Finite State Automata

Prsented by - JPK

Finite State Automata – FSA Model

a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	………

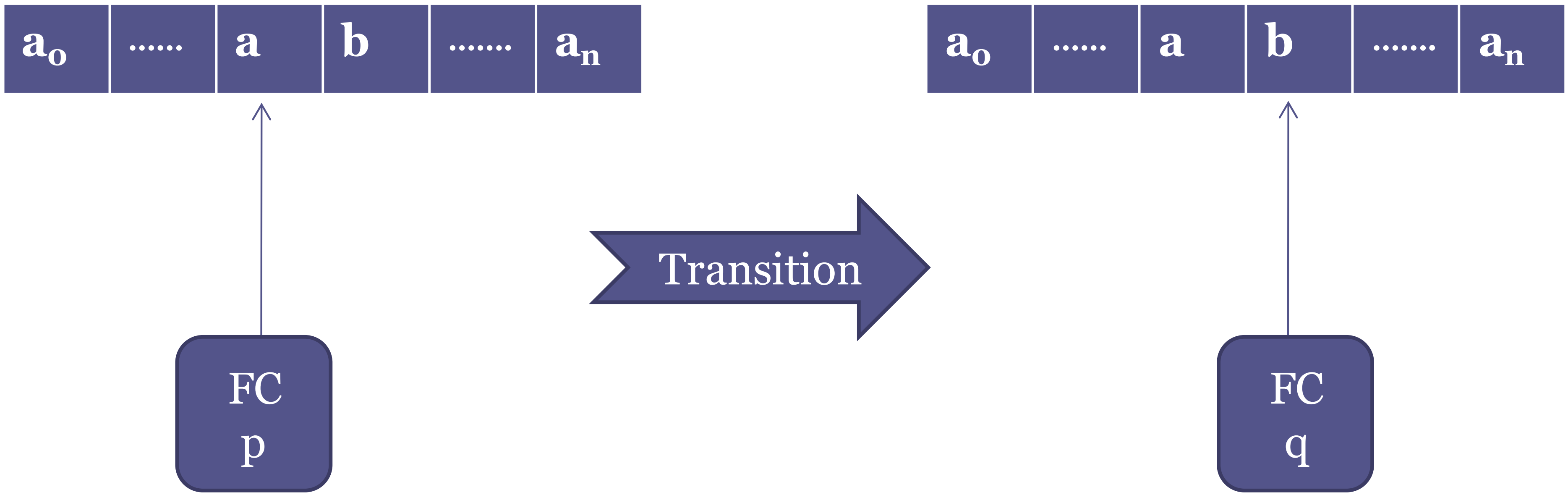
Input Tape
It contains input symbols $a_0, a_1, a_2 \ldots.$
Where a_i's are input symbols

Tape Head
It is used to read input symbol

FC
q

Finite Control
It is **processing element** of FSA.
FC reads input symbol under tape head and depending upon current state moves to some next state and tape pointer is advanced to next symbol.

Finite State Automata Transition



FSA Transition

Suppose current state of FC is p and input symbol read is a then FC moves to next state q.

Symbolic Notation $\delta(p,a)$ is q

FSA - Types

- 1. Deterministic Finite State Automata – DFA
- 2. Non-Deterministic Finite State Automata – NFA
- NFA is easy to design than DFA
- **NFA and DFA are equivalent**
- It means both NFA and DFA have same computing power.
- FSA is **abstract model of computing.**
- It accepts **regular language.**

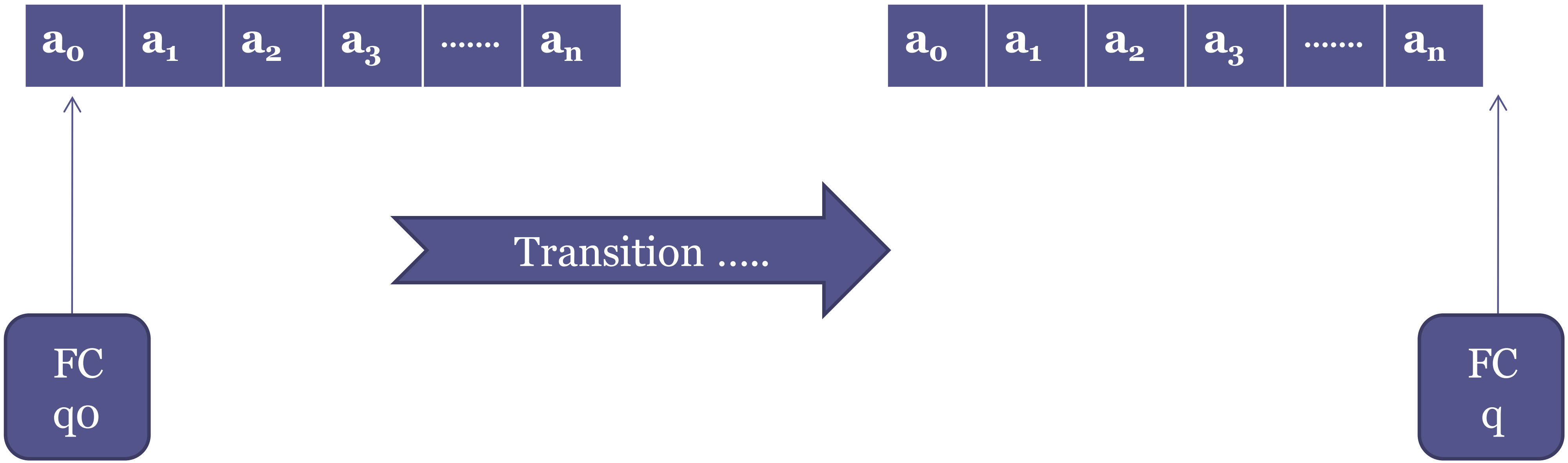
DFA – Formal Definition

- A DFA M is a **5 tuple.**
- **$M=(Q,\sum,\delta,q_o,F)$**
- Where
- Q – is a finite set of states
- $\sum$ – is a finite set of input alphabets
- q_o- initial state. It belongs to set Q. **(Unique initial state)**
- F – set of final states/ accept states. It is subset of Q. **(Multiple final states)**

- δ – It is **transition function or mapping from set Q and $\sum$ to Q.** It maps a given state p and input symbol a to some next state q. It can be specified using a **state transition function, table or diagram.**
- **Symbolically $\delta : Q \times \sum \text{->} Q$**

- **Note – In DFA we have a unique transition of finite control for each and every state and input combination.**

DFA Working



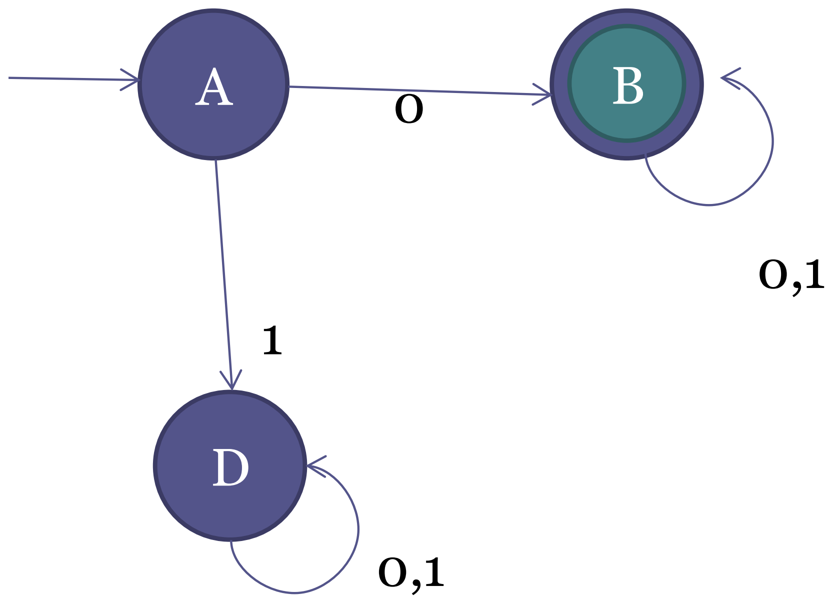
DFA working

DFA M reads current input symbol and depending upon current state makes a transition to next state. This process continues until all the input symbols are read by DFA and ultimately it reaches to some state q.

If q belongs to final/accept state we say that input string w is accepted Otherwise input string w is rejected by DFA.

DFA Examples

State Transition Diagram



State Transition Function

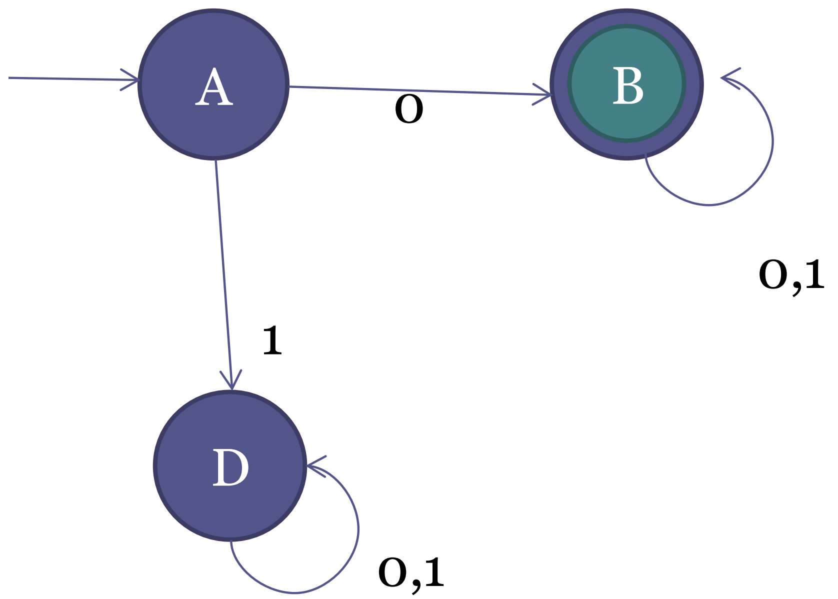
$\delta(A,0)=B$
$\delta(A,1)=D$
$\delta(B,0)=B$
$\delta(B,1)=B$
$\delta(D,0)=D$
$\delta(D,1)=D$

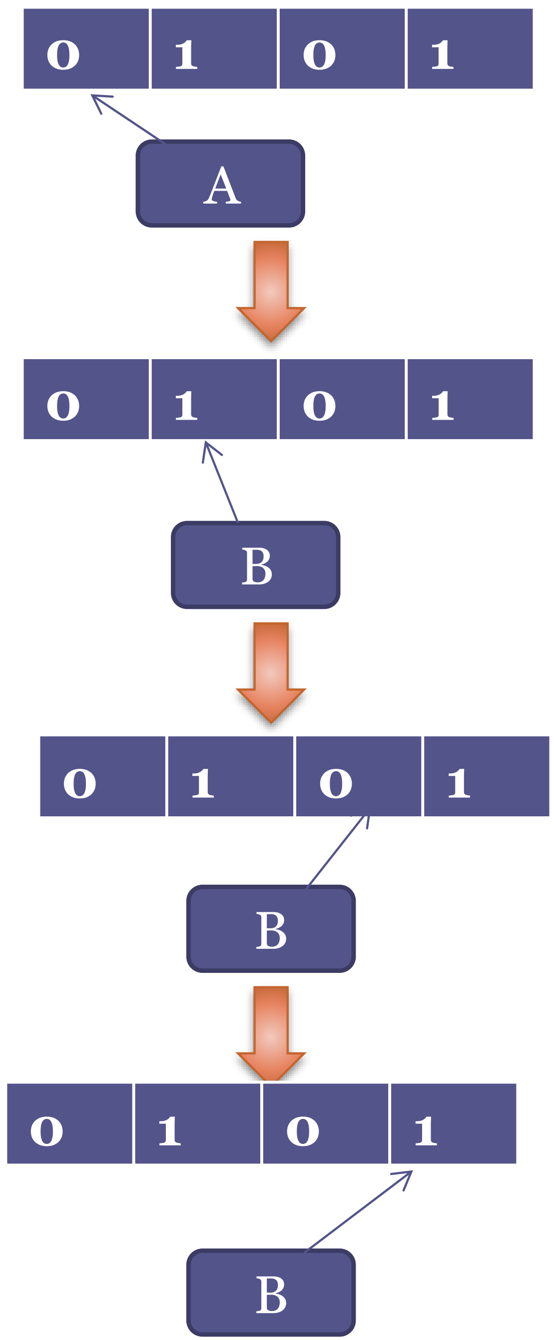
State Transition Table

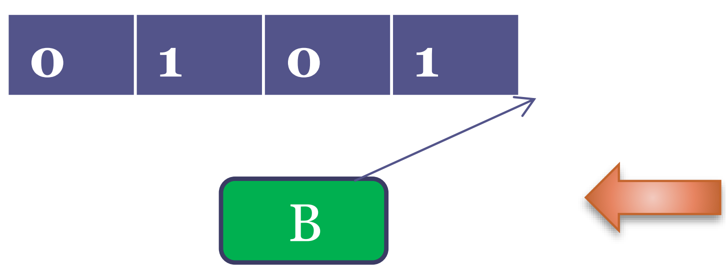
δ	0	1
A	B	D
B	B	B
D	D	D

DFA Working – Input 0101

State Transition Diagram

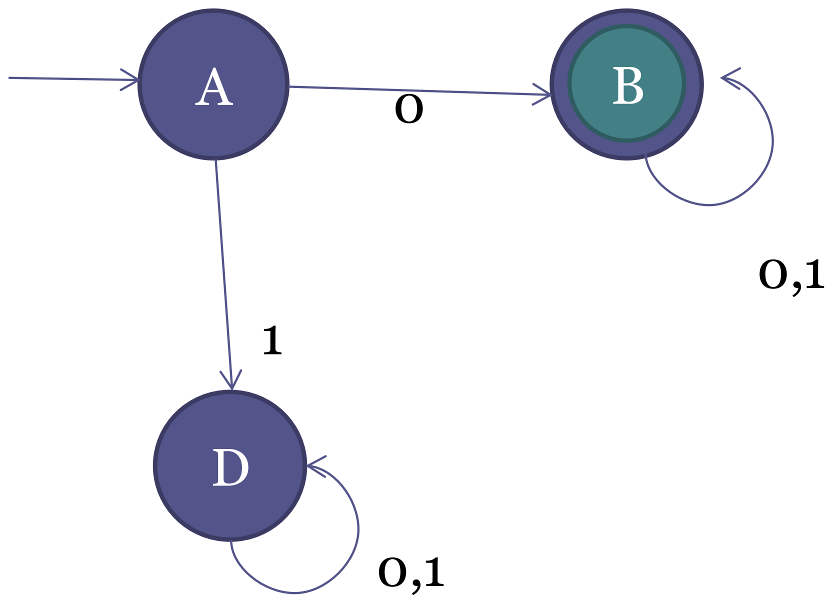


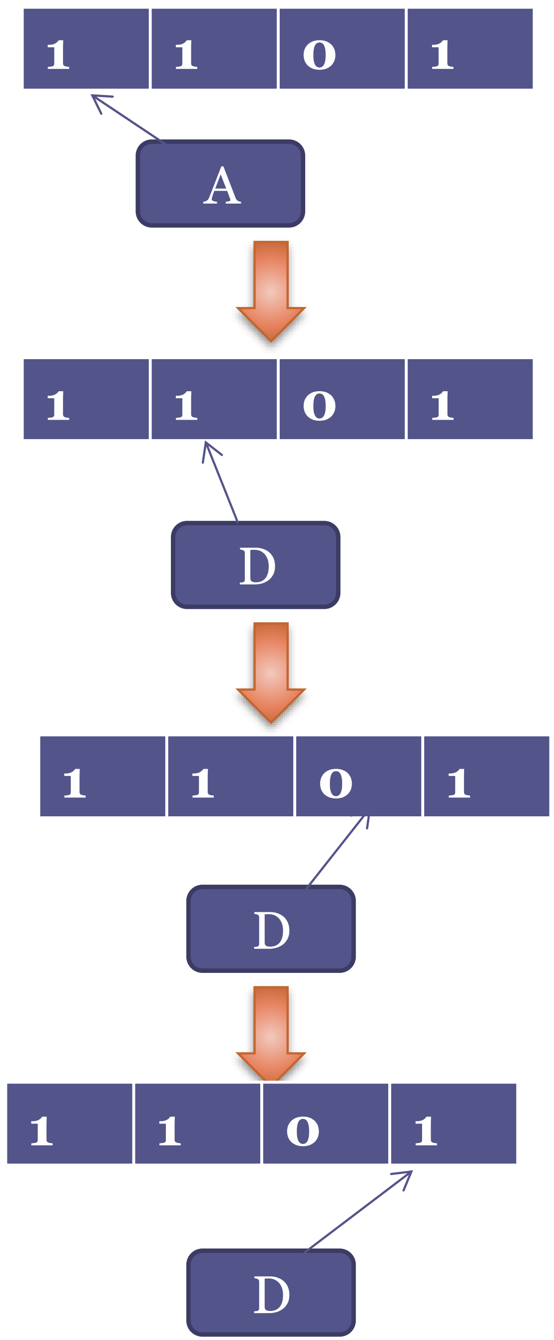


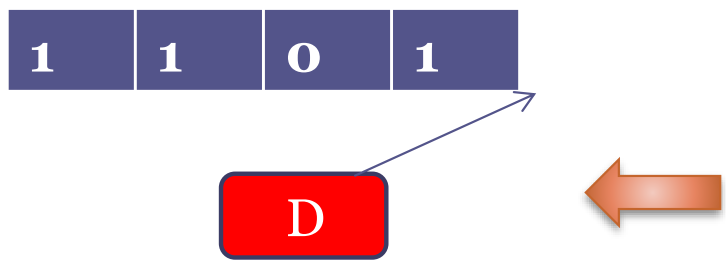


DFA Working – Input 1101

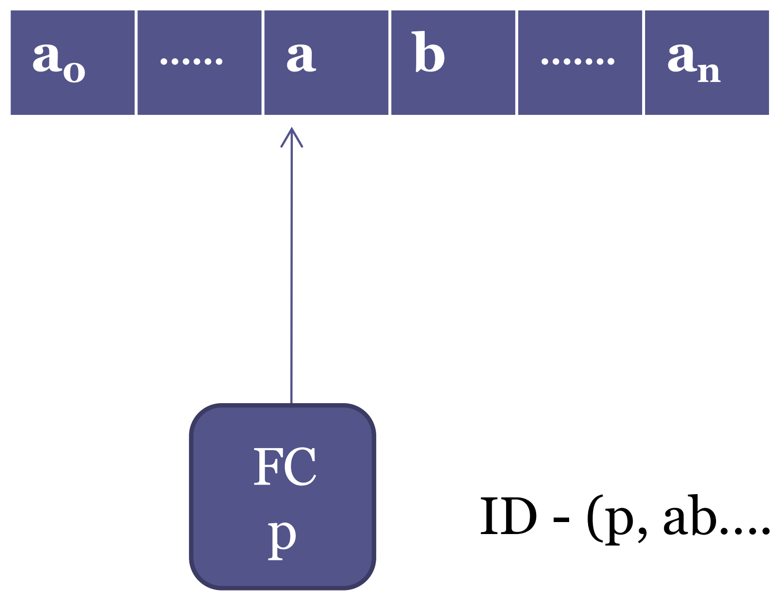
State Transition Diagram







Instantaneous Description of DFA



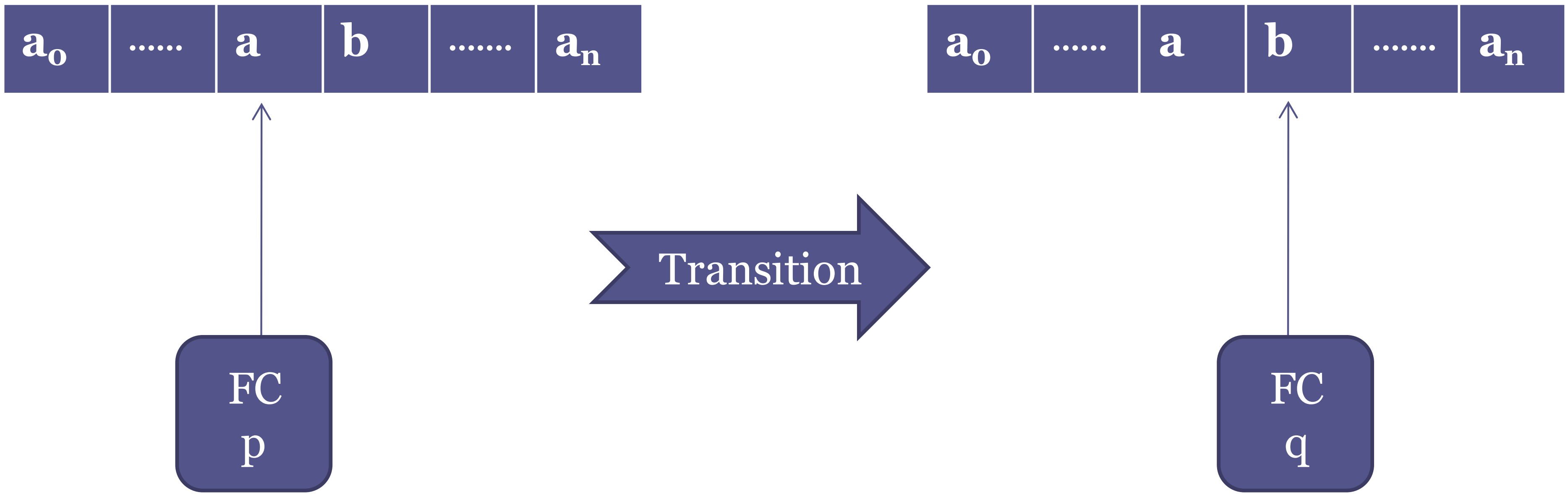
ID - (p, ab….a_n)

Instantaneous Description of DFA
ID is a **snapshot** of a DFA M. It specifies up to what point computation has progressed.

ID (q, abcde…) of a DFA M contains two things

1. Current state
2. Portion of input string left to be read by DFA

Entails |- relationship of ID



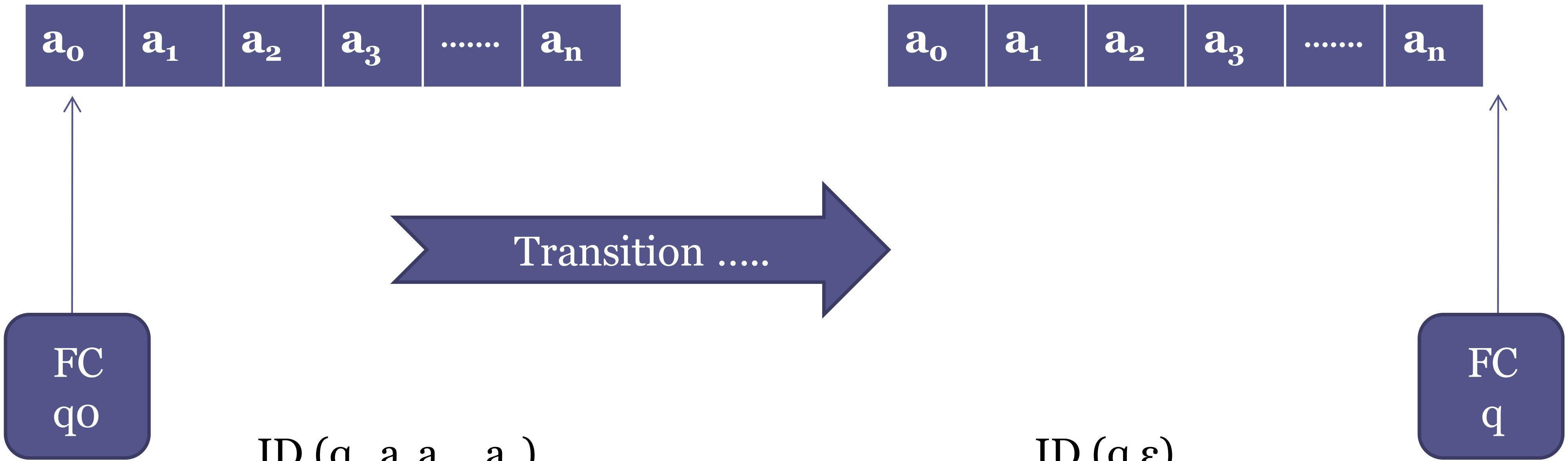
ID - (p, ab….a_n)

ID - (q, b….a_n)

(p,ab…..a_n) |- (q,b…..a_n)
as we have transition
$\delta(p, a)=q$

|- means entails in one step

Acceptance of String w



ID $(q_0, a_0 a_1 \ldots a_n)$

ID (q, ε)

A string w is said to be accepted by DFA M if DFA M reaches a final state after reading input string w one symbol at a time.

Symbolically

$(q_0, w) \vdash^* (q, \varepsilon)$ where $q \in F$

Where $\vdash^*$ is reflexive transitive closure of $\vdash$

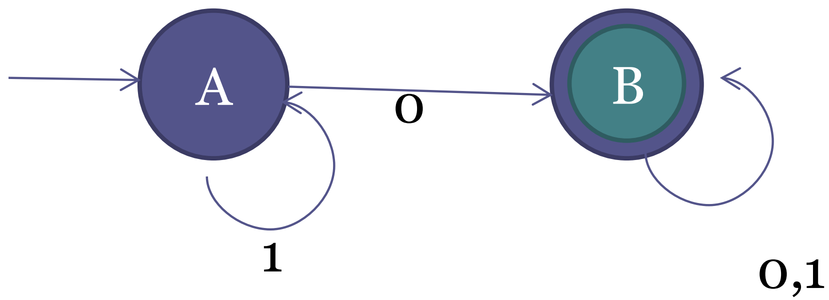
And it means entails in 0 or more steps

Language accepted by DFA M

- The language accepted by DFA M is defined as set of all strings w belonging to $\sum^*$ such that $(q_o,w)|\!-^*(q, \varepsilon)$ and q ε F (final/accept state)
- **$L(M)=\{w \varepsilon \sum^* \mid (q_o,w)|\!-^*(q, \varepsilon) \Lambda q \varepsilon F\}$**

DFA Examples II

State Transition Diagram



State Transition Function

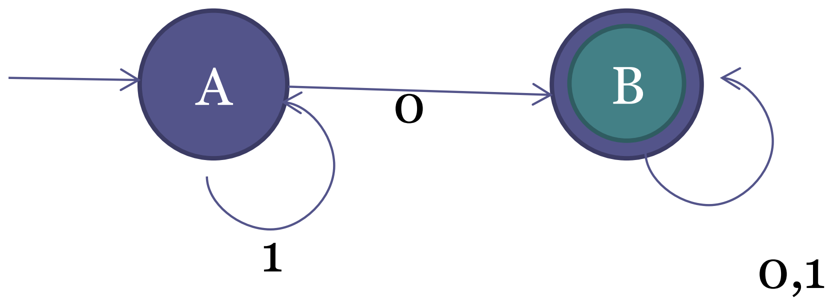
$\delta(A,0)=B$
$\delta(A,1)=A$
$\delta(B,0)=B$
$\delta(B,1)=B$

State Transition Table

δ	0	1
A	B	A
B	B	B

DFA Examples II

State Transition Diagram



Acceptance of Strings

1 . 1101

(A,1101) |- (A,101) |- (A, 01) |- (B,1) |- (B, ε) accept

2. 111

(A,111) |- (A,11) |- (A, 1) |- (A, ε) reject

3. 010

(A,010) |- (B,10) |- (B, 0) |- (B, ε) accept

L(M)={w ε {0,1}*| w contains symbol '0'}