Lecture 5.1: Introduction to Finite Automata

Presenter: Warida Rashid *Scribe: Warida Rashid*

Finite Automata/ Finite State Machine

Finite Automata (FA) or Finite State Machine (FSM) is a mathematical model that recognizes patterns within input taken from some character set (or alphabet).

A Finite Automaton does the following things:

- Stores a finite amount of information
- Given a string of input symbol, it either accepts or rejects it

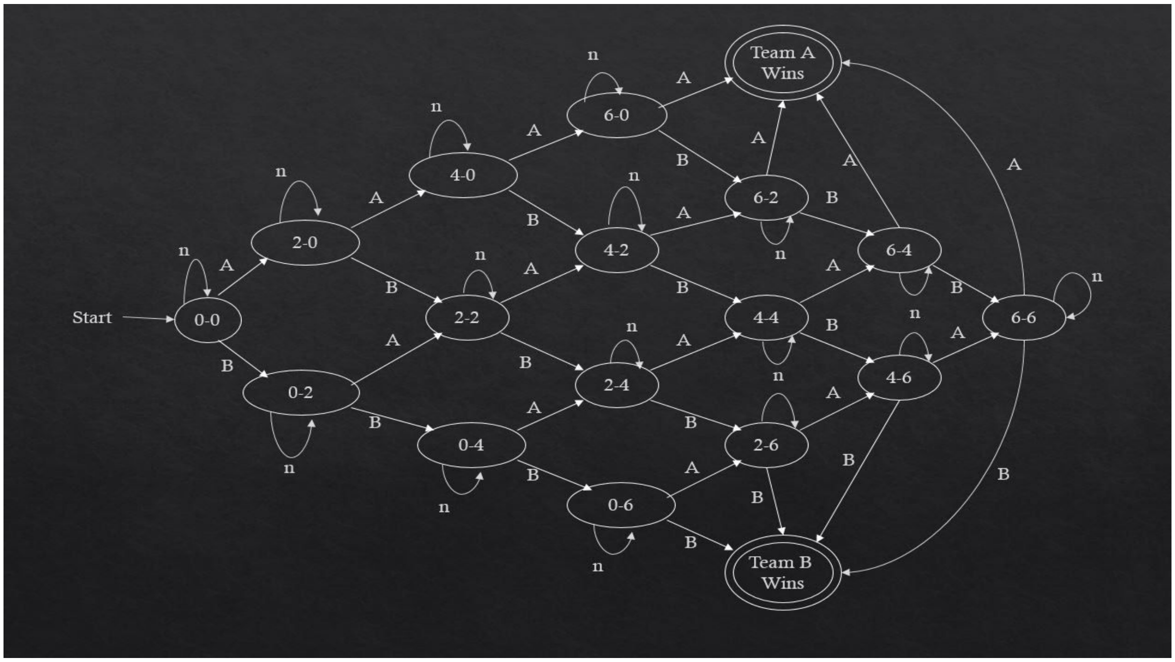
A Finite Automaton consists of:

- A finite number of states to represent a finite amount of information
- A designated start state
- A set of designated final states
- A set of rules that defines how the automaton moves from one state to another in response to input symbols

Example

Let's say we have to design an automaton for a buzzer quiz. The rules of the quiz are:

- The quiz is played between two teams: Team A and Team B.
- Both the teams are asked a question.
- Whichever team presses the buzzer first gets to answer the question.
- If the answer is correct, the team scores 2 points.
- If the answer is wrong, no one scores a point and the quiz moves on to the next question.
- The team that scores 8 points, wins.
- The quiz must have a winner. If a winner is not found, the quiz would be considered to be "Invalid".



Explanation:

1. The automaton is represented as a graph.
2. Each node represents a state.
3. The edges represent transitions from one state to another.
4. The label or name of the states represent the score of team A and team B in (Team A score - Team B score) format. The states store the information about the current scores of the teams.
5. The labels on the edges represent the team that scores a point.
6. There is a transition for each question depending on one of the 3 outcome :
   - Team A scores a point, marked as **A**
   - Team B scores a point, marked as **B**
   - None of the teams score a point, marked as **n**

Transitions

1. The automaton starts at state **0-0**, i.e. the scores of both team A and team B is 0 at the beginning.
2. After the 1st question at state **0-0** the following transitions are made:
   - If team A gets the correct answer, the automaton follows the transition labeled "A" and goes to state **2-0** i.e. the score of team A is 2 and team B is 0.
   - If team B gets the correct answer, the automaton follows the transition labeled "B" and goes to state **0-2** i.e. the score of team A is 0 and team B is 2.
   - If the answer was wrong, the scores are still **0-0** so the automaton follows the transition labeled **n** that loops back on the state **0-0.**

3. From the states **2-0** and **0-2**, considering the 3 outputs for each (as stated in #6 in the explanation part), the automaton has transitions to states **4-0**, **2-2, 2-0** and **2-2, 0-4, 0-2** respectively. All the other transitions are designed in the same way.
4. There are two final states: **Team A Wins** and **Team B Wins.**

Processing an Input Sequence/ String

1. The alphabet of the automaton is the set {A, B, n}
2. An example string: ABnnBAnBAA
3. Following the transitions labeled by the characters of the input string in order, the automaton moves through the sequence of states starting from 0-0:

   2-0, 2-2, 2-2, 2-2, 2-4, 4-4, 4-4, 4-6, 6-6, Team A Wins

4. Final output: After processing an input sequence of characters, the automaton reached the final state **Team A Wins**. Therefore, the input sequence is "valid". It is accepted by the automaton.

CSE331	Automata and Computability

Lecture 5.2: Deterministic Finite Automata

Presenter: Warida Rashid *Scribe: Warida Rashid*

Classification of Finite Automata

There are two types of finite automata:

1. Deterministic Finite Automata (DFA)
2. Non Deterministic Finite Automata

Deterministic Finite Automata (DFA)

A DFA is an automaton where for each input symbol, there is exactly one transition. A DFA can be represented by a 5-tuple (Q, Σ, δ, q_0, F) where"

- Q is a finite set of states
- Σ is is a finite set of symbols called the alphabet
- δ is the transition function defined as $\delta : (Q \times \Sigma) \rightarrow Q$
- q_0 is the initial/start state from where any input is processed and $q_0 \in Q$
- F is a set of final state(s) and $F \subseteq Q$

Transition Function, δ

The transition function takes in two parameters:

1. A state, $q_{current} \in Q$
2. An input symbol $a \in \Sigma$

The output of a transition function is exactly one state, $q_{next} \in Q$

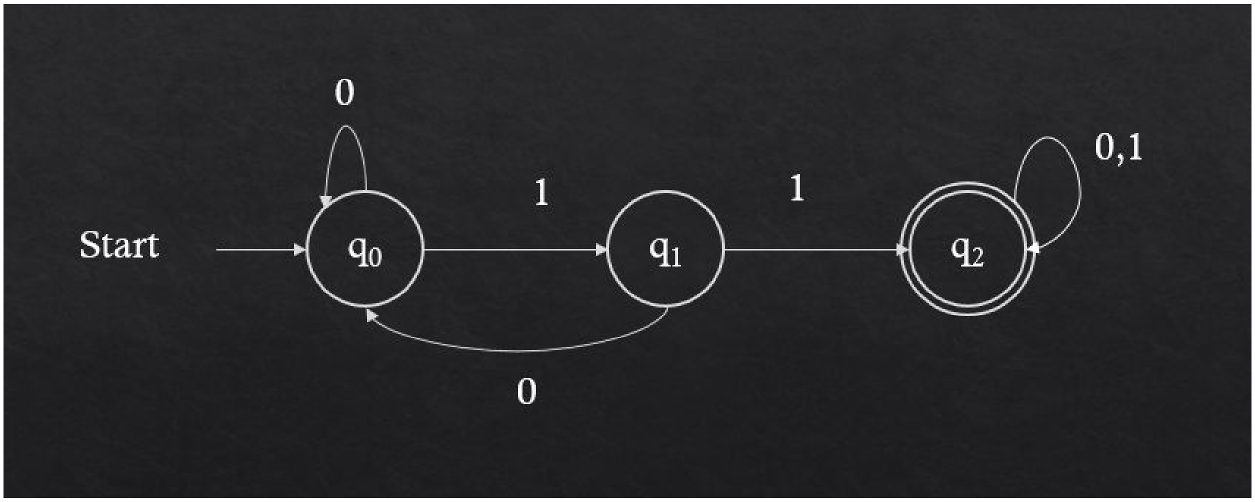
Given a state and an input symbol, a transition function dictates the next state of the automaton.

Example

A DFA that accepts binary strings with two consecutive 1s or binary strings that contain the substring 11.

Graph Representation

This is the graph representation of the DFA. In this representations, the start state is marked by an arrow and the final state(s) is marked with a double circle.



In this DFA:

- Q is $\{q_0, q_1, q_2\}$
- Σ is {1, 0} since it works with binary strings
- q_0 is the initial/start state
- F is a $\{q_2\}$

The transition function for this DFA yeilds:

- $\delta\ (q_0, 0) \rightarrow q_0$
- $\delta\ (q_0, 1) \rightarrow q_1$
- $\delta\ (q_1, 0) \rightarrow q_0$
- $\delta\ (q_1, 1) \rightarrow q_2$
- $\delta\ (q_2, 0) \rightarrow q_2$
- $\delta\ (q_2, 1) \rightarrow q_2$

Transition Table

A DFA can also be represented as a table where the rows contain the states and the columns contain the input symbols. In a transition table, the start state is marked by an arrow and the final state(s) is marked by an asterisk . For the given DFA, the transition table would be:

	0	**1**
$\rightarrow q_0$	q_0	q_1
q_1	q_0	q_2
$*q_2$	q_2	q_2

Lecture 5.3: Deterministic Finite Automata

Presenter: Warida Rashid *Scribe: Warida Rashid*

Extended Transition Function

The extended transition function takes a state q and an input string w and yields the resulting state in which the processing of the string ends. The definition proceeds by induction over the length of w. The definition precedes by induction over length of w. It is sometimes represented as $\hat{\delta}$ (delta hat) to distinguish it from the transition function.

Induction basis: When w is of length 0 or an empty string (ε), it is defined as δ (q, ε) = q

Inductive Step: In this step, we determine the transition of a string (w) of length ***l+1*** from a string of length ***l***. Let's say, w is of the form "va" where v is a string of length *l* and "*a*" is a symbol. Therefore, δ (q, va) = δ (δ (q, v), a)

Example

Let's consider the DFA that accepts strings with two consecutive 1s and we have to determine δ (q_0, 110).

Transition table:

	0	**1**
→q_0	q_0	q_1
q_1	q_0	q_2
***q_2**	q_2	q_2

δ (q_0, 110)

= δ (δ (q0, 11), 0)

= δ (δ (δ (q0, 1),1), 0)

= δ (δ (δ (δ (q0, ε), 1),1), 0)

= δ (δ (δ (q0, 1),1), 0) ***[The basis of the induction]***

= δ (δ (q1,1), 0)

= δ (q2, 0)

= q2

Language of a DFA in terms of the Extended Transition Function

The language of a DFA (Q, Σ, δ, q_0, F) is a set that contains strings w such that δ (q0, w) $\in$ F

Lecture 6.1 and 6.2: DFA Examples

Presenter: Warida Rashid *Scribe: Warida Rashid*

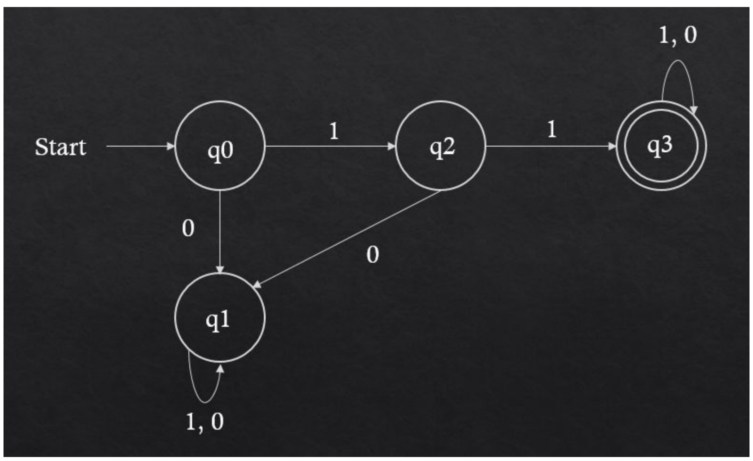
1. **L(A) = {w | w ∈ (0,1)* and w starts with 11} i.e {11, 110, 111, 1100, 1101............}**

   q0: Start state. Seen nothing.

   q1: First symbol was 0. It's a **trap** state. Once the DFA is in this state, the string has already violated the condition for the language and therefore is rejected regardless of the next symbols.

   q2: Last symbol seen was 1. It was also the 1st symbol of the string.

   q3: Last symbol was 1 and it was the second symbol of the string. Therefore, the string starts with 11 and q3 is the final state.

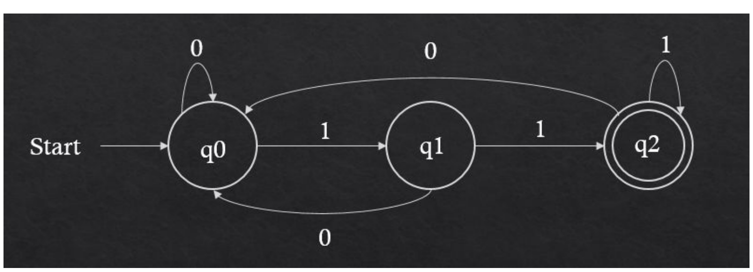


2. **L(A) = {w | w ∈ (0,1)* and w ends with 11} i.e. {11, 011, 111............}**

   q0: Start state. Seen nothing or the last symbol was 0.

   q1: Last symbol seen was 1. Seen the 1st *1* of the substring *11.*

   q2: Last symbol seen was 1 which is the 2nd *1* of the substring *11.* If the string ends after seeing 11, it should be accepted. Therefore, q2 is the final state.

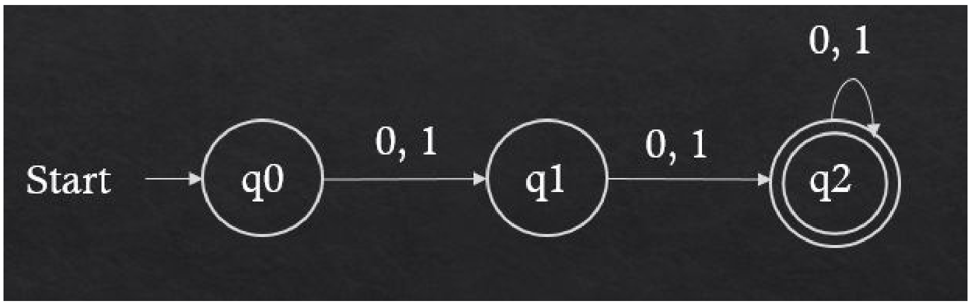


3. **L(A) = {w | w ∈ (0,1)* and |w| ≥ 2} i.e. length of the string is at least 2**

   q0: Start state. Seen nothing.

   q1: Seen 1 symbol (0 or 1), length 1.

   q2: Seen 2 symbols (Any combination of 0s and 1s), length 2. This is the final state.



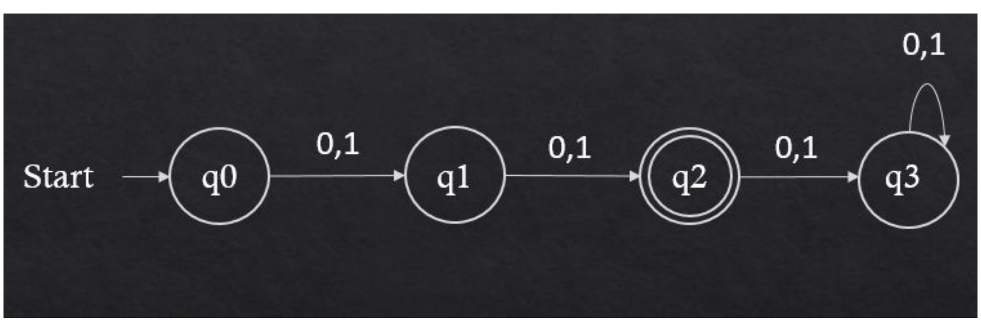
4. **L(A) = {w | w ∈ (0,1)* and |w| = 2} length of the string is 2**

   q0: Start state. Seen nothing.

   q1: Seen 1 symbol (0 or 1), length 1.

   q2: Seen 2 symbols (Any combination of 0s and 1s), length 2. This is the final state.

   q3: Seen 3 symbols (Any combination of 0s and 1s), length 3. This is the **trap state** since the condition is violated.



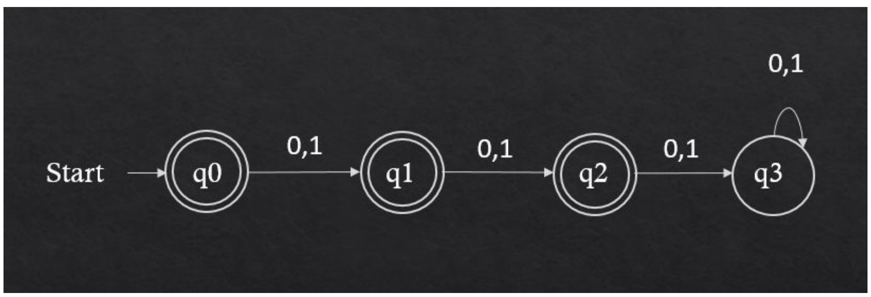
5. **L(A) = {w | w ∈ (0,1)* and |w| ≤ 2} i.e. length of the string is at most 2**

q0: Start state. Seen nothing and it is one of the final states.

q1: Seen 1 symbol (0 or 1), length 1 and it is one of the final states.

q2: Seen 2 symbols (Any combination of 0s and 1s), length 2 and it is one of the final states.

q3: Seen 3 symbols (Any combination of 0s and 1s), length 3. This is the **trap state** since the condition is violated.

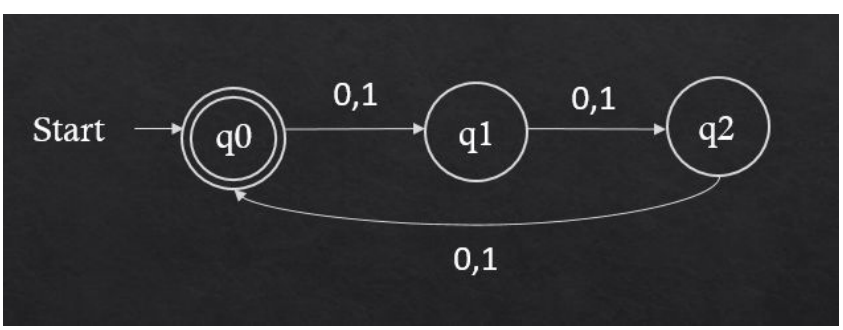


6. **L(A) = {w | w ∈ (0,1)* and the length of w is divisible by 3} i.e. w of length 0, 3, 6, 9, 12, 15**

q0: Start state and the remainder is 0 when the length is divided by 3 i.e. the length is divisible by 3. Therefore, it is also the final state

q1: The remainder is 1 when the length is divided by 3.

q2: The remainder is 2 when the length is divided by 3.



7. **L(A) = {w | w ∈ (0,1)* and w as a binary integer is divisible by 5}**

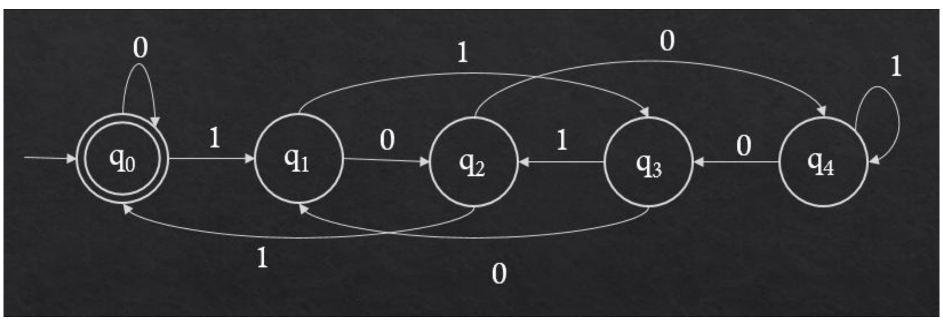
q0: The string processed so far represents an integer of the form 5m where m is any integer. Since numbers of this form are divisible by 5, it is the final state.

q1: The string processed so far represents an integer of the form 5m+1 where m is any integer i.e. the remainder is 1 when the number is divided by 5.

q2: The string processed so far represents an integer of the form 5m+2 where m is any integer i.e. the remainder is 2 when the number is divided by 5.

q3: The string processed so far represents an integer of the form 5m+3 where m is any integer i.e. the remainder is 3 when the number is divided by 5.

q4: The string processed so far represents an integer of the form 5m+4 where m is any integer i.e. the remainder is 4 when the number is divided by 5.



CSE331 Automata and Computability

Lecture 6.3: DFA Cross Product Operation

Presenter: Warida Rashid *Scribe: Warida Rashid*

The cross product is the process of constructing a DFA that simulates the steps of two different DFAs in parallel.

Let the two DFAs be M1 and M2 accepting regular languages L1 and L2

1. $A_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$
2. $A_2 = (Q_2, \Sigma, \delta_1, q_0^2, F_2)$

From these two DFAs, we can construct a DFA, **$A = (Q, \Sigma, \delta, q_0, F)$** that recognizes

1. $A_1 \cup A_2$
2. $A_1 \cap A_2$
3. $A_1 - A_2$

For the DFA, **$A = (Q, \Sigma, \delta, q_0, F)$:**

Set of states

Q is the set of states. It contains pairs of states from A_1 and A_2

$$Q = Q_1 \text{ X } Q_2 = \{(q_1, q_2) \mid q_1 \in Q_1 \text{ and } q_2 \in Q_2\}$$

Start State

$q_0 = (q_0^1, q_0^2)$

Transition Function

$\delta((q_1, q_2), a) = (\delta_1(q_1, a), \delta_2(q_2, a))$ where q_1 and q_2 are states of A_1 and A_2 respectively and (q_1, q_2) represents the state of the new DFA, A that is derived from A_1 and A_2

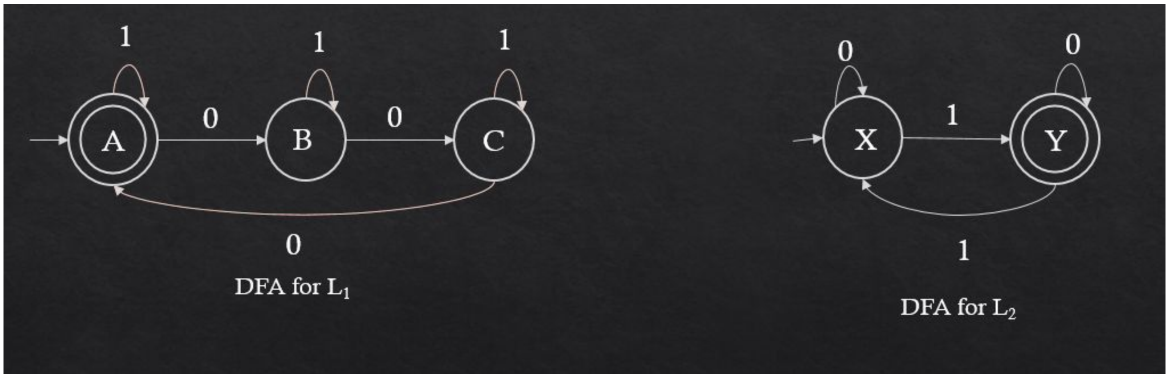
Set of Final States

1. For $A = A_1 \cup A_2$, $F = \{(q_1, q_2) \mid q_1 \in F_1$ **OR** $q_2 \in F_2\}$, it accepts when either A_1 or A_2 accepts
2. For $A = A_1 \cap A_2$, $F = \{(q_1, q_2) \mid q_1 \in F_1$ **AND** $q_2 \in F_2\}$, it accepts when both A_1 and A_2 accepts
3. or $A = A_1 - A_2$, $F = \{(q_1, q_2) \mid q_1 \in F_1$ **AND** $q_2 \notin F_2\}$ it accepts when A_1 accepts and A_2 rejects

Example

Let's consider the DFAs for the following languages:

1. L1 = {w | w ∈ (0,1)* and n0(w) is divisible by 3} i.e. the number of 0 is divisible by 3
2. L2 = {w | w ∈ (0,1)* and n1(w) is odd} i.e. the number of 1 is odd



Here, the **set of states** for the first DFA, $\mathbf{Q_1}$ **= {A, B, C}** the second DFA, $\mathbf{Q_2}$ **= {X, Y}**

Start state of the first DFA is **A** and the second DFA is **X**

Set of final state of the first DFA is $\mathbf{F_1}$ **= {A}** and the second DFA is $\mathbf{F_2}$ **= {Y}**

Therefore for the new DFA:

The set of states is, **Q =** $\mathbf{Q_1}$ **X** $\mathbf{Q_2}$ **= {(A, X), (A, Y), (B, X), (B, Y), (C, X), (C, Y) }**

For simplicity, let's name the states as AX, AY, BX, BY, CX, CY. So, **Q = {AX, AY, BX, BY, CX, CY}**

The start state is the state that corresponds to **(A, X)**, which is **AX**

The set of final states:

1. If **L = L1 ∪ L2** = {w | w ∈ (0,1)* and the $n_0(w)$ is divisible by 3 **OR** $n_1(w)$ is odd}

   F = {AX, AY, BY, CY}
2. If **L = L1 ∩ L2** = {w | w ∈ (0,1)* and the $n_0(w)$ is divisible by 3 **AND** $n_1(w)$ is odd},

   F = {AY}
3. **L = L1 - L2** = {w | w ∈ (0,1)* and the $n_0(w)$ is divisible by 3 **AND** $n_1(w)$ is **NOT** odd},

   F = {AX}

The Transitions:

We can figure out the transitions for the new DFA from the transitions of the two DFAs it simulates. It is easier to see from the transition table.

	0	1
→ * A	B	A
B	C	B
C	A	C

Image: Transition table for L_1

	0	1
→ X	X	Y
* Y	Y	X

Image: Transition table for L_2

δ (A, 0) = B and δ (X, 0) = X
So, δ (AX, 0) = BX

δ (B, 0) = C and δ (X, 0) = X
So, δ (BX, 0) = CX

δ (C, 0) = A and δ (X, 0) = X
So, δ (CX, 0) = AX

δ (A, 0) = B and δ (Y, 0) = Y
So, δ (AY, 0) = BY

δ (B, 0) = C and δ (Y, 0) = Y
So, δ (BY, 0) = CY

δ (C, 0) = A and δ (Y, 0) = Y
So, δ (CY, 0) = AY

δ (A, 1) = A and δ (X, 1) = Y
So, δ (AX, 1) = AY

δ (B, 1) = B and δ (X, 1) = Y
So, δ (BX, 1) = BY

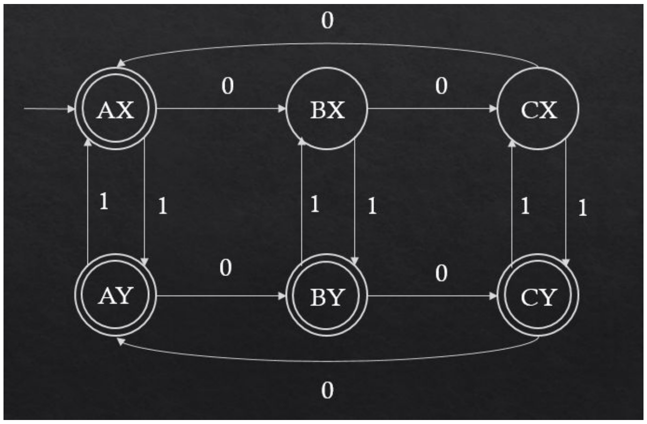
δ (C, 1) = C and δ (X,1) = Y
So, δ (CX, 1) = CY

δ (A, 1) = A and δ (Y, 1) = X
So, δ (AY, 1) = AX

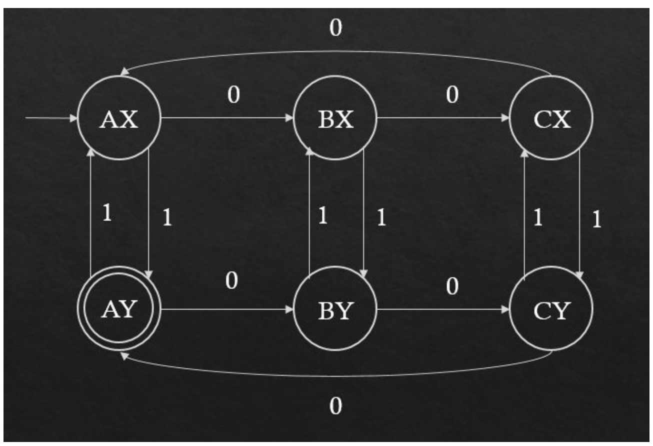
δ (B, 1) = B and δ (Y, 1) = X
So, δ (BY, 1) = BX

δ (C, 1) = C and δ (Y, 1) = X
So, δ (CY, 1) =CX

The DFA for L = L1 ∪ L2 = {w | w ∈ (0,1)* and the $n_0(w)$ is divisible by 3 **OR** $n_1(w)$ is odd}:



The DFA for L = L1 ∩ L2 = {w | w ∈ (0,1)* and the $n_0(w)$ is divisible by 3 **AND** $n_1(w)$ is odd}:



The DFA for L = L1 - L2 = {w | w ∈ (0,1)* and the $n_0(w)$ is divisible by 3 **AND** $n_1(w)$ is **NOT** odd}: