpha \psi$

  Example:
  - **ab**A**bcd** $\rightarrow$ **ab**AB**bcd**

    $\phi \Longrightarrow ab$

    $\alpha \Longrightarrow AB$

    $\psi \Longrightarrow bcd$
  - **A**C $\rightarrow$ A

    $\phi \Longrightarrow A$

    $\alpha \Longrightarrow \wedge$

    $\psi \Longrightarrow \wedge$

Chomsky Classification of Languages

- $C \rightarrow \wedge$

  $\phi \implies \wedge$

  $\alpha \implies \wedge$

  $\psi \implies \wedge$

Chomsky Classification of Languages

- A production of the form $\phi A\ \psi \rightarrow \phi\alpha\psi$ is called Type-1 production or Context-sensitive Language, if $\alpha \neq \wedge$

  $\Longrightarrow$ In type-1 production erasing 'A' is not allowed

Chomsky Classification of Languages

- A production of the form $\phi A\ \psi \rightarrow \phi\alpha\psi$ is called Type-1 production or Context-sensitive Language, if $\alpha \neq \wedge$

  $\Longrightarrow$ In type-1 production erasing 'A' is not allowed

  Example:
  - aAbCD $\rightarrow$ abcDbcD is a type 1 production.
    A is replaced by bcD $\neq \wedge$
  - AB $\rightarrow$ AbBc is a type 1 production.
  - A $\rightarrow$ abA is a type 1 production.
- A grammar is called type 1 or Context sensitive or Context-dependent if all its productions are type 1 productions.

Chomsky Classification of Languages

- A production of the form $\phi A\ \psi \rightarrow \phi\alpha\psi$ is called Type-1 production or Context-sensitive Language, if $\alpha \neq \wedge$

  $\Longrightarrow$ In type-1 production erasing 'A' is not allowed

  Example:
  - aAbCD $\rightarrow$ abcDbcD is a type 1 production.
    A is replaced by bcD $\neq \wedge$
  - AB $\rightarrow$ AbBc is a type 1 production.
  - A $\rightarrow$ abA is a type 1 production.
- A grammar is called type 1 or Context sensitive or Context-dependent if all its productions are type 1 productions.
- The production S $\rightarrow \wedge$ is also allowed in type 1 grammar, but in this case S does not appear on the right-hand side of any production.

Chomsky Classification of Languages

- A production of the form $\phi A\ \psi \rightarrow \phi\alpha\psi$ is called Type-1 production or Context-sensitive Language, if $\alpha \neq \wedge$
  $\Longrightarrow$ In type-1 production erasing 'A' is not allowed
  Example:
  - aAbCD $\rightarrow$ abcDbcD is a type 1 production.
    A is replaced by bcD $\neq \wedge$
  - AB $\rightarrow$ AbBc is a type 1 production.
  - A $\rightarrow$ abA is a type 1 production.
- A grammar is called type 1 or Context sensitive or Context-dependent if all its productions are type 1 productions.
- The production S $\rightarrow \wedge$ is also allowed in type 1 grammar, but in this case S does not appear on the right-hand side of any production.
- The language generated by a type-1 grammar is called a type-1 or context-sensitive language

Chomsky Classification of Languages

- A grammar G= (V_N, Σ,P,S) is monotonic (or length-increasing) if every production in P is of the form $\alpha \rightarrow \beta$ with $|\alpha| \leq |\beta|$ or $S \rightarrow \wedge$. In second case, S does not appear on right-hand side of any production in P.

Chomsky Classification of Languages

- A grammar G= (V_N, Σ,P,S) is monotonic (or length-increasing) if every production in P is of the form $\alpha \rightarrow \beta$ with $|\alpha| \leq |\beta|$ or S $\rightarrow \wedge$. In second case, S does not appear on right-hand side of any production in P.
- Type-2: Context free Grammar generates context free language
- A Type-2 production is a production of the form A $\rightarrow \alpha$ where A $\in V_N$ and $\alpha \in (V_N \vee \Sigma)^*$

Chomsky Classification of Languages

- A grammar G= (V_N, Σ,P,S) is monotonic (or length-increasing) if every production in P is of the form $\alpha \rightarrow \beta$ with $|\alpha| \leq |\beta|$ or S $\rightarrow \wedge$. In second case, S does not appear on right-hand side of any production in P.
- Type-2: Context free Grammar generates context free language
- A Type-2 production is a production of the form A $\rightarrow \alpha$ where A $\in V_N$ and $\alpha \in (V_N \vee \Sigma)^*$
- In other words, in Type-2 $\Longrightarrow$
  - It should be in Type-1
  - L.H.S. production should have only 1 variable i.e. $|A| = 1$ and there is no restriction on α
    Example: S $\rightarrow$ Aa, A $\rightarrow$a, B $\rightarrow$ abc, A $\rightarrow \wedge$ are type-2 productions.

Chomsky Classification of Languages

- A production of the form A $\rightarrow$ a or A $\rightarrow$ aB, where A,B $\in V_N$ and a$\in \Sigma$ is called a type-3 production.
- A grammar is called a type-3 or **Regular Grammar** if all its productions are type-3 productions.
- A production S $\rightarrow \wedge$ is allowed in type-3 grammar, but in this case S does not appear on the right-hand side of any production

Chomsky Classification of Languages

1. Find the highest type number which can be applied to the following productions:
   - S $\rightarrow$ Aa, A $\rightarrow$ c |Ba, B $\rightarrow$ abc
   - S $\rightarrow$ ASB |d, A $\rightarrow$ aA
   - S $\rightarrow$ aS |ab
2. Differentiate between Recursive Set and Recursively Enumerable Set
3. Prove that Context-sensitive language is recursive.
4. Prove that there exists a recursive set which is not a contxt-sensitive language over {0,1}.
5. Let G=({A,B,S},{0,1},P.S) where P consists of S $\rightarrow$ 0AB, A0 $\rightarrow$ S0B, A1 $\rightarrow$SB1, B $\rightarrow$SA, B $\rightarrow$01. Show that L(G)= ϕ.
6. Find the language generated by grammar S $\rightarrow$ AB, A $\rightarrow$A1 |0, B $\rightarrow$ 2B |3. Can the above language be generated by a grammar of higher type?

Chomsky Classification of Languages

7. Construct a grammar which generates all even integer upto 998.
8. Construct CFG to generate the following:
   - $\{0^m 1^n \mid m \neq n,\ m,n \geq 1\}$
   - $\{a^l b^m c^n \mid$ one of l,m,n equals 1 and remaining tqo are equal$\}$
   - $\{a^l b^m c^n \mid l+m=n\}$
   - The set of all strings over $\{0,1\}$ containing twice as many 0's and 1's
9. Show that $G_1=(\{S\},\{a,b\},P_1,S)$ where $P_1=\{S \rightarrow aSb \mid ab\}$ is equivalent to $G_2=(\{S,A,B,C\},\{a,b\},P_2,S)$, where P_2 consists of $S \rightarrow AC$, $C \rightarrow SB$, $S \rightarrow AB$, $A \rightarrow a$, $B \rightarrow b$
10. What are the applications of different grammar types?

Regular Expression

- A language is regular if there exists a finite acceptor for it
  $\therefore$ Every regular language can be described as DFA or NDFA

Regular Expression

- A language is regular if there exists a finite acceptor for it
  $\therefore$ Every regular language can be described as DFA or NDFA
- Regular Expression: Algebraic description of languages

Regular Expression

- A language is regular if there exists a finite acceptor for it
  $\therefore$ Every regular language can be described as DFA or NDFA
- Regular Expression: Algebraic description of languages
- Let Σ be a given alphabet, then:
  1. $\phi, \wedge$ and $a \in \Sigma$ are all regular expressions, called **Primitive regular expressions**.
  2. If R_1 and R_2 are regular expressions, so are $R_1 + R_2, R_1.R_2, {R_1}^*$ and (R_1)
  3. A string is a regular expression if and only if it can be derived from primitive regular expressions by a finite number of applications of the rules in (2)

Regular Expression

- A language is regular if there exists a finite acceptor for it
  $\therefore$ Every regular language can be described as DFA or NDFA
- Regular Expression: Algebraic description of languages
- Let Σ be a given alphabet, then:
  1. $\phi, \wedge$ and a $\in \Sigma$ are all regular expressions, called **Primitive regular expressions**.
  2. If R_1 and R_2 are regular expressions, so are $R_1 + R_2, R_1.R_2, {R_1}^*$ and (R_1)
  3. A string is a regular expression if and only if it can be derived from primitive regular expressions by a finite number of applications of the rules in (2)
- **Example:** For Σ={a,b,c}, the string $(a + b.c)^*.(c + \phi)$ is a regular expression
  while $(a + b+)$ is not a regular expression.

Language Associated with Regular Expression

The language L(R) denoted by any regular expression 'R' is defined by following rules

1. ϕ is a R.E. denoting empty set
2. $\wedge$ is a R.E. denoting $\{\wedge\}$
3. For every $a \in \Sigma$, **a** is a R.E. denoting $\{\mathbf{a}\}$
   If R_1 and R_2 are R.E. , then
4. $L(R_1+R_2)=L(R_1) \vee L(R_2)$
5. $L(R_1.R_2)=L(R_1)L(R_2)$
6. $L((R_1))=L(R_1)$
7. $L(R_1^*)=(L(R_1))^*$

Language Associated with Regular Expression

The language L(R) denoted by any regular expression 'R' is defined by following rules

1. ϕ is a R.E. denoting empty set
2. $\wedge$ is a R.E. denoting $\{\wedge\}$
3. For every $a \in \Sigma$, **a** is a R.E. denoting $\{\mathbf{a}\}$
   If R_1 and R_2 are R.E. , then
4. $L(R_1+R_2)=L(R_1) \vee L(R_2)$
5. $L(R_1.R_2)=L(R_1)L(R_2)$
6. $L((R_1))=L(R_1)$
7. $L(R_1^*)=(L(R_1))^*$

Example: For Σ={a,b}, the expression R=$(a+b)^*$(a+bb) is regular

Language Associated with Regular Expression

The language L(R) denoted by any regular expression 'R' is defined by following rules

1. ϕ is a R.E. denoting empty set
2. $\wedge$ is a R.E. denoting $\{\wedge\}$
3. For every $a \in \Sigma$, **a** is a R.E. denoting $\{\mathbf{a}\}$
   If R_1 and R_2 are R.E. , then
4. $L(R_1+R_2)=L(R_1) \vee L(R_2)$
5. $L(R_1.R_2)=L(R_1)L(R_2)$
6. $L((R_1))=L(R_1)$
7. $L(R_1^*)=(L(R_1))^*$

Example: For Σ={a,b}, the expression R=$(a+b)^*$(a+bb) is regular

$\Longrightarrow$ L(R)={a,bb,aa,abb,ba,bbb,....}

Language Associated with Regular Expression

The language L(R) denoted by any regular expression 'R' is defined by following rules

1. ϕ is a R.E. denoting empty set
2. $\wedge$ is a R.E. denoting $\{\wedge\}$
3. For every $a \in \Sigma$, **a** is a R.E. denoting $\{\mathbf{a}\}$
   If R_1 and R_2 are R.E. , then
4. $L(R_1+R_2)=L(R_1) \vee L(R_2)$
5. $L(R_1.R_2)=L(R_1)L(R_2)$
6. $L((R_1))=L(R_1)$
7. $L(R_1^*)=(L(R_1))^*$

Example: For Σ={a,b}, the expression R=$(a+b)^*$(a+bb) is regular

$\Longrightarrow$ L(R)={a,bb,aa,abb,ba,bbb,....}

$\Longrightarrow$ L(R) is the set of all strings on {a,b}, terminated by either 'a' or 'bb'.

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1} \mid n \geq 0, m \geq 0\}$

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1} \mid n \geq 0, m \geq 0\}$
- For $\Sigma = \{0,1\}$. Give a regular expression 'R' such that
  L(R)=$\{w \in \Sigma^* \mid$ w has atleast one pair of consecutive zeroes$\}$

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1}$ |n$\geq$0, m$\geq$0$\}$
- For Σ =$\{$0,1$\}$. Give a regular expression 'R' such that
  L(R)=$\{$w$\in \Sigma^*$|w has atleast one pair of consecutive zeroes$\}$
  Solution: R=$(0+1)^*00(0+1)^*$

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1} \mid n \geq 0, m \geq 0\}$
- For $\Sigma = \{0,1\}$. Give a regular expression 'R' such that
  L(R)=$\{w \in \Sigma^* \mid$ w has atleast one pair of consecutive zeroes$\}$
  Solution: R=$(0+1)^*00(0+1)^*$
- Find R.E. for language
  L=$\{w \in \{0,1\}^* \mid$ w has no pair of consecutive zeroes$\}$

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1} \mid n\geq 0, m\geq 0\}$
- For $\Sigma = \{0,1\}$. Give a regular expression 'R' such that
  L(R)=$\{w\in \Sigma^* \mid$ w has atleast one pair of consecutive zeroes$\}$
  Solution: R=$(0+1)^*00(0+1)^*$
- Find R.E. for language
  L=$\{w \in \{0,1\}^* \mid$ w has no pair of consecutive zeroes$\}$
  Solution: R=$(1+01)^*(0+\wedge)$
  R=$(1^*011^*)^*(0+\wedge)+1^*(0+\wedge)$

Language Associated with Regular Expression

- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1} \mid n\geq 0, m\geq 0\}$
- For Σ =$\{0,1\}$. Give a regular expression 'R' such that
  L(R)=$\{w\in \Sigma^* \mid$w has atleast one pair of consecutive zeroes$\}$
  Solution: R=$(0+1)^*00(0+1)^*$
- Find R.E. for language
  L=$\{w \in \{0,1\}^* \mid$w has no pair of consecutive zeroes$\}$
  Solution: R=$(1+01)^*(0+\wedge)$
  R=$(1^*011^*)^*(0+\wedge)+1^*(0+\wedge)$
- Find all strings in L$((a+b)^*b(a+ab)^*)$ of length less than 4

Language Associated with Regular Expression

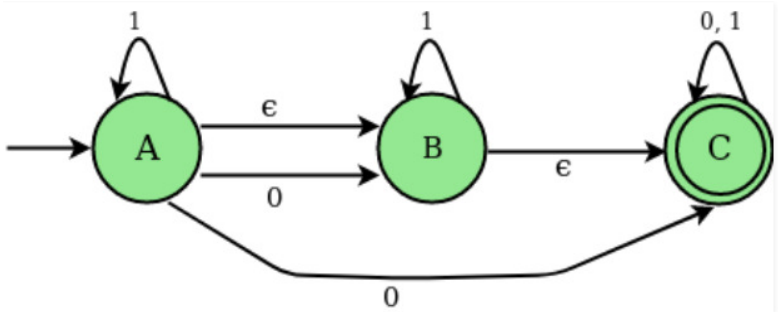
- R=$(aa)^*(bb)^*b$
  denotes the set of all strings with even number of a's followed by an odd number of b's
  L(R)=$\{a^{2n}b^{2m+1} \mid n\geq 0, m\geq 0\}$
- For $\Sigma=\{0,1\}$. Give a regular expression 'R' such that
  L(R)=$\{w\in \Sigma^* \mid$ w has atleast one pair of consecutive zeroes$\}$
  Solution: R=$(0+1)^*00(0+1)^*$
- Find R.E. for language
  L=$\{w \in \{0,1\}^* \mid$ w has no pair of consecutive zeroes$\}$
  Solution: R=$(1+01)^*(0+\wedge)$
  R=$(1^*011^*)^*(0+\wedge)+1^*(0+\wedge)$
- Find all strings in L$((a+b)^*b(a+ab)^*)$ of length less than 4
- Find R.E. for set $\{a^nb^m$:(n+m) is even$\}$

Identities for Regular Expression

1. $\phi + \mathrm{R} = \mathrm{R}$
2. $\phi \mathrm{R} = \mathrm{R}\ \phi = \phi$
3. $\Lambda \mathrm{R} = \mathrm{R} \Lambda = \mathrm{R}$
4. $\Lambda^* = \Lambda$ and $\phi^* = \Lambda$
5. $\mathrm{R} + \mathrm{R} = \mathrm{R}$
6. $R^* R^* = R^*$
7. $\mathrm{R} R^* = R^* \mathrm{R}$
8. $(R^*)^* = R^*$
9. $\Lambda + \mathrm{R} R^* = R^* = \Lambda + R^* R$
10. $(PQ)^* P = \mathrm{P}(QP)^*$
11. $(P + Q)^* = (P^* Q^*)^* = (P^* + Q^*)^*$
12. $(P + Q)R = PR + QR$ and $R(P + Q) = RP + RQ$

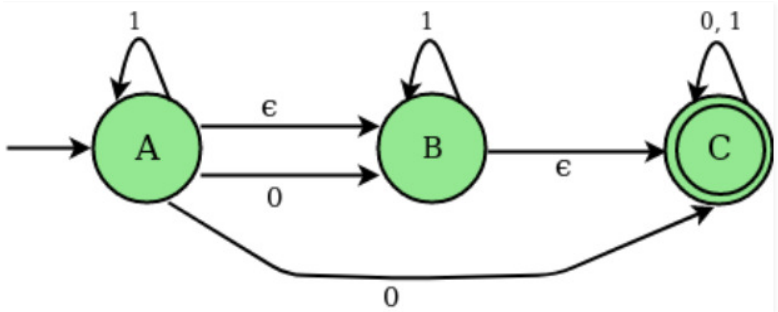
ϵ-NFA

$\Longrightarrow$ Moving without reading a symbol from Input Tape.



ϵ-NFA

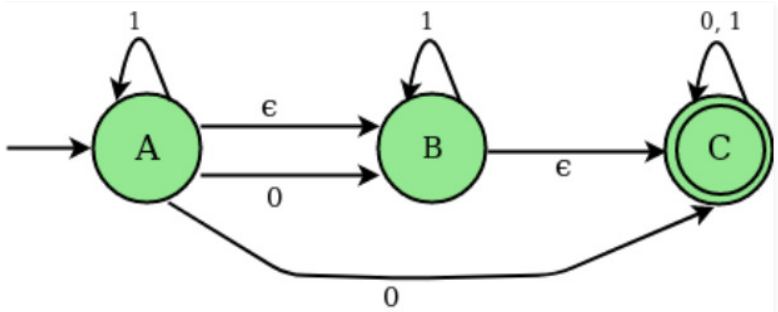
$\Longrightarrow$ Moving without reading a symbol from Input Tape.



State/Input	0	1	ϵ
A	B,C	A	B
B	-	B	C
C	C	C	-

ϵ-NFA

$\Longrightarrow$ Moving without reading a symbol from Input Tape.



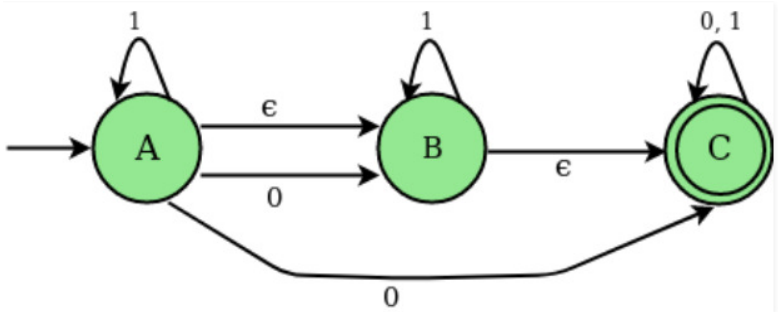
State/Input	0	1	ϵ
A	B,C	A	B
B	-	B	C
C	C	C	-

Epsilon Closure

ϵ-closure for a given state X is a set of States which can be reached from states X with only (null) or ϵ moves including the state X itself.

ϵ-NFA

$\Longrightarrow$ Moving without reading a symbol from Input Tape.



State/Input	0	1	ϵ
A	B,C	A	B
B	-	B	C
C	C	C	-

Epsilon Closure

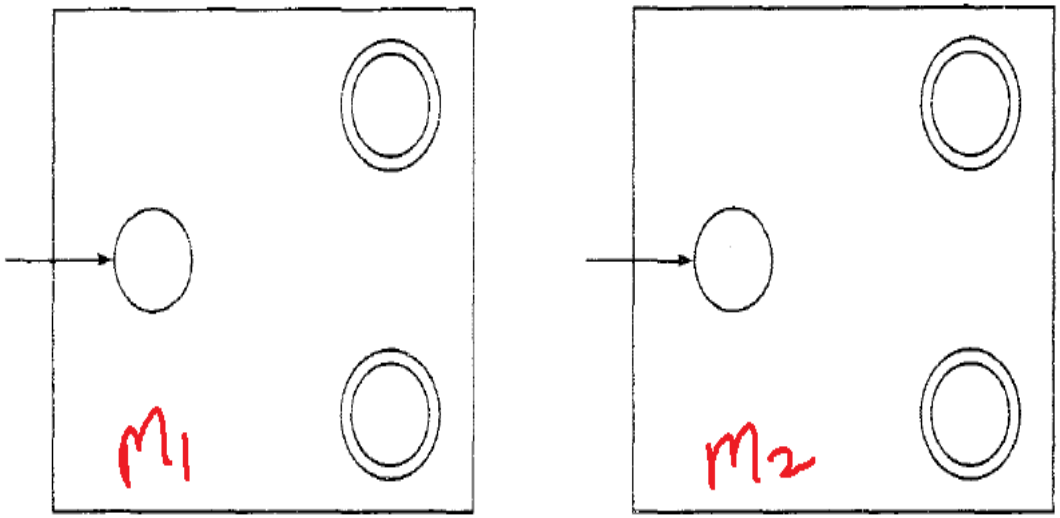
ϵ-closure for a given state X is a set of States which can be reached from states X with only (null) or ϵ moves including the state X itself.

Example: ϵ closure (A)={A,B,C}

ϵ closure (B)={B,C}

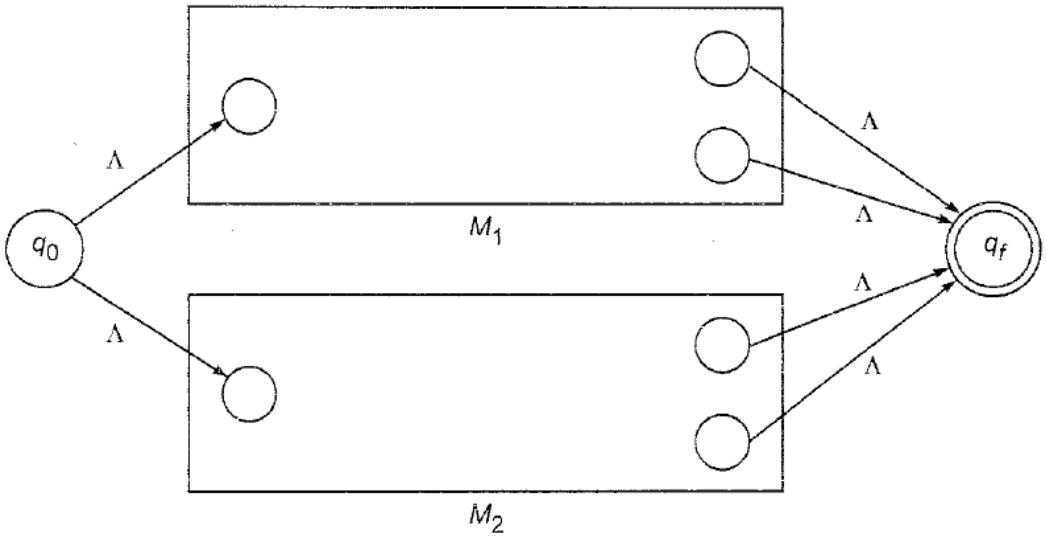
ϵ closure (C)={C}

Automata and Regular Expression



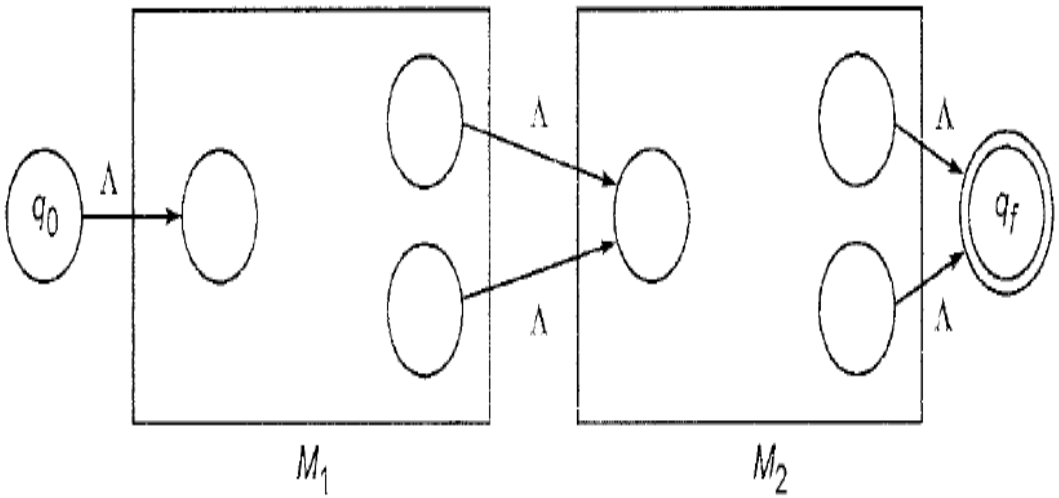
- Automaton for $L(R_1 + R_2)$

Automata and Regular Expression



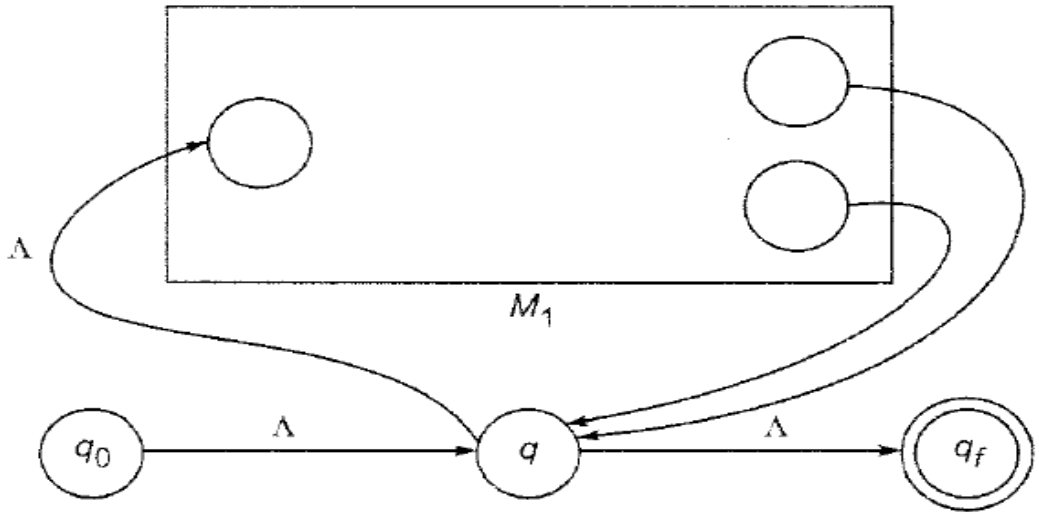
- Automaton for L($R_1.R_2$)

Automata and Regular Expression



- Automaton for $L(R_1{}^*)$

Automata and Regular Expression



Automata and Regular Expression

- Generalized Transition Graph: A transition graph whose edges are labelled as R.E.

Automata and Regular Expression

- Generalized Transition Graph: A transition graph whose edges are labelled as R.E.
- **Example:** L(R)=$(a^* + a^*(a + b)c^*)$

Automata and Regular Expression

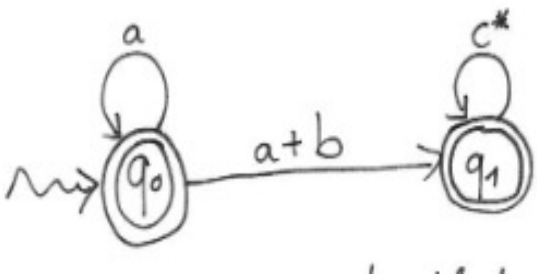
- Generalized Transition Graph: A transition graph whose edges are labelled as R.E.
- **Example:** L(R)=$(a^* + a^*(a + b)c^*)$
- Equivalence of Generalized Transition Graph:
  Let R be a regular expression. Then, there exists some NFA that accepts L(R). Consequently, L(R) is a regular language.
- Find NDFA which accepts L(R) where R=$(a + bb)^*(ba^* + \Lambda)$

Automata and Regular Expression

- The Strings denoted by such regular expressions are a subset of the language accepted by GTG, with full language being the union of all such generated subsets

Automata and Regular Expression

- The Strings denoted by such regular expressions are a subset of the language accepted by GTG, with full language being the union of all such generated subsets
- **Example:** The language accepted by the following GTG is



$L(a^* + a^*(a + b)c*)$

- State Elimination method
- Arden's Theorem

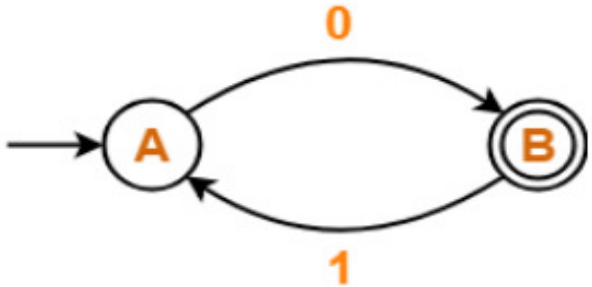
Automata and Regular Expression

State Elimination method

1. Initial State should not have any incoming edge
2. Final State should not have any outgoing edge
3. Only 1 final state
4. Eliminate each non-initial/final vertex one by one

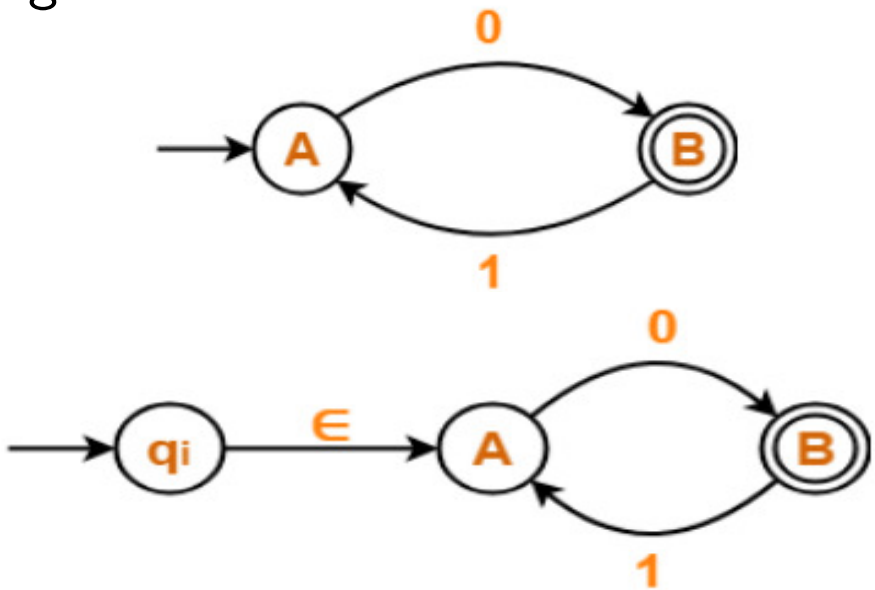
State Elimination Method

1. Find R.E. for given DFA



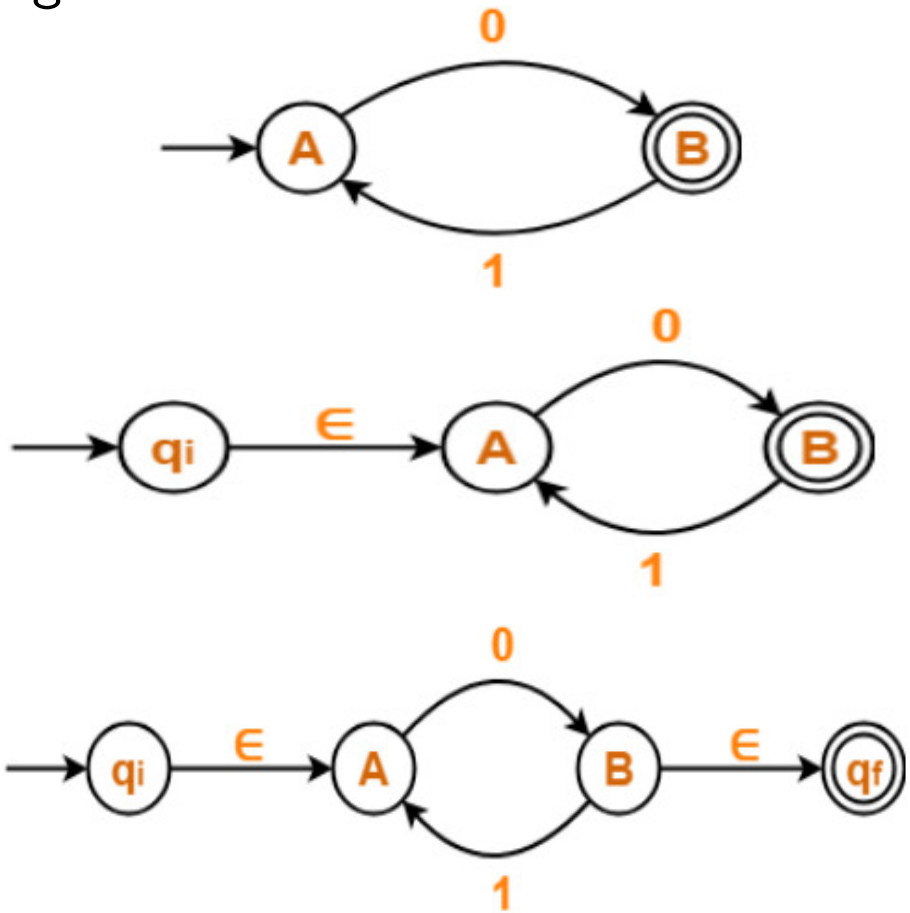
State Elimination Method

1. Find R.E. for given DFA



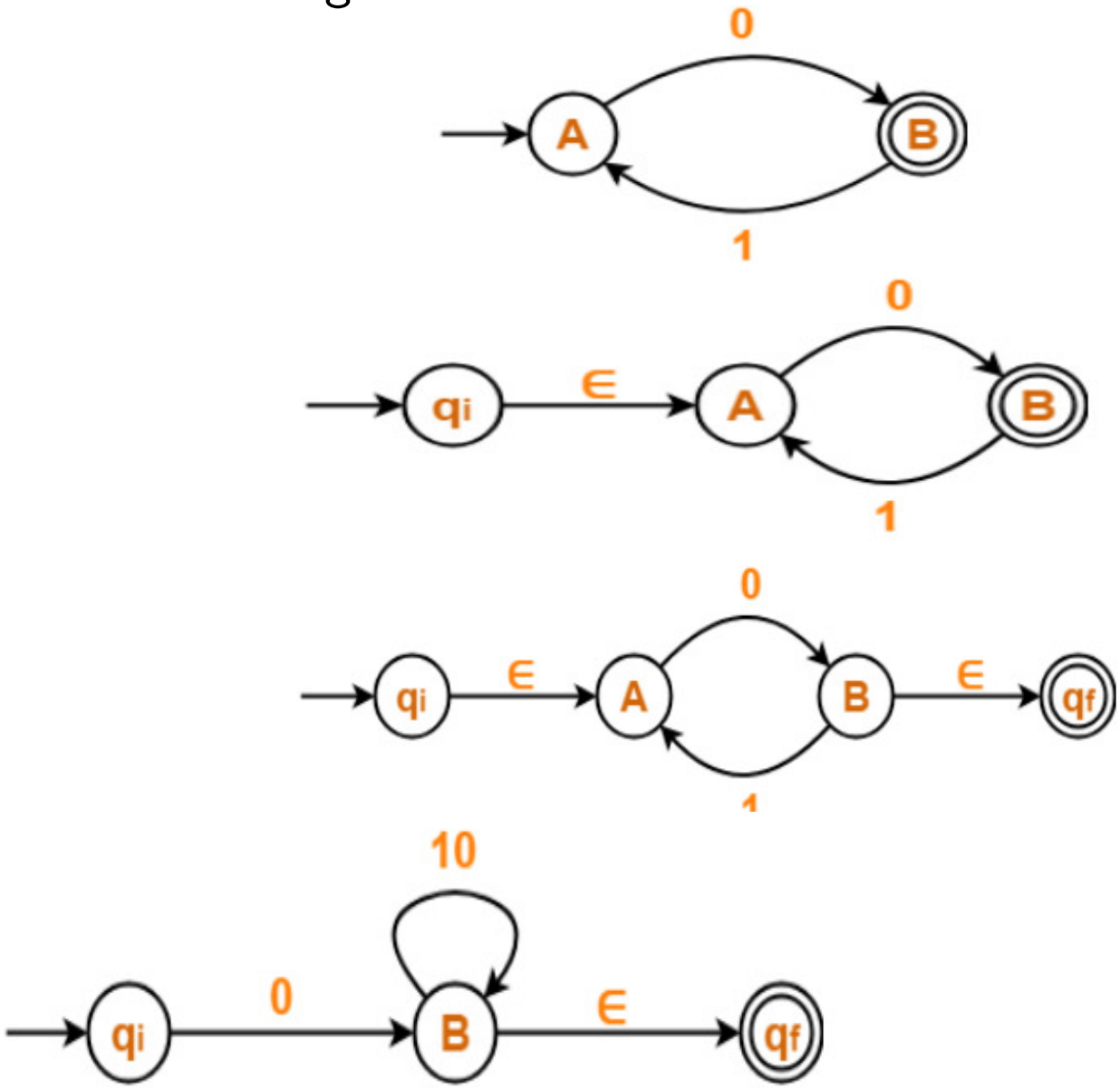
State Elimination Method

1. Find R.E. for given DFA



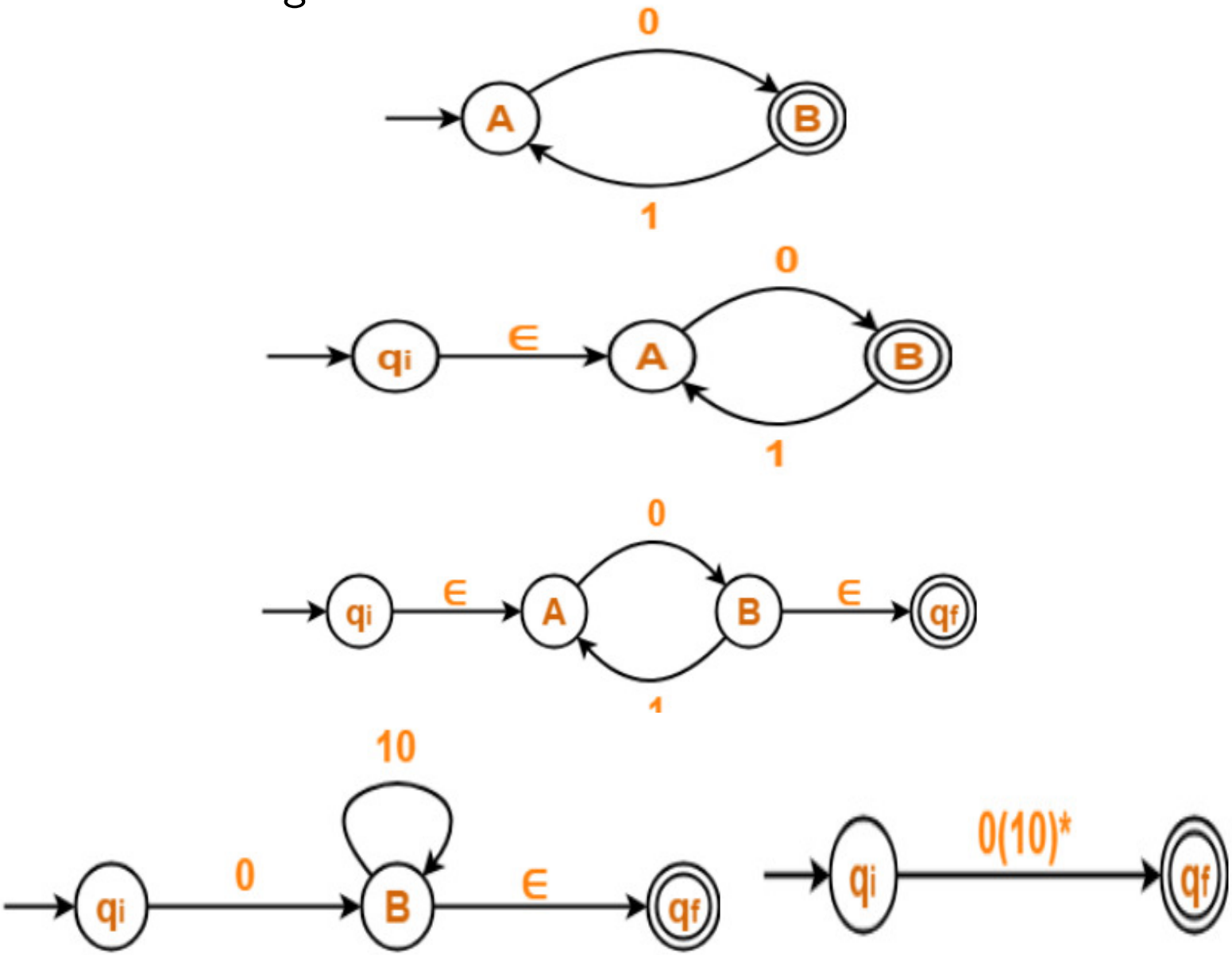
State Elimination Method

1. Find R.E. for given DFA



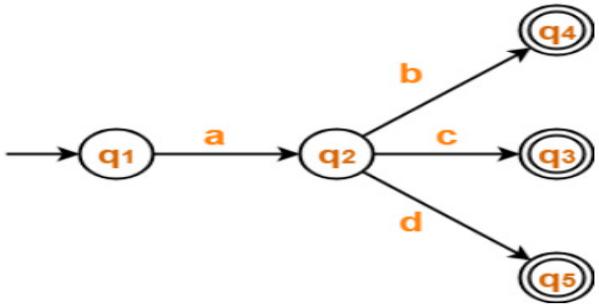
State Elimination Method

1. Find R.E. for given DFA



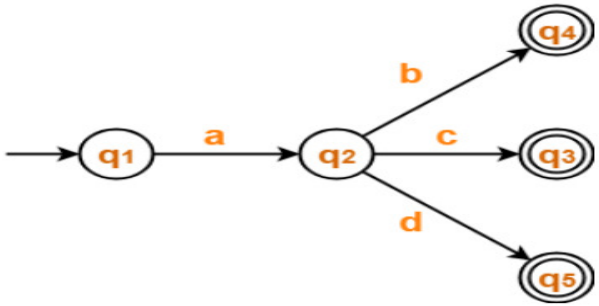
State Elimination Method

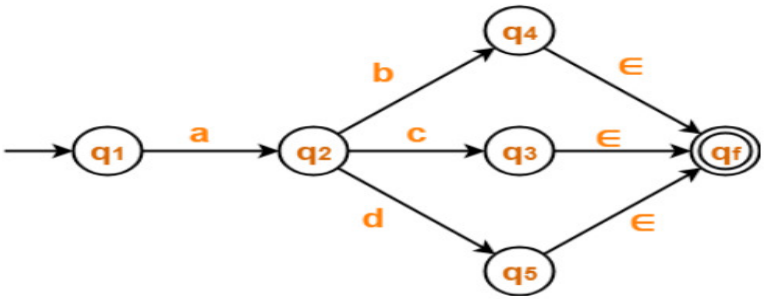
2. Find R.E. for given DFA



State Elimination Method

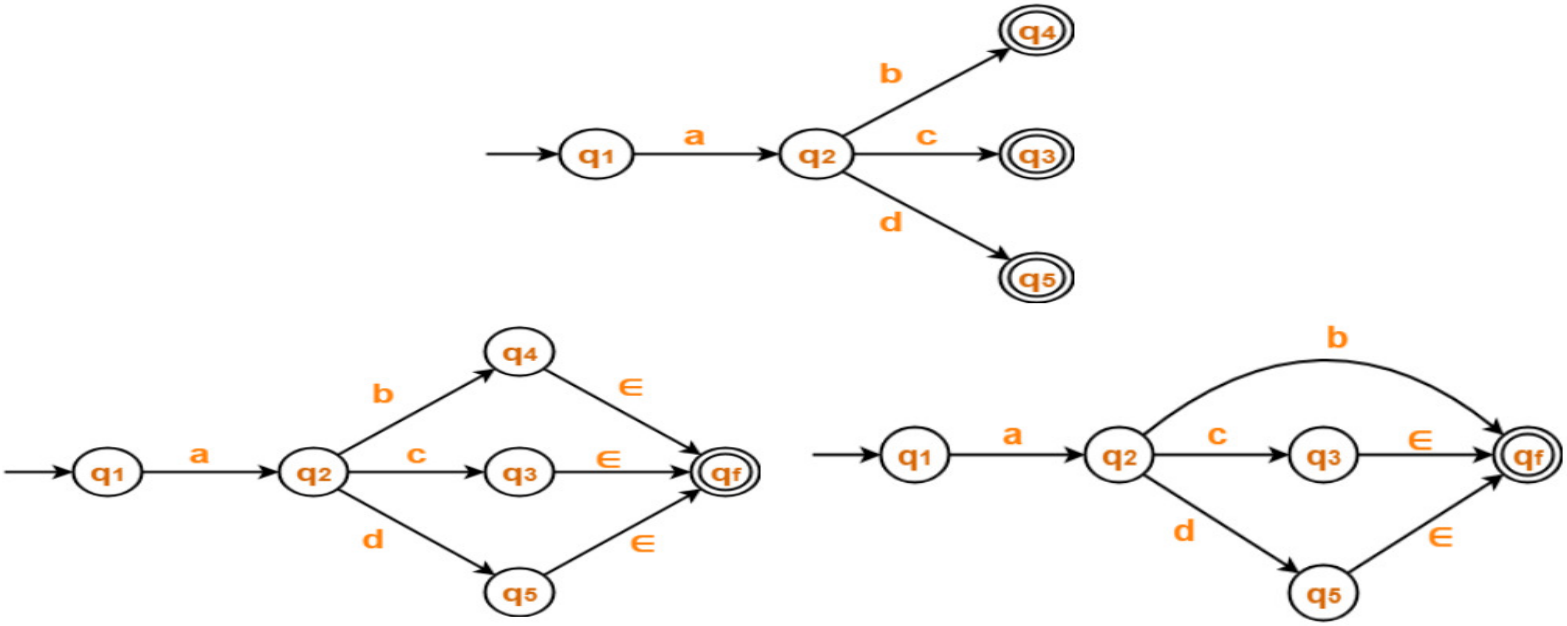
2. Find R.E. for given DFA





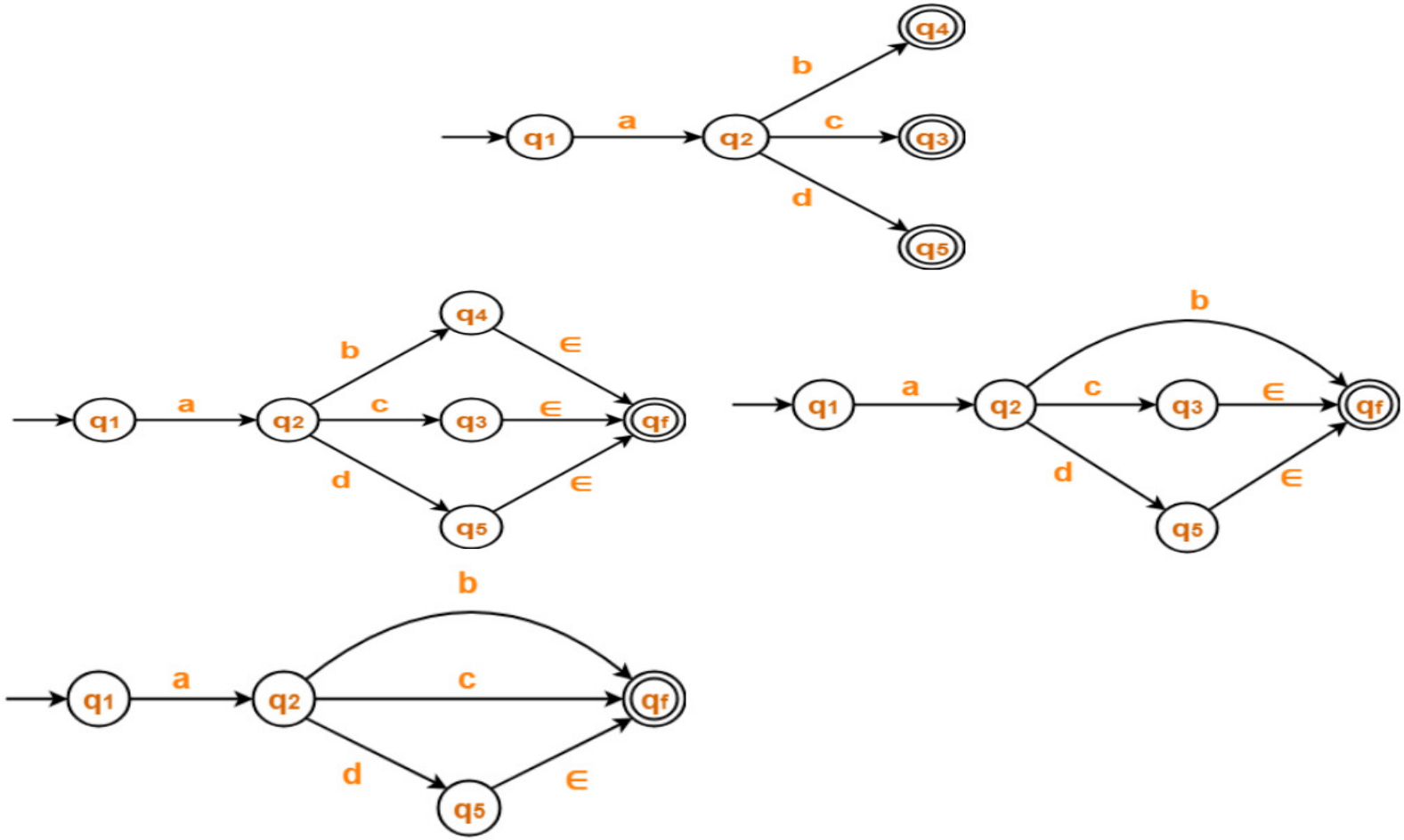
State Elimination Method

2. Find R.E. for given DFA



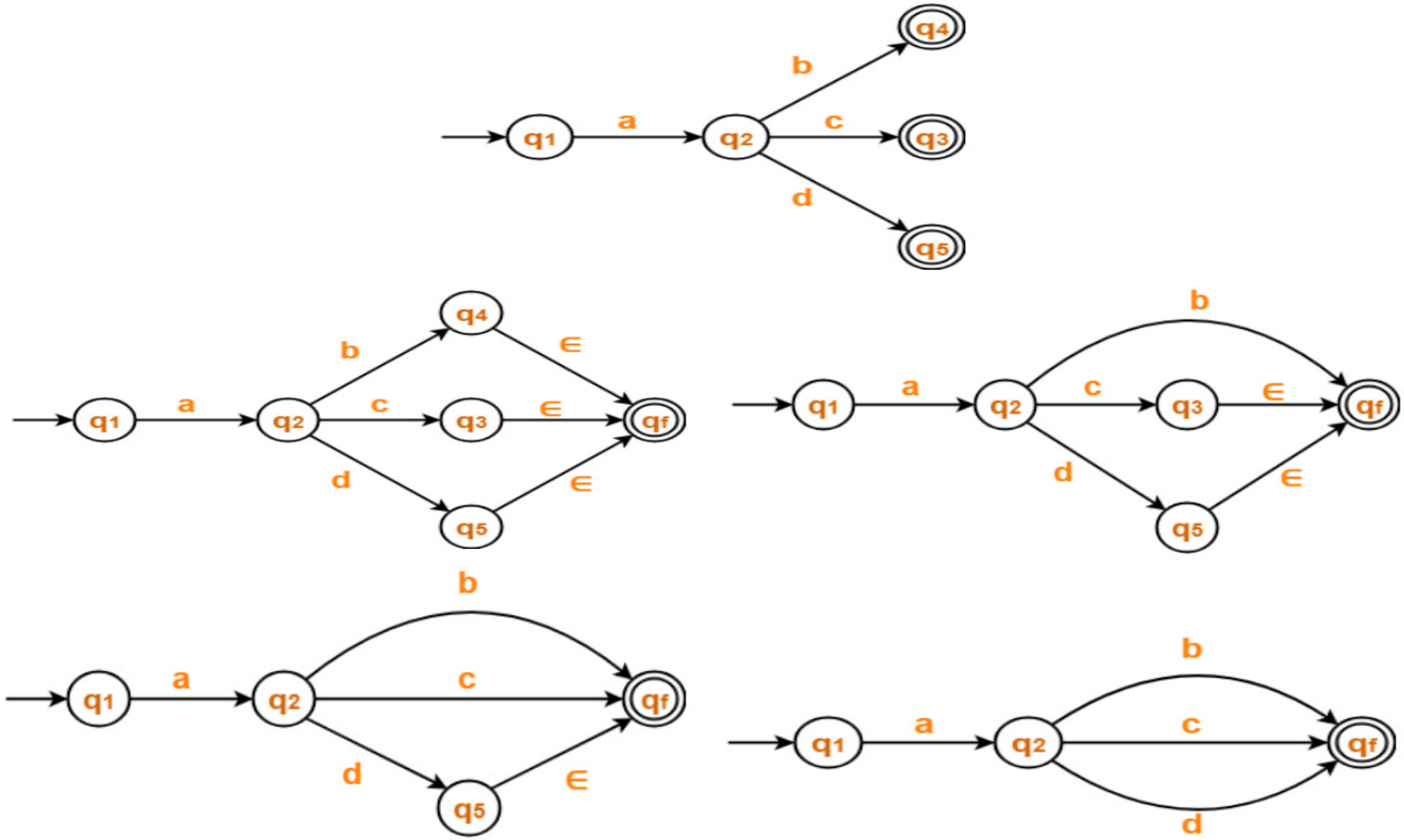
State Elimination Method

2. Find R.E. for given DFA



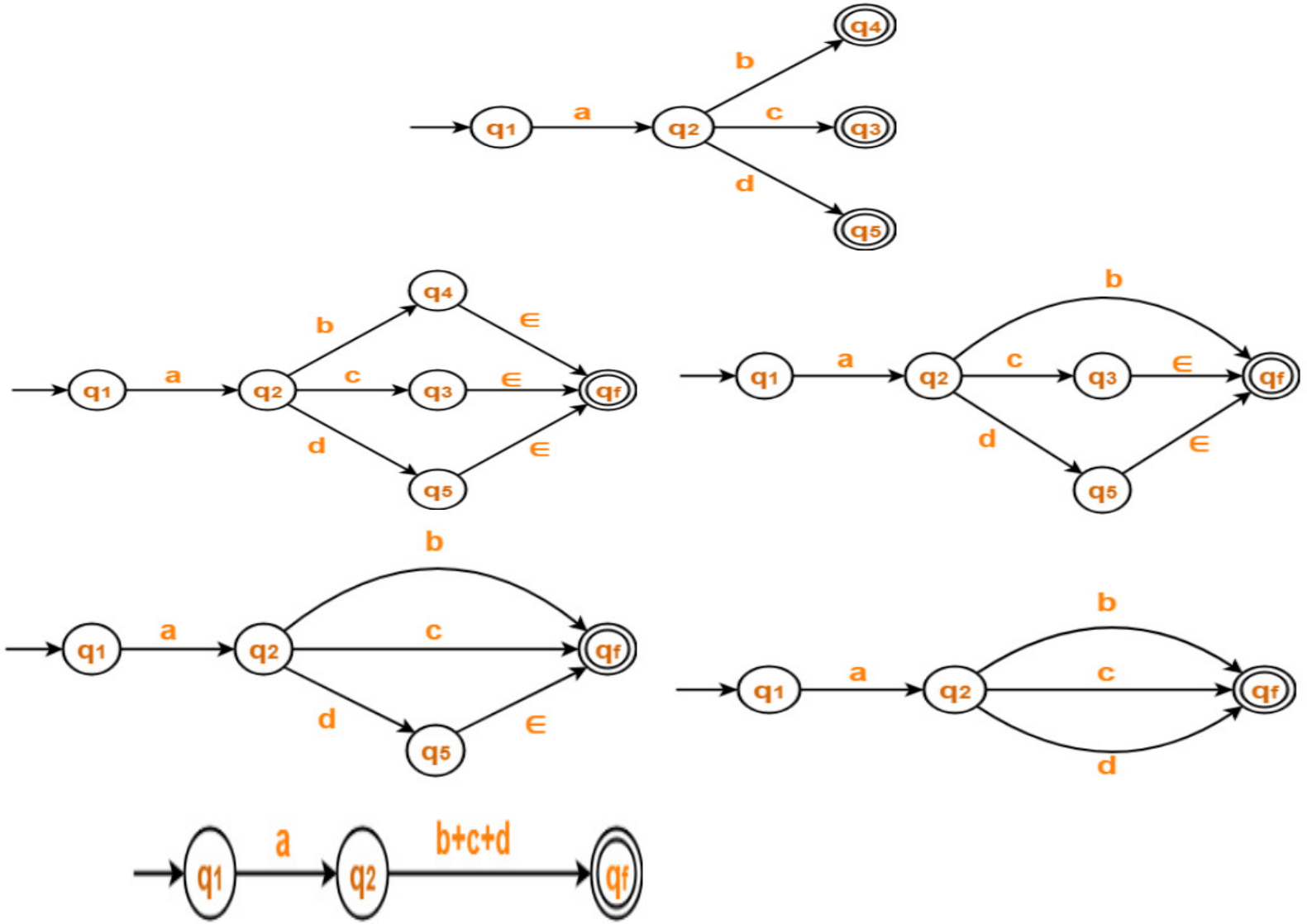
State Elimination Method

2. Find R.E. for given DFA



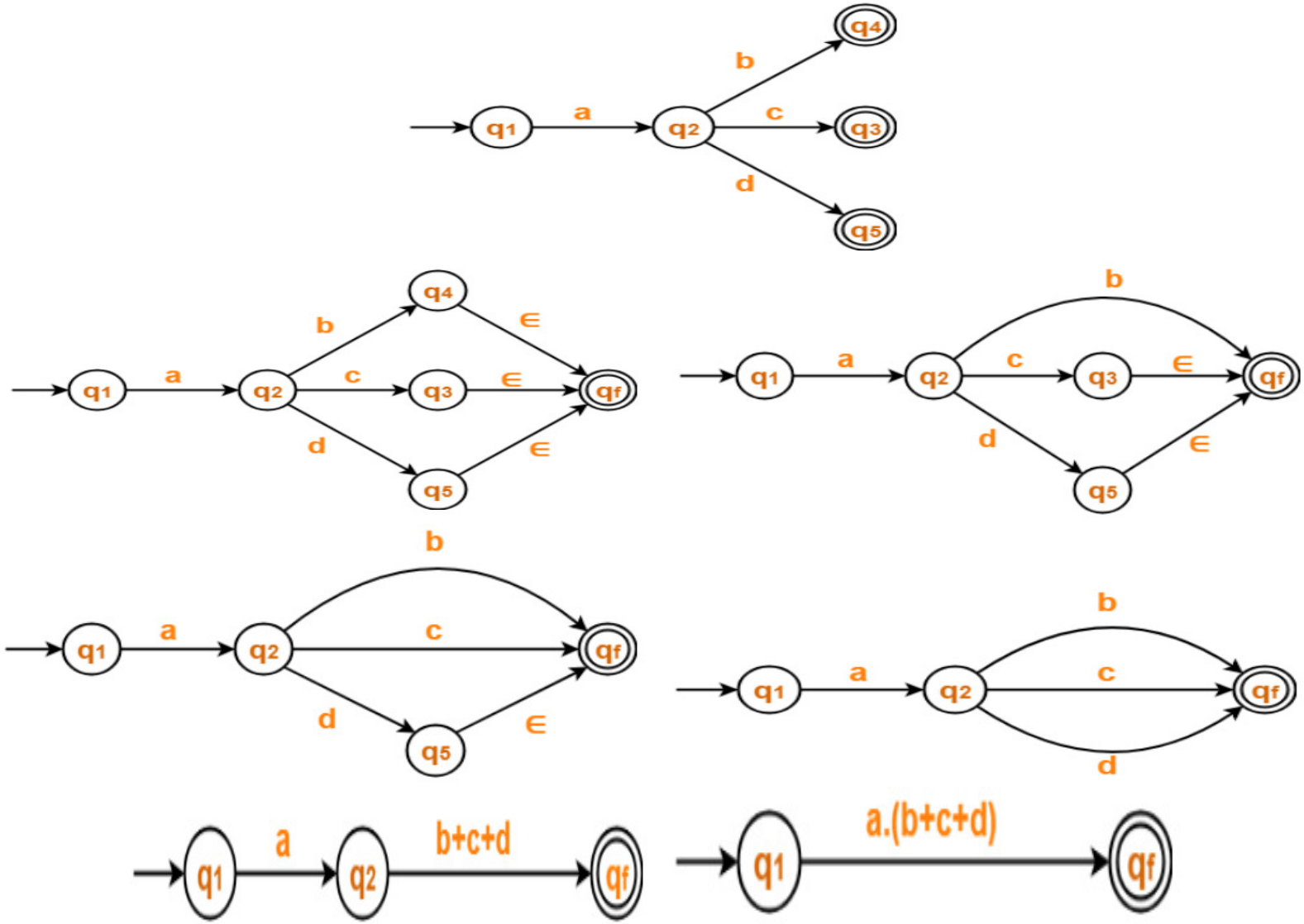
State Elimination Method

2. Find R.E. for given DFA



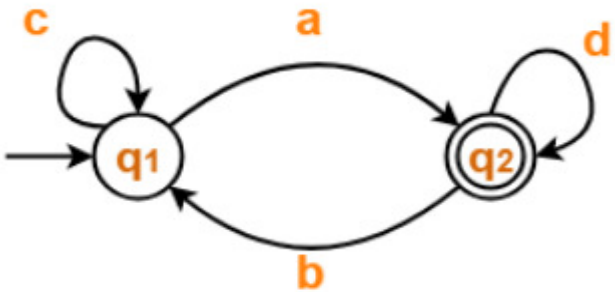
State Elimination Method

2. Find R.E. for given DFA



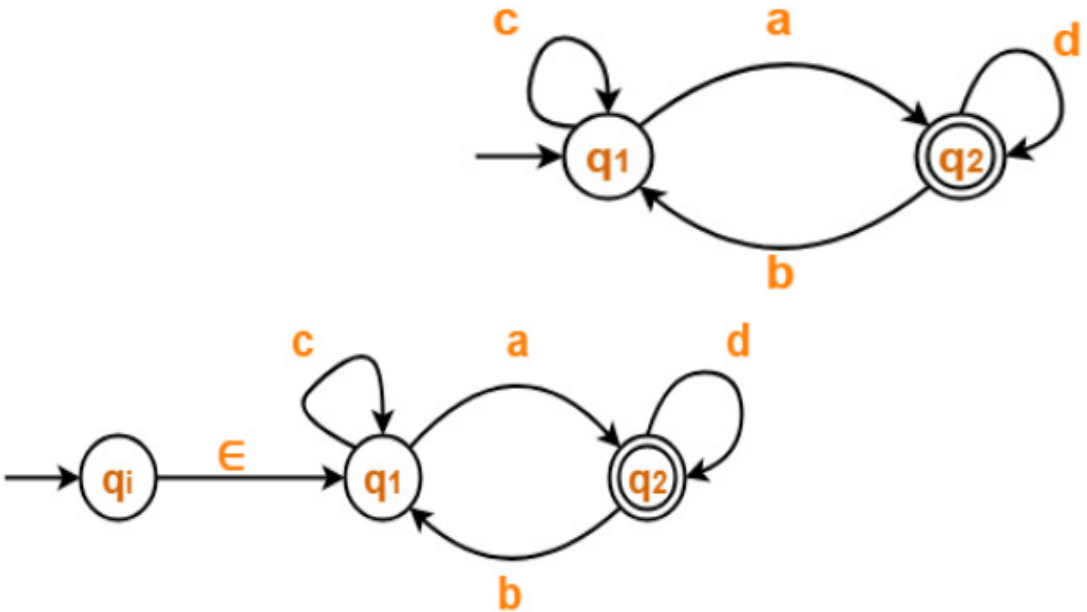
State Elimination Method

3. Find R.E. for given DFA



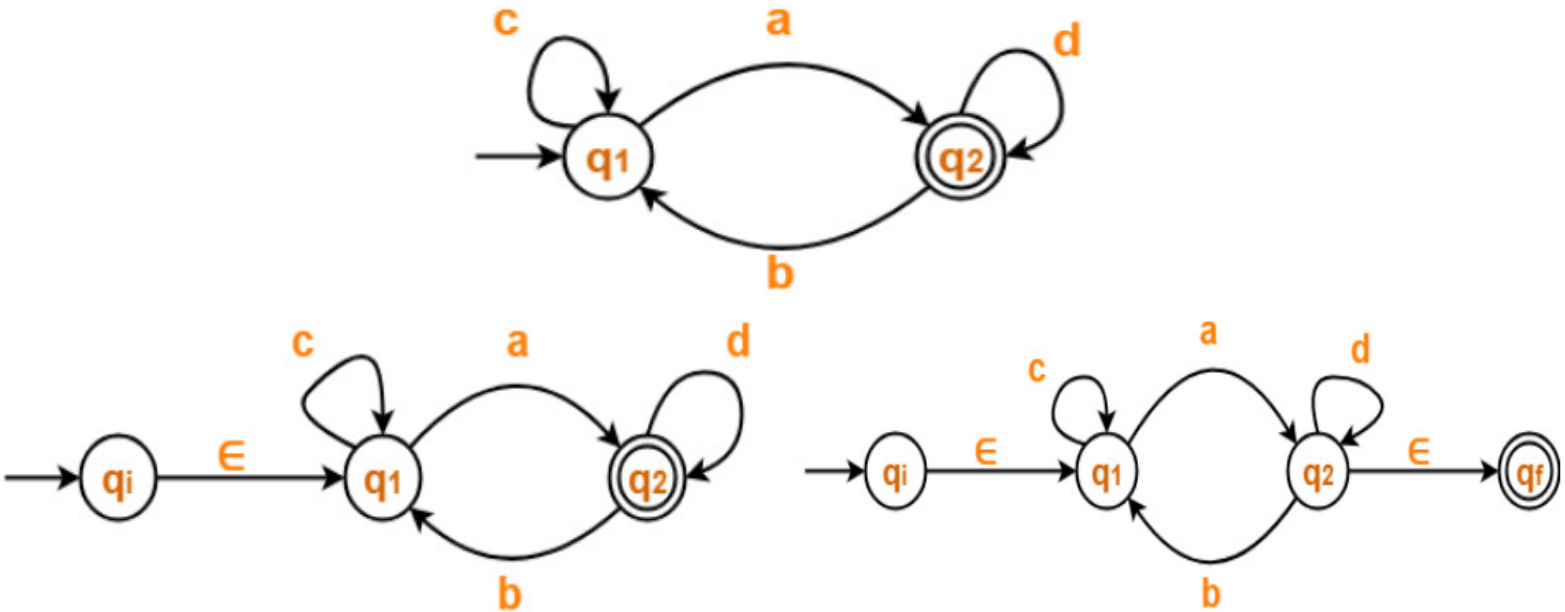
State Elimination Method

3. Find R.E. for given DFA



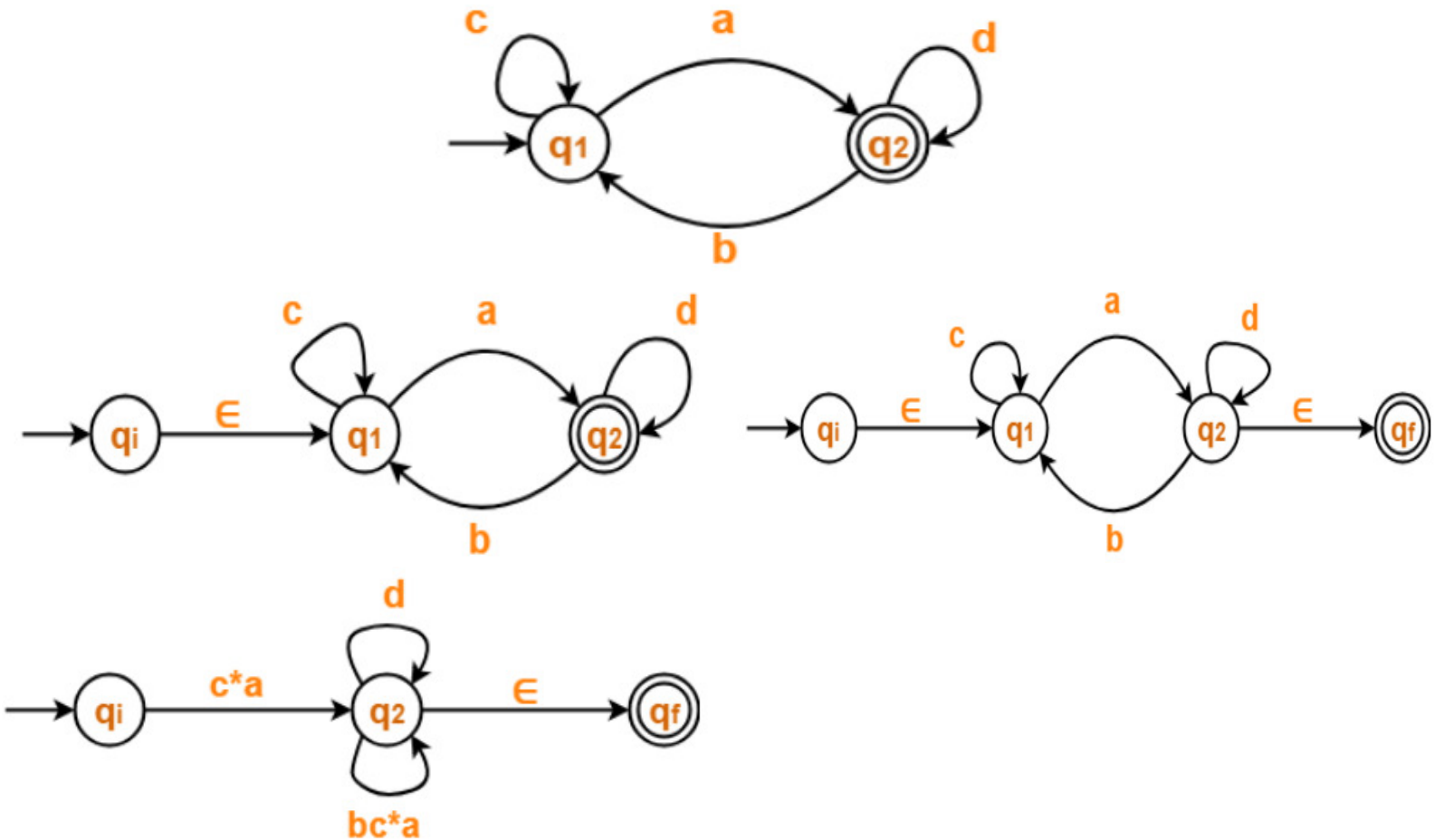
State Elimination Method

3. Find R.E. for given DFA



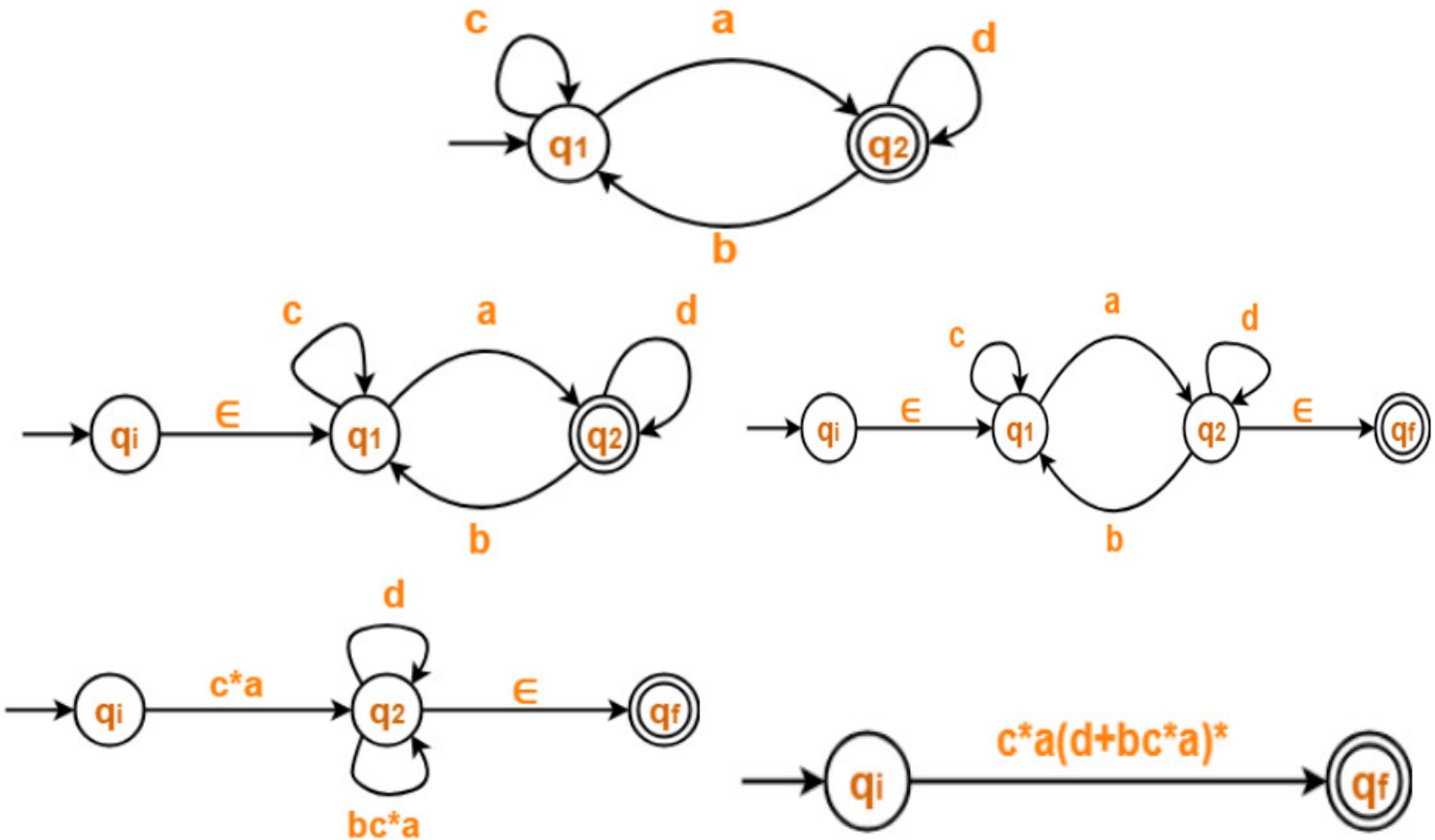
State Elimination Method

3. Find R.E. for given DFA



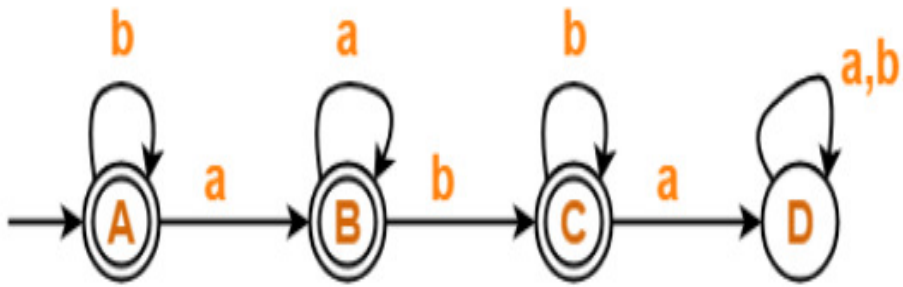
State Elimination Method

3. Find R.E. for given DFA



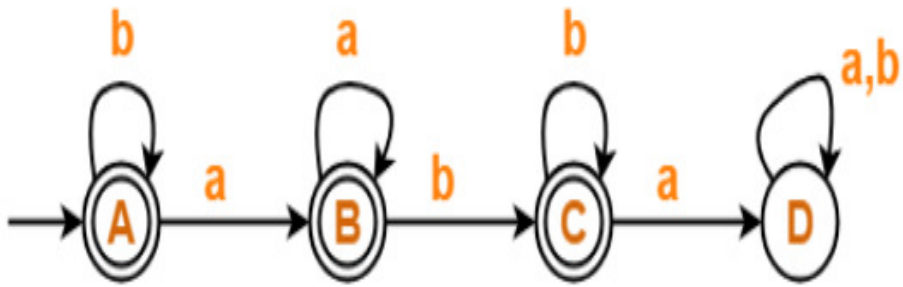
State Elimination Method

1. Find R.E. for the given DFA

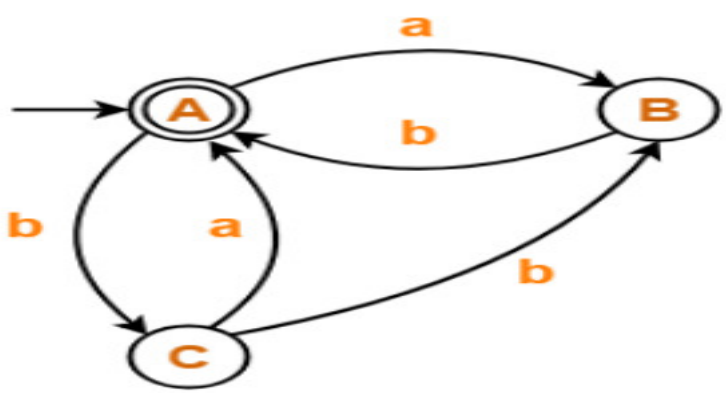


State Elimination Method

1. Find R.E. for the given DFA



2. Find R.E. for the given DFA



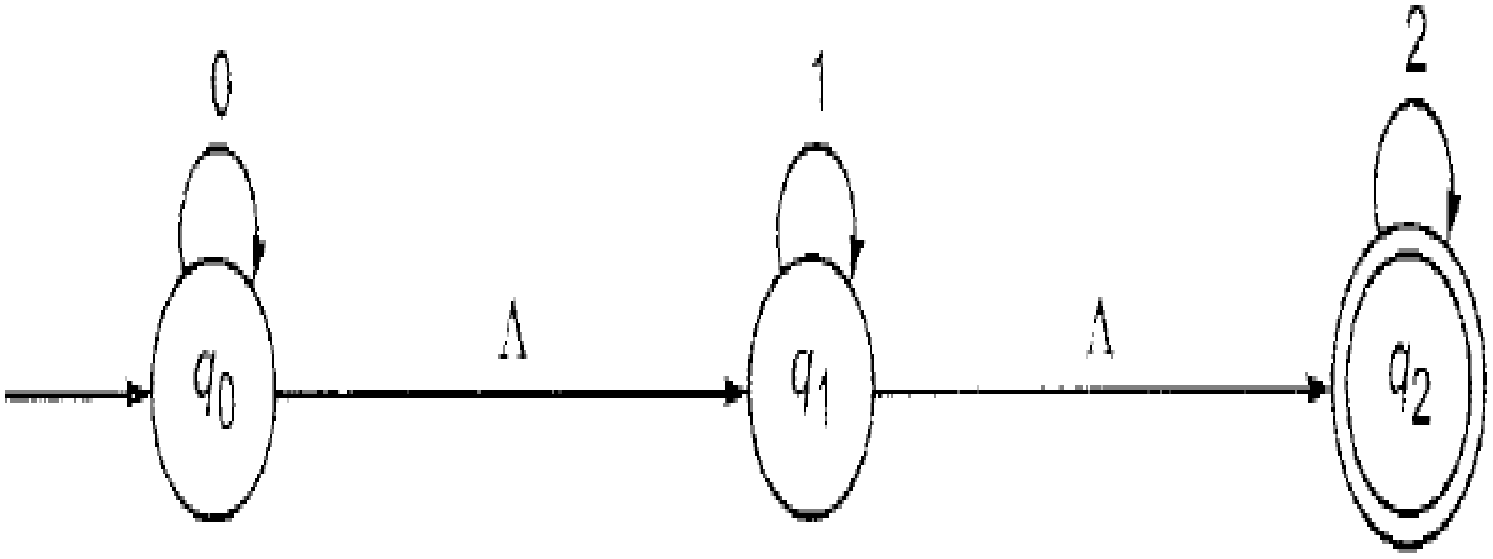
Elimination of $\epsilon/\wedge$ moves

Suppose we want to replace a $\wedge$-move from vertex v_1 to vertex v_2 Then proceed as follows:

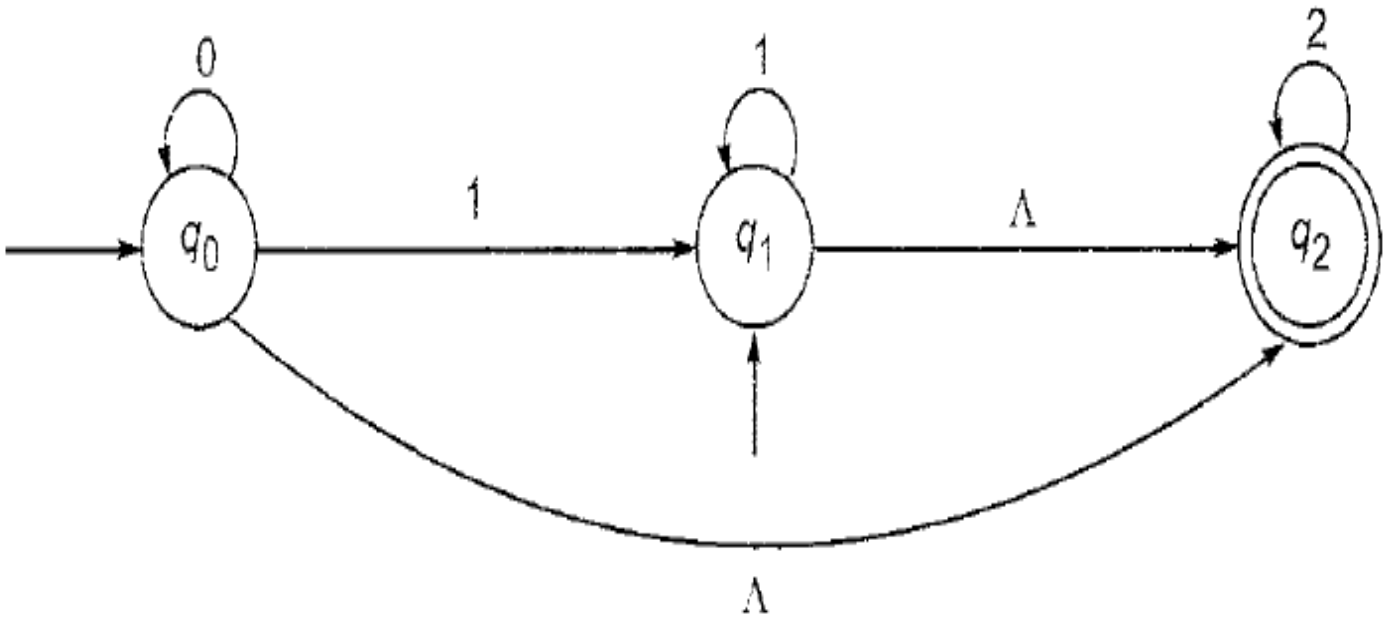
1. Find all the edges starting from v_2
2. Duplicate all these edges starting from v_1, without changing the edge labels.
3. If v_1 is an initial state, make v_2 also as initial state
4. If v_2 is a final state. make v_1 also as the final state

Elimination of $\epsilon/\wedge$ moves

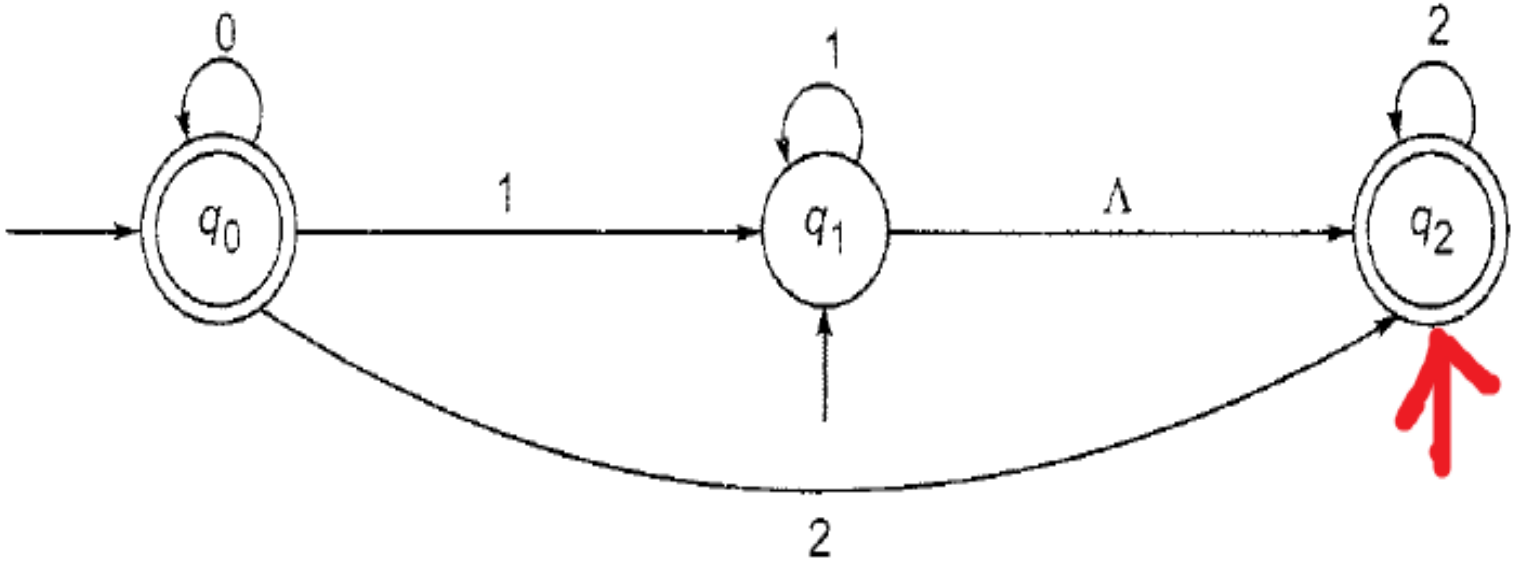
Example:



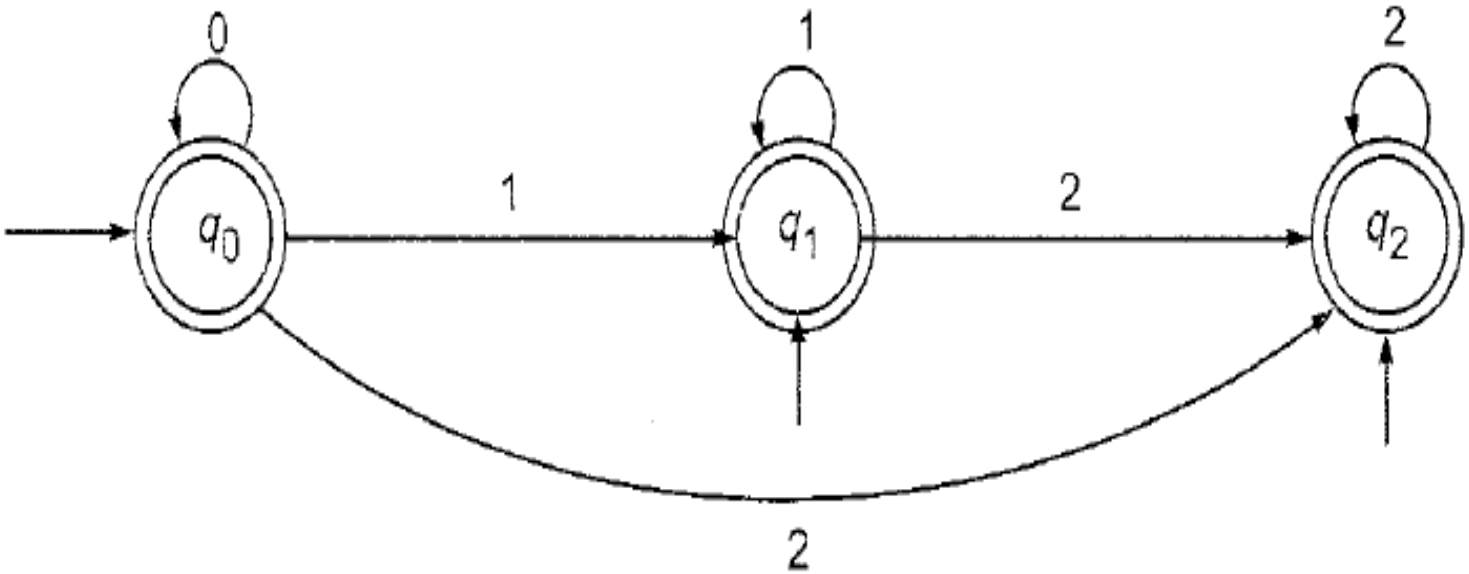
Elimination of $\epsilon/\wedge$ moves



Elimination of $\epsilon/\wedge$ moves

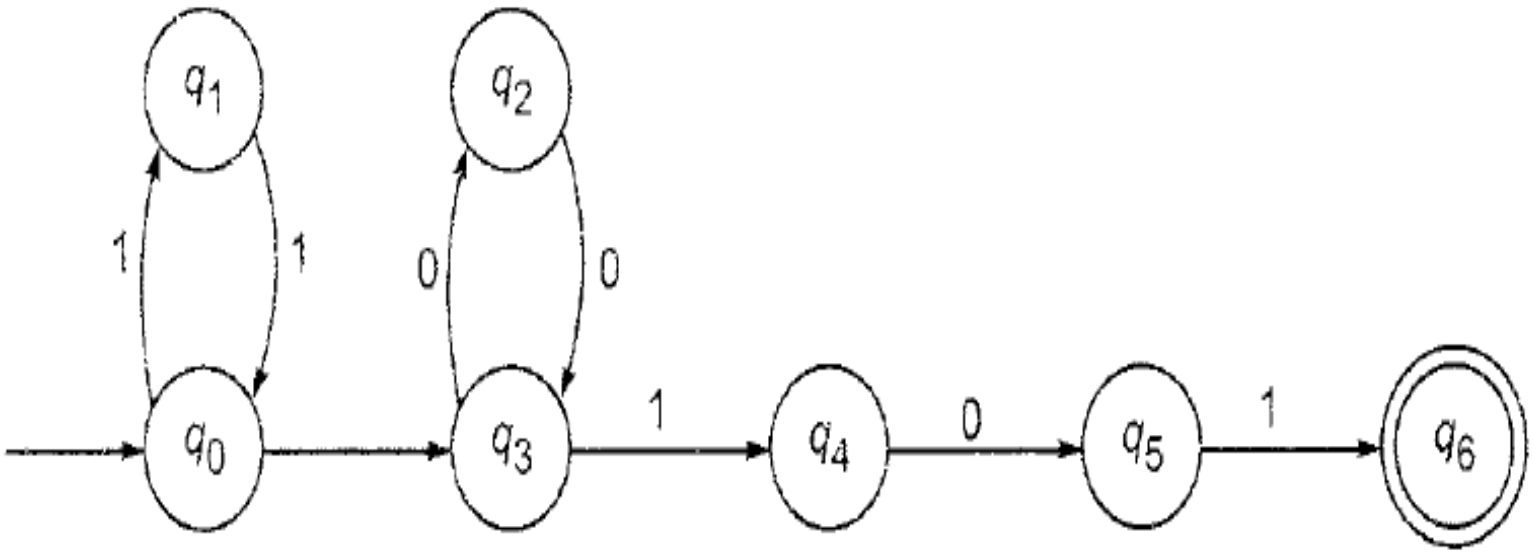


Elimination of $\epsilon/\wedge$ moves



Elimination of $\epsilon/\wedge$ moves

Example:



Conversion of ϵ-NFA to DFA

Steps:

1. Take ϵ closure of initial state as beginning state
2. Find states that can be traversed from present state for each input symbol
3. If any new state is found, repeat step 2 till we get no new state in the transition table.
4. Mark states containing final states as new final state.

State/Input	0	1
{A,B,C}	{B,C}	{A,B,C}

Conversion of ϵ-NFA to DFA

Steps:

1. Take ϵ closure of initial state as beginning state
2. Find states that can be traversed from present state for each input symbol
3. If any new state is found, repeat step 2 till we get no new state in the transition table.
4. Mark states containing final states as new final state.

State/Input	0	1
{A,B,C}	{B,C}	{A,B,C}
{B,C}	{C}	{B,C}

Conversion of ϵ-NFA to DFA

Steps:

1. Take ϵ closure of initial state as beginning state
2. Find states that can be traversed from present state for each input symbol
3. If any new state is found, repeat step 2 till we get no new state in the transition table.
4. Mark states containing final states as new final state.

State/Input	0	1
{A,B,C}	{B,C}	{A,B,C}
{B,C}	{C}	{B,C}
{C}	{C}	{C}

Conversion of ϵ-NFA to DFA

Steps:

1. Take ϵ closure of initial state as beginning state
2. Find states that can be traversed from present state for each input symbol
3. If any new state is found, repeat step 2 till we get no new state in the transition table.
4. Mark states containing final states as new final state.

State/Input	0	1
{A,B,C}	{B,C}	{A,B,C}
{B,C}	{C}	{B,C}
{C}	{C}	{C}

Now, create transition diagram

Arden's Theorem

Let P and Q be two regular expressions over Σ. **If P does not contain** Λ, then the following equation in R i.e. R=Q+RP has a unique solution R=QP^*

Proof:

$$R = Q + RP \tag{1}$$

putting "R=Q+RP" in equation 1

R=Q+QP+RPP

putting R recursively again and again

we get:

R=Q+QP+QP^2+QP^3+ ...

R= Q $(\Lambda + P + P^2 + P^3 + \ldots)$

R=QP^*

Arden's Theorem

Assumptions:

- The transition diagram must not have null transitions.
- It must only 1 initial state, let v_1
- Its vertices are $v_1, \ldots, v_n$.
- V_i, the R.E. represents the set of strings accepted by the system even though v_i, is a final state.
- α_{ij} denotes the R.E. representing the set of labels of edges from v_i to v_j. When there is no such edge, $\alpha_{ij} = \phi$. Consequently, we can get the following set of equations in $V_1, \ldots, V_n$:

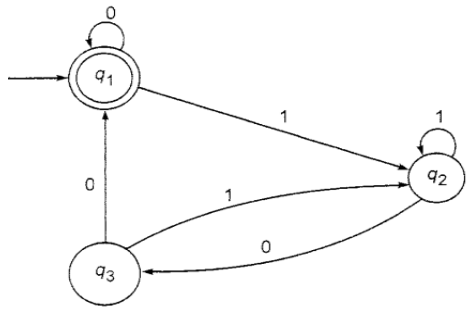
$$V_1 = V_1\alpha_{11} + V_2\alpha_{21} + \cdots + V_n\alpha_{n1} + \wedge$$
$$V_2 = V_1\alpha_{12} + V_2\alpha_{22} + \cdots + V_n\alpha_{n2}$$
$$\vdots$$
$$V_n = V_1\alpha_{1n} + V_2\alpha_{2n} + \cdots + V_n\alpha_{nn}$$

Arden's Theorem

- By repeatedly applying substitutions and Arden's theorem, we can express V_i in terms of α_{ij}.
- For getting the set of strings recognized by the transition system, we have to take the "union" of all V_i corresponding to final states

Arden's Theorem

1. Find R.E. for the following DFA



$$q_1 = q_1 0 + q_3 0 + \wedge \tag{2}$$

$$q_2 = q_1 1 + q_2 1 + q_3 1 \tag{3}$$

$$q_3 = q_2 0 \tag{4}$$

putting equation 4 in equation 3

$$\begin{aligned} q_2 &= q_1 1 + q_2 1 + q_3 1 \\ &= q_1 1 + q_2 1 + (q_2 0) 1 \\ &= q_1 1 + q_2 (1 + 01) \end{aligned}$$

Arden's Theorem

Applying Arden's Theorem

$$q_2 = q_1 1(1+01)^* \tag{5}$$

putting 5 in equation 2

$$\begin{aligned} q_1 &= q_1 0 + q_3 0 + \wedge \\ &= q_1 0 + q_2 00 + \wedge \\ &= q_1 0 + (q_1 1(1+01)^* 00) + \wedge \\ &q_1(0 + 1(1+01)^* 00) + \wedge \end{aligned}$$

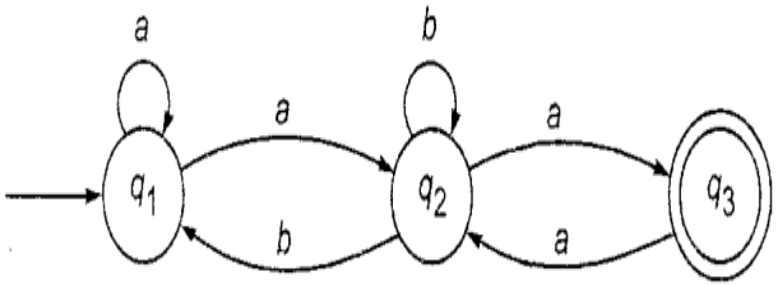
using arden's theorem again

$$\begin{aligned} q_1 = \wedge &(0 + 1(1+01)^* 00)^* \\ &(0 + 1(1+01)^* 00)^* \end{aligned}$$

As q_1 is the final state. $\therefore$ r= $(0 + 1(1+01)^* 00)^*$

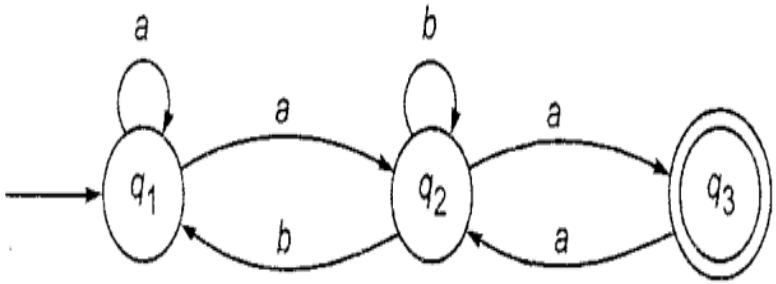
Arden's Theorem

- Consider the transition system below. Prove that the strings recognized are $(a + a(b + aa)^* b)^* a(b + aa)^* a$



Arden's Theorem

- Consider the transition system below. Prove that the strings recognized are $(a + a(b + aa)^* b)^* a(b + aa)^* a$



Solution:

$$q_1 = q_1 a + q_2 b + \wedge \tag{6}$$

$$q_2 = q_1 a + q_2 b + q_3 a \tag{7}$$

$$q_3 = q_2 a \tag{8}$$

Arden's Theorem

Putting equation 8 in equation 7

$$q_2 = q_1 a + q_2 b + q_2 aa$$
$$q_2 = q_1 a + q_2 (b + aa)$$
$$q_2 = q_1 a (b + aa)^*$$

Now putting q_2 in equation 6

$$q_1 = q_1 a + q_2 b + \wedge$$
$$= q_1 a + q_1 a (b + aa)^* b + \wedge$$
$$= q_1 (a + a(b + aa)^* b) + \wedge$$
$$= \wedge (a + a(b + aa)^* b)^*$$
$$= (a + a(b + aa)^* b)^*$$

putting this in q_2

Arden's Theorem

$$\begin{aligned} q_2 &= (a + a(b + aa)^* b)^* a + q_2 b + q_2 aa \\ &= (a + a(b + aa)^* b)^* a + q_2 (b + aa) \\ &= (a + a(b + aa)^* b)^* a(b + aa)^* \end{aligned}$$

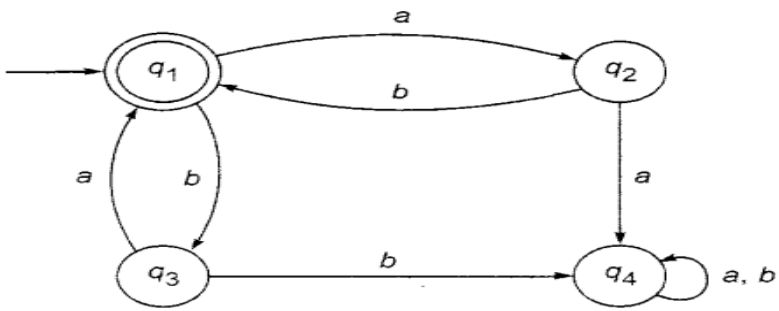
putting q_2 in q_3

$$q_3 = (a + a(b + aa)^* b)^* a(b + aa)^* a \tag{9}$$

- Since q_3 is the final state.
  $\therefore$ r$= (a + a(b + aa)^* b)^* a(b + aa)^* a$

Arden's Theorem

- Prove that the finite automaton whose transition diagram below accepts the set of all strings over alphabet {a,b} with an equal number of **a's** and **b's**, such that each prefix has atmost has atmost one more **a** than the **b's** and atmost one more **b** than the **a's**



Arden's Theorem

Solution:

$$q_1 = q_2 b + q_3 a + \wedge \quad (10)$$

$$q_2 = q_1 a \quad (11)$$

$$q_3 = q_1 b \quad (12)$$

$$q_4 = q_2 a + q_3 b + q_4 a + q_4 b \quad (13)$$

putting q_2 and q_3 in q_1

$$q_1 = q_1 ab + q_1 ba + \wedge$$

$$q_1 = q_1(ab + ba) + \wedge$$

applying Arden's theorem

$$q_1 = \wedge(ab + ba)^*$$

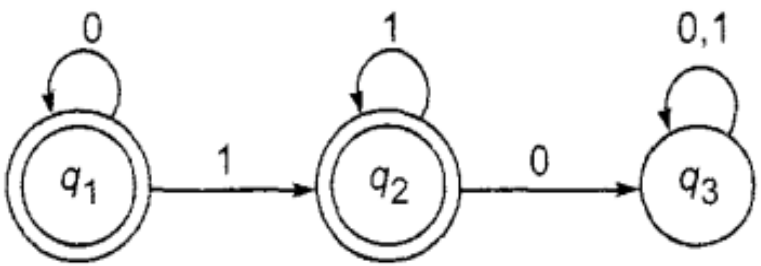
$$q_1 = (ab + ba)^*$$

Arden's Theorem

- Now, the prefix can be even or odd in length. For Prefix x of even length, the number of a's and b's shall be equal as x is a substring formed by ab's and ba's. For prefix x of odd length, then we can write 'x' as ya or yb. As y has even number of symbols, which implies x has one more a than b or vice-versa

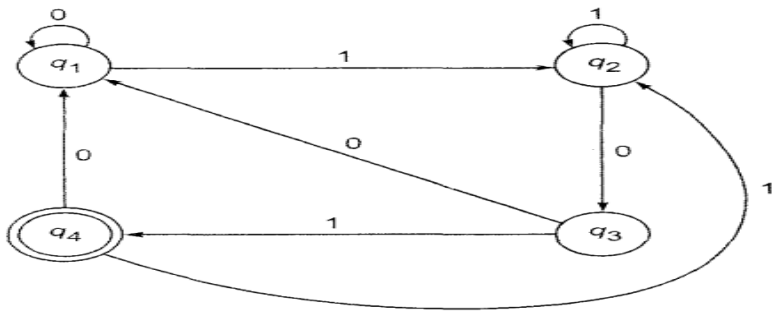
Arden's Theorem

- Describe in English the set accepted by finite automaton whose transition diagram is as under:



Arden's Theorem

- Construct a regular expression corresponding to the state diagram described as under:



- Give R.E. for representing the set L of strings in which every 0 is immediately followed by atleast two 1's. Prove that R.E. r= $\wedge + 1^*(011)^*(1^*(011)^*)^*$ also describes the same set of strings.
- Prove $(1+00^*1)+(1+00^*1)(0+10^*1)^*(0+10^*1) = 0^*1(0+10^*1)^*$

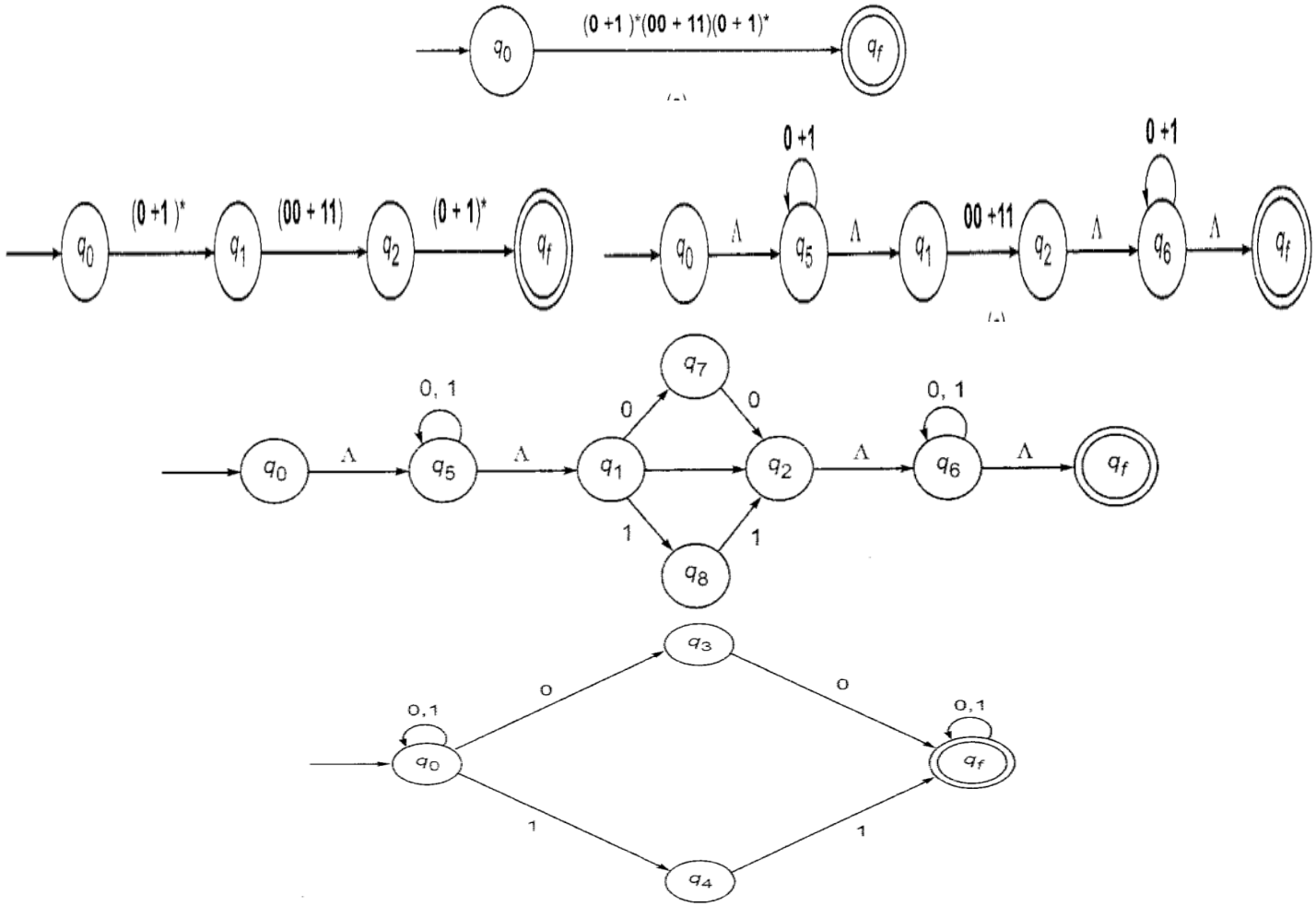
Construction of FA equivalent to given RE

The method for constructing a finite automaton equivalent to a given regular expression is called the subset method which involves four steps.

1. Construct a transition system equivalent to the given regular expression using $\wedge$-moves.
2. Construct the transition table for the transition graph obtained in step 1.
3. Construct the DFA equivalent to NDFA.
4. Reduce the number of states if possible.

- Construct FA equivalent to Regular Expression.
  $(0+1)^*(00+11)(0+1)^*$
  Solution:

Construction of FA equivalent to given RE



Construction of FA equivalent to given RE

State/Σ	0	1
q_0	q_0, q_3	q_0, q_4
q_3	q_f	
q_4		q_f
q_f	q_f	q_f

converting to DFA

State/Σ	0	1
q_0	q_0, q_3	q_0, q_4
q_0, q_3	q_0, q_3, q_f	q_0, q_4
q_0, q_4	q_0, q_3	q_0, q_4, q_f
q_0, q_3, q_f	q_0, q_3, q_f	q_0, q_4, q_f
q_0, q_4, q_f	q_0, q_3, q_f	q_0, q_4, q_f

reducing

State/Σ	0	1
q_0	q_0, q_3	q_0, q_4
q_0, q_3	q_0, q_3, q_f	q_0, q_4
q_0, q_4	q_0, q_3	q_0, q_3, q_f
q_0, q_3, q_f	q_0, q_3, q_f	q_0, q_3, q_f

Construction of FA equivalent to given RE

1. Construct DFA with reduced states equivalent to R.E. $(10+(0+11)0^*1)$.
2. Construct transition system equivalent to R.E.
   - $(ab + c^*)^* b$
   - $a + bb + bab^* a$
   - $(a + b)^* abb$
3. Prove that $(a^*ab + ba)^* a^* = (a + ab + ba)^*$
4. Construct a finite automata accepting all strings over $\{0,1\}$ ending in 010 or 0010.
5. Construct a regular grammar which can generate the set of all strings starting with a letter (A to Z) followed by a string of letters or digits (0 to 9).

2-way DFA

- 2-way DFA allows the read head to move left or right on the input
- Two end-markers
- Needs only 1 accept or reject state.
- A tuple $M = \{Q, \Sigma, \vdash, \dashv, \delta, s, t, r\}$
  where Q is the set of states
  Σ is the input alphabet set
  $\vdash$ is the left end marker
  $\dashv$ is the right end marker
  δ is $Q \times (\Sigma \cup \{\vdash, \dashv\}) \rightarrow Q \times \{L, R\}$
  s is start state
  t is the accept state
  r is reject state such that $r \neq t$

2-Way DFA

- Determine the acceptability of 101001 for the following:

State/ Σ	0	1
$\rightarrow q_0$	$(q_0$,R)	$(q_1$, R)
q_1	$(q_1$,R)	$(q_2$, L)
q_2	$(q_0$,R)	$(q_2$, L)

where Q=$\{q_0, q_1, q_2\}$, s=q_0, t=q_1, r=q_2

Pumping Lemma for Regular Sets

- Let M = (Q, Σ, δ, q_0, F) be a finite automaton with 'n' states. Let L be the regular sets accepted by M.
- Let w $\in$ L and |w |$\geq$ n, then $\exists$ x,y,z such that w=xyz, y $\neq \wedge$ and $xy^iz \in$ L for each i $\geq$ 0.
- **Applications of Pumping Lemma:** Used to prove that certain set are not regular.
- **Steps to prove that given set is not regular:**
  1. Assume L is regular. Let 'n' be the number of states in corresponding FA.
  2. Choose a string 'w' such that |w |$\geq$ n. Use pumping lemma to write w=xyz with |xy |$\leq$ n and |y |$>$ 0
  3. Fing a suitable integer i such that $xy^iz \notin$ L. This contradicts our assumption. Hence, L is not regular.

Pumping Lemma for Regular Sets

Show that the set L $=\{a^{i^2} \mid i \leq 1\}$ is not regular

Solution:

- Let L is regular
  Let 'n' be number of states in FA accepting L.
- Let w$=a^{n^2} \implies |w| = n^2 > n$
  by pumping lemma, w=xyz with $|xy| \leq$ n and $|y| > 0$
- Consider xy^2z
  $|xy^2z| = |x| + 2|y| + |z| > |x| + |y| + |z| \because |y| > 0$
  $\implies n^2 = |xyz| = |x| + |y| + |z| < |xy^2z|$
  As $|xy| \leq$ n, $|y| \leq$ n
- $\therefore |xy^2z| = |x| + 2|y| + |z| \leq n^2 + n < n^2 + n + n + 1$.
  Hence, $|xy^2z|$ lies between n^2 and $(n+1)^2$ but not equal to any one of them.
  $\therefore |xy^2z|$ is not a perfect square and so $xy^2z \notin$ L.
  $\therefore$ this is a contradiction. This implies not Regular

Pumping Lemma for Regular Sets

Show that L= $\{a^p \mid$p is a prime$\}$ is not regular.

Solution:

1. Let L is regular. Let 'n' be number of states in finite automata accepting L.
2. Let 'p' be a prime number greater than 'n'.
   Let w=a^p
   by pumping lemma, w=xyz with $|xy| \leq$ n and $|y| > 0$
   x, y, z are simply strings of a's.
   So, y= a^m for some m $\geq$ 1 (and $\leq$ n)
3. Let i= p+1, then
   $|xy^iz|$= |xyz |+ $|y^{i-1}|$=p+(i-1)m=p+pm=p(1+m) which is not prime.
   $\therefore xy^iz \notin$ L. $\implies$ contradiction.
   So, L is not regular.

Pumping Lemma for Regular Sets

1. Show that L=$\{0^i 1^i \mid i \geq 1\}$ is not regular.
2. Show that L= $\{ww \mid w \in \{a, b\}^* \}$ is not regular.
3. Is L = $\{a^{2n} \mid n \geq 1\}$ regular ?

Regular Sets and Regular Grammar

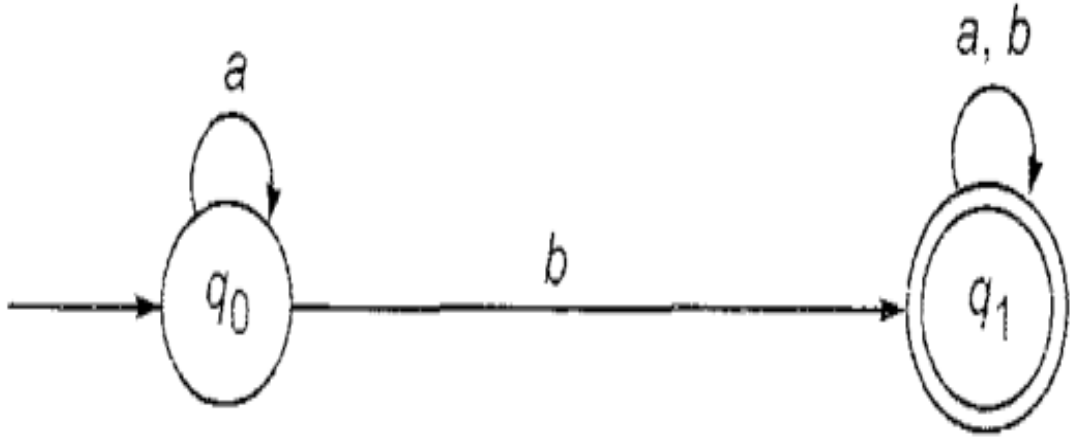
We can construct a Regular Grammar from a Regular Sets and vice versa.

Construction of a Regular Grammar from a Regular Sets

- We can show that $L(G) = T(M)$ by using the construction of P such that:
  $A_i \rightarrow aA_j$ iff $\delta(q_i, a) = q_j$
  $A_i \rightarrow a$ iff $\delta(q_i, a) \in F$
- Construct a regular grammar G generating the regular set represented by
  P= $a^*b(a+b)^*$.

Regular Sets and Regular Grammar

Solution:



Let G=($\{q_0, q_1\}$,{a,b},P,q_0)
where P is given by:

$q_0 \rightarrow aq_0$

$q_0 \rightarrow bq_1,\ q_0 \rightarrow b$

$q_1 \rightarrow aq_1,\ q_1 \rightarrow bq_1$

$q_1 \rightarrow a, q_1 \rightarrow b$

Regular Sets and Regular Grammar

Construction of a Regular Set from a Regular Grammar

- We define M as:
  1. Each production $A_i \to aA_j$ induces a transition from q_i to q_j with label a i.e $\delta(q_i, a) = q_j$,
  2. Each production $A_i \to a$ induces a transition from q_i to q_f with label a i.e $\delta(q_i, a) = q_f \in F$
  3. $S \to \wedge$, corresponding transition is from q_0 to q_f with a label $\wedge$ or q_0 is also a final state.
- Let G=($\{$A,B$\}$,$\{$a,b$\}$,P,A) where P consists of
  A $\to$ aB, B $\to$ bB
  B $\to$ a, B $\to$ bA
  Construct a transition system M accepting L(G).

Regular Sets and Regular Grammar

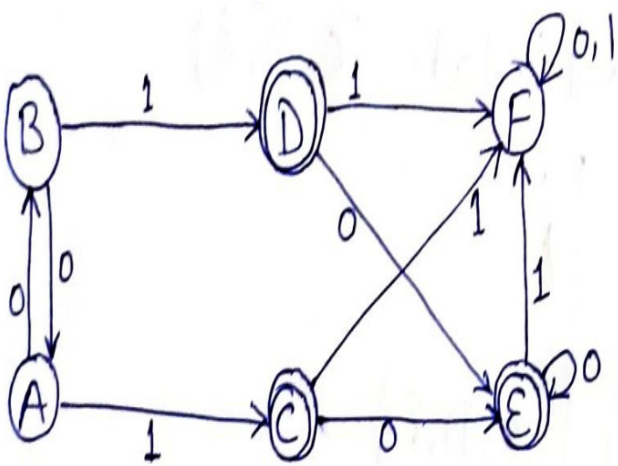
- If a regular grammar G is given by S $\rightarrow$ aS |a. Find M accepting L(G).
- Construct a DFA equivalent to grammar
  S $\rightarrow$ aS |bS |aA
  A $\rightarrow$ bB
  B $\rightarrow$ aC, C $\rightarrow \wedge$

Myhill Nerode Theorem

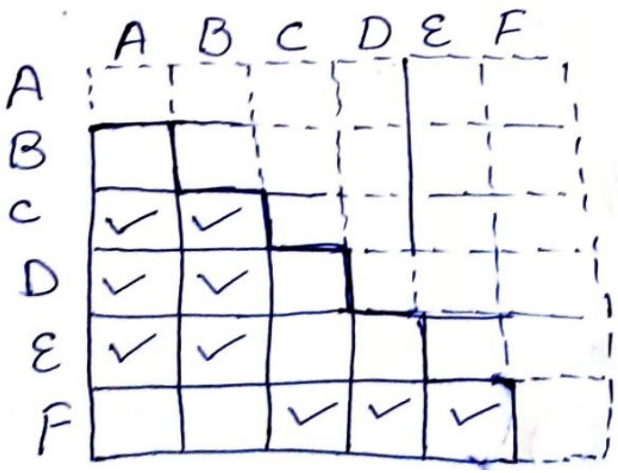
Myhill-Nerode Theorem-states that L can be accepted by an FA if and only if the set of equivalence classes of I_L is finite.
It also states that the number of states in the smallest automaton accepting L is equal to the number of equivalence classes in R_L.

- It is used to prove whether or not a language L is regular
- It is also used for minimization of states in DFA
- Steps:
  1. Draw a table for all unordered pair of states (P,Q).
  2. Mark all pairs where P $\in$ F and Q $\notin$ F.
  3. If there are unmarked pairs (P,Q) such that $[\delta(P, a), \delta(Q, a)]$ is marked, then mark [P,Q], where $a \in \sum$ is an input symbol.
  4. Repeat step 3 untill no more marking can be made.
  5. Combine all un-marked pairs and make them a single state in minimized DFA.
- Minimize the given deterministic finite automata

Myhill Nerode Theorem



Solution:



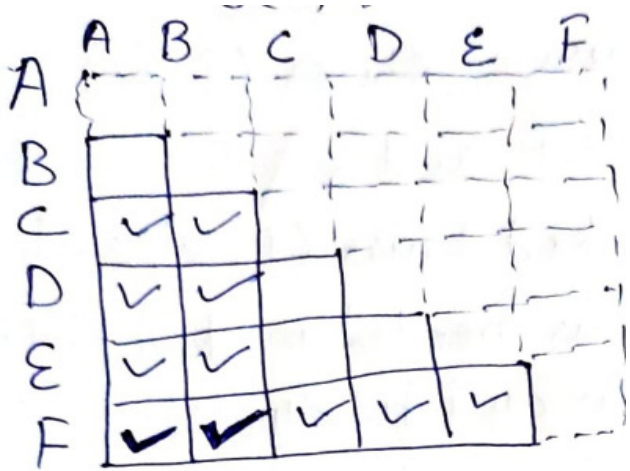
Myhill Nerode Theorem

$$
\begin{aligned}
\text{(B,A)} \implies \delta(B,0) &= \text{A} \\
\delta(A,0) &= \text{B } \textbf{Unmarked} \\
\delta(B,1) &= \text{D} \\
\delta(A,1) &= \text{C } \textbf{unmarked} \\
\text{(D,C)} \implies \delta(D,0) &= \text{E} \\
\delta(D,0) &= \text{E } \textbf{Unmarked} \\
\delta(C,1) &= \text{F} \\
\delta(C,1) &= \text{F } \textbf{unmarked} \\
\text{(E,C)} \implies \delta(E,0) &= \text{E} \\
\delta(C,0) &= \text{E } \textbf{Unmarked} \\
\delta(E,1) &= \text{F} \\
\delta(C,1) &= \text{F } \textbf{unmarked} \\
\text{(E,D)} \implies \delta(E,0) &= \text{E} \\
\delta(E,0) &= \text{E } \textbf{Unmarked} \\
\delta(D,1) &= \text{F}
\end{aligned}
$$

Myhill Nerode Theorem

$$
\begin{aligned}
& & \delta(D,1) &= \text{F } \textbf{unmarked} \\
\text{(F,A)} &\implies & \delta(F,0) &= \text{F} \\
& & \delta(A,0) &= \text{B } \textbf{Unmarked} \\
& & \delta(F,1) &= \text{F} \\
& & \delta(A,1) &= \text{C } \textbf{Marked, Mark (F,A)} \\
\text{(F,B)} &\implies & \delta(F,0) &= \text{F} \\
& & \delta(B,0) &= \text{A } \textbf{Marked, Mark (F,B)} \\
& & \delta(F,1) &= \text{F} \\
& & \delta(B,1) &= \text{D } \textbf{Marked, Mark (F,B)}
\end{aligned}
$$

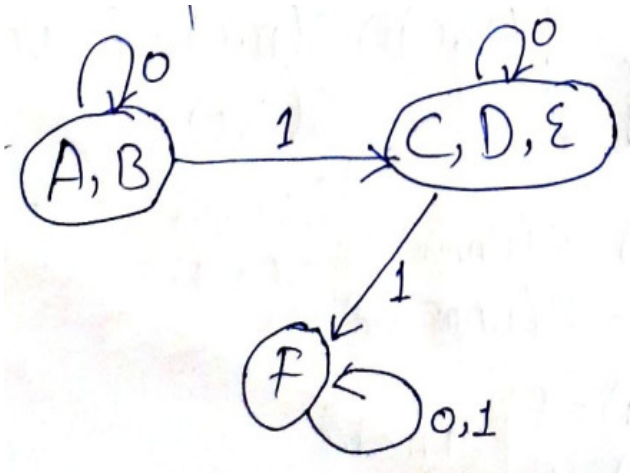
Myhill Nerode Theorem



Therefore, unmarked pairs:
(A,B),(D,C),(E,C),(E,D)
here, (C,D,E) form a common pair. So, combined state (C,D,E)

States	0	1
{A,B}	{A,B}	{C,D,E}
{C,D,E}	{C,D,E}	{F}
{F}	{F}	{F}

Myhill Nerode Theorem



Questions?

+91 9911375598, akyadav@akyadav.in

Thank you