<u>Compiler Construction (CS-636)</u>

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

Note:

It is intimated that following Lectures will not be repeated and would be part of mid-term & final exam as well.

(Week 5) Lecture 9 & 10

Objectives: Learning objectives of this lecture are

- ***Students will able to understand the Deterministic Finite State Automata (DFA) for the regular languages.***
    - ***What is Finite State Automata or DFA***
    - ***Why to Study Finite State Automata***
    - ***Formal Definition of FSA***
    - ***Notations used for FSA***
    - ***Simple Examples***
- ***Similarly, students will able to understand the Non- Deterministic Finite Automata (NFA) for the regular Languages.***
    - ***Meaning of NFA***
    - ***Choosing where to go***
    - ***The ability to make choices***
    - ***How to run an NFA***
    - ***Examples***
    - ***Language of NFA***
    - ***ε- transitions***

Text Book & Resources:

1. Compilers Principles Techniques and Tools (2nd Edition) by Alfread V. Aho, Ravi Sethi.
2. Introduction to Computer Theroy By Daniel I.A. Cohen.

Video Links:

https://youtu.be/8IhMeSx9Eqk (Part 1)
https://youtu.be/6Gq0OpT3LH0 (Part 2)
https://youtu.be/sjpjYcuIFFY (Part 3)
https://youtu.be/JtSSi6n9TaE (Part 4)
https://youtu.be/UbygOXVAJzA (Part 5)

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

What is Finite State Automata(FSA)

- It is a graph like structure also known as directed graph.
- Finite state automata (FSAs) sound complicated, but the basic idea is as simple as drawing a map.
- A finite-state automaton is a device that can be in one of a finite number of *states*.
- In certain conditions, it can switch to another state.
    - This is called a *transition*.
- When the automaton starts working (when it is switched on), it can be in one of its initial states.
- There is also another important subset of states of the automaton: the final states.
    - If the automaton is in a final state when it stops working, it is said to *accept* its input. The input is a sequence of symbols.
- The interpretation of the symbols depends on the application; they usually represent events, or can be interpreted as ``the event that a particular data became available''.
- The symbols must come from a finite set of symbols, called the *alphabet*.
- If a particular symbol in a particular state triggers a transition from that state to another one, that transition is *labeled* with that symbol.
    - The labels of transitions can contain one particular symbol that is not in the alphabet.
    - A transition is labeled with ε (not present in the alphabet) if it can be traversed with no input symbol.
- It is convenient to present automata as directed graphs.
- The vertices denote states.
- They are portrayed as small circles.
- The transitions form the edges - arcs with arrows pointing from the source state (the state where the transition originates) to the target state.
- They are labeled with symbols. Unless it is clear from the context, the initial states have short arrows that point to them from ``nowhere''.
- The final states are represented as two concentric circles.

Why to Study Finite State Automata

- FSA is machine to model the regular languages.
- So, to model the problem belonging to regular language needs finite state automata

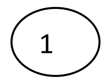
Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid **email id: naseer@biit.edu.pk Whatsapp# 0346-5100010**

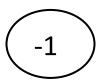
Formal Definition of FSA

- A finite automaton (DFA) is a 5-tuple (Q, $\sum$, δ, q_0, F) where
    - Q is a finite set of states
    - $\sum$ is an alphabet
    - δ: $Q \times \sum \rightarrow Q$ is a transition function
    - $q_0 \in Q$ is the initial state
    - $F \subseteq Q$ is a set of accepting states (or final states).

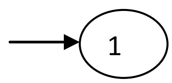
Notations Used to build FSA

- Sates are represented by circles having name inside it. 

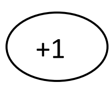
-
- Initial Sate is represented by circle having negative sign inside it or arrow symbol like



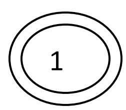
Or



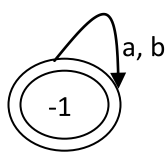
- Transitions having label from $\sum$
    - Transitions omits from states are called outgoing transitions for that particular state.
    - Transitions comes to states are called incoming transitions.
- Final state or acceptance state is represented with double circle or positive sign inside.



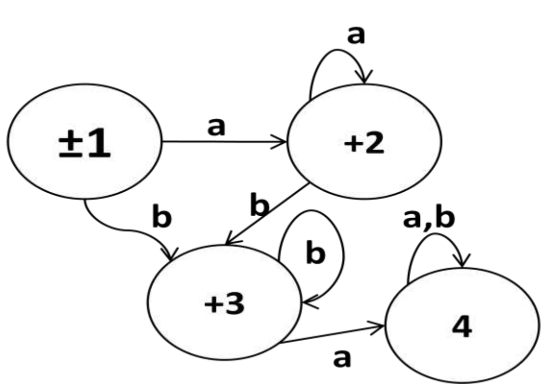
Or



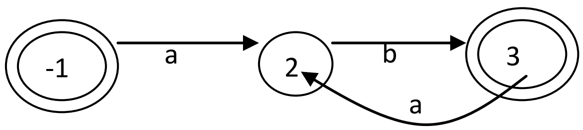
Example #1: FSA for $\sum$* over $\sum$ = {a b}



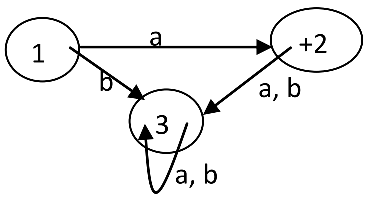
Example #2: FSA for a*b*



Example #3: FSA to accept (ab)*

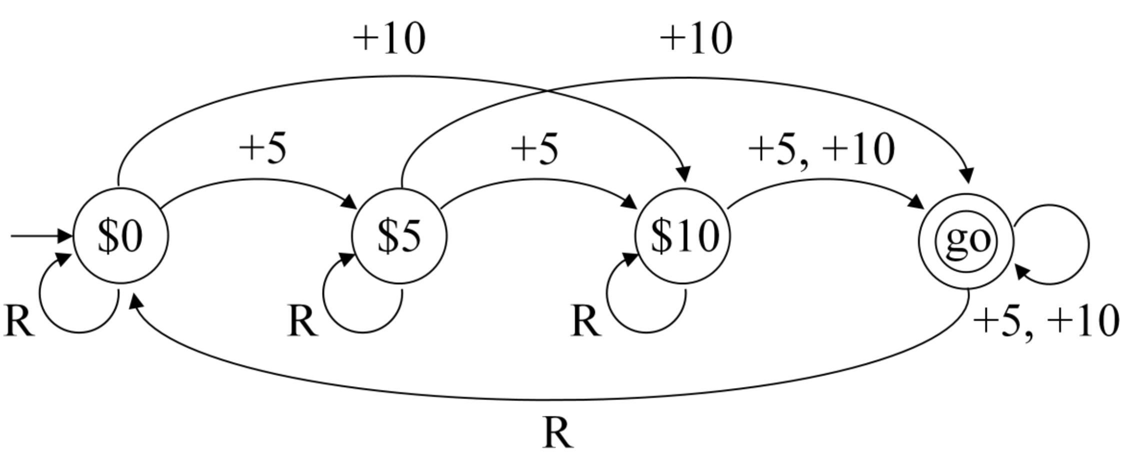


Example #4: FSA to accept a



Example #5: Designing Game ball for following conditions

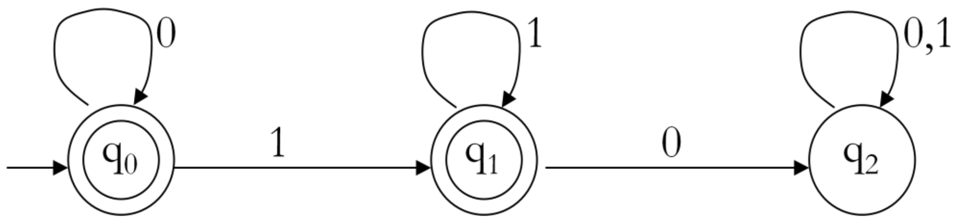
- **There are states \$0, \$5, \$10, go, the start state is \$0**
- **The automaton takes inputs from {+5, +10, R}**
- **The state go is an accepting state**
    - **Game is allowed to play if user puts \$15**
    - **User will not be allowed to play if he puts less than \$15.**



In diagrams, the accepting states will be denoted by double loops

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

Example #6: 0*1*



- alphabet S = {0, 1}
- states $Q = \{q_0, q_1, q_2\}$
- initial state q_0
- accepting states $F = \{q_0, q_1\}$

Transition Table

states	inputs: 0	inputs: 1
q_0	q_0	q_1
q_1	q_2	q_1
q_2	q_2	q_2

The language of a DFA (Q, S, d, q_0, F) is the set of all strings over S that, starting from q_0 and following the transitions as the string is read left to right, will reach some accepting state.

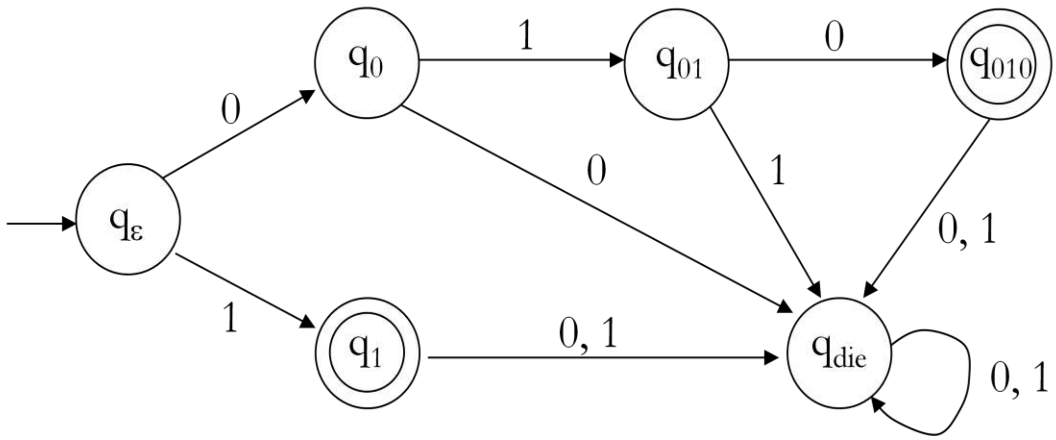
Examples #7:

Construct a DFA that accepts the language

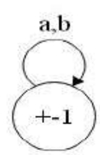
$$L = \{010, 1\} \qquad (\Sigma = \{0, 1\})$$

Compiler Construction (CS-636)

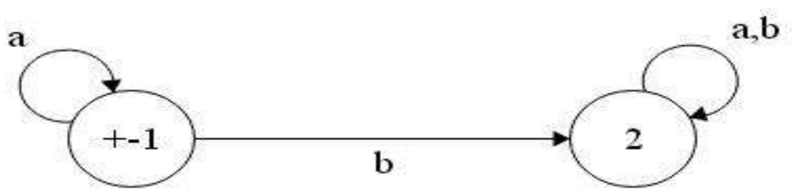
Dr. Naseer Ahmed Sajid **email id: naseer@biit.edu.pk Whatsapp# 0346-5100010**



Examples # 8: FSA for $(a+b)^*$

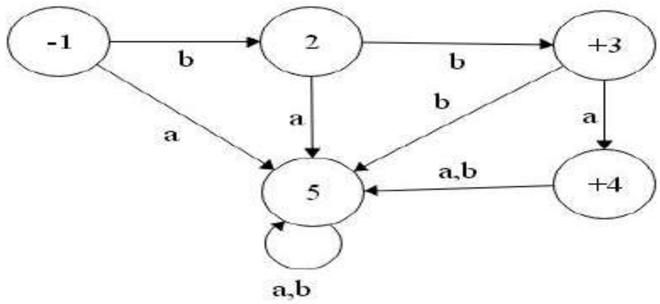


Examples # 9: FSA for a*

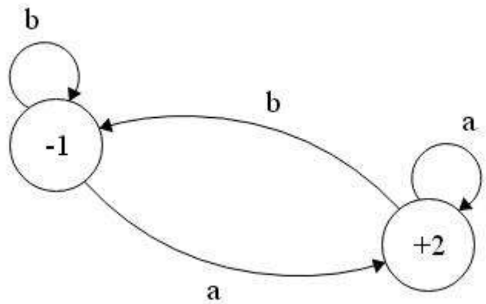


Examples # 10: FSA for bba+bb

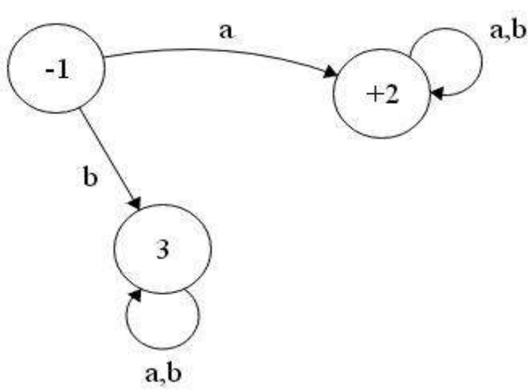
Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010



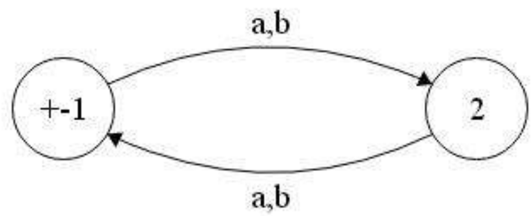
Examples # 11: FSA for (a+b)*a



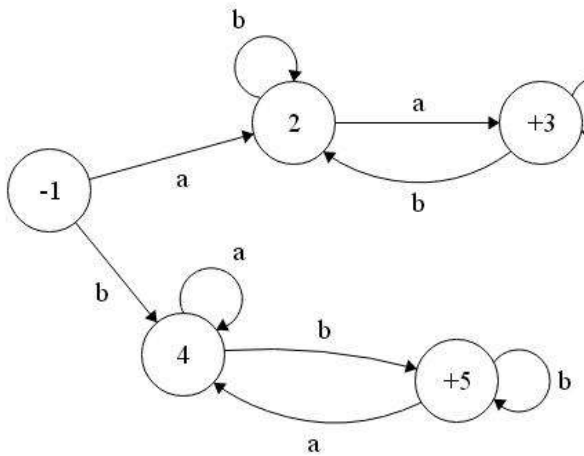
Examples # 12: FSA for a(a+b)*



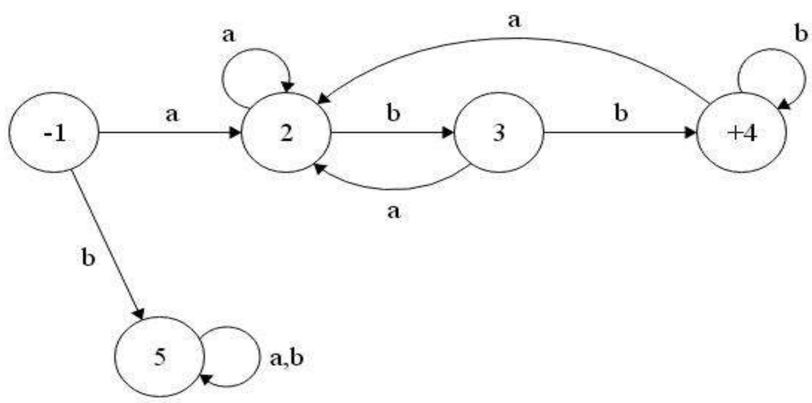
Examples # 13: Language which has even length



Examples # 14: Language which start and end with same letters and length >=2



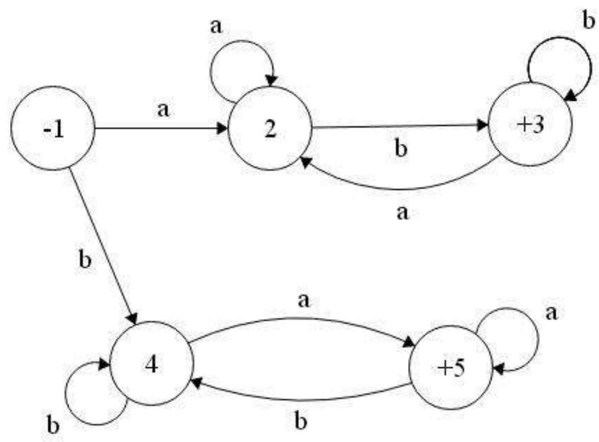
Examples # 15: Language which start with 'a' and end with 'bb'



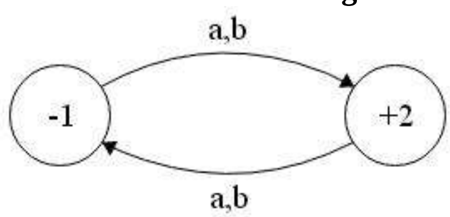
Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

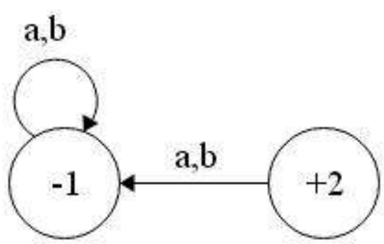
Examples # 16: Language which start and end with different letters



Examples # 17: Language of all words which has odd length



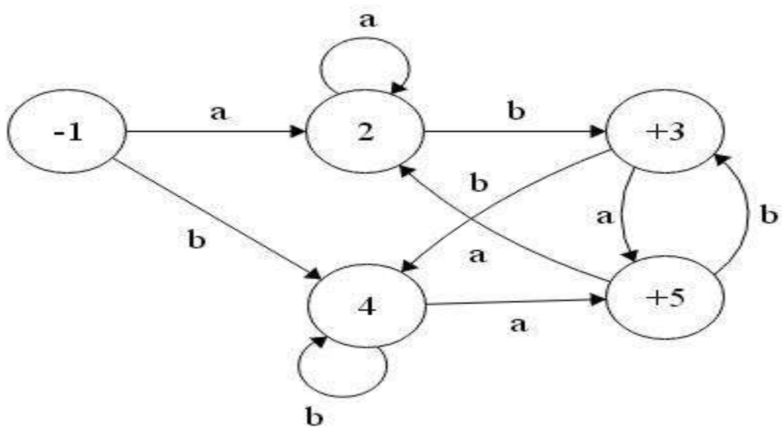
Examples # 18: Language which does not accept any letter/string



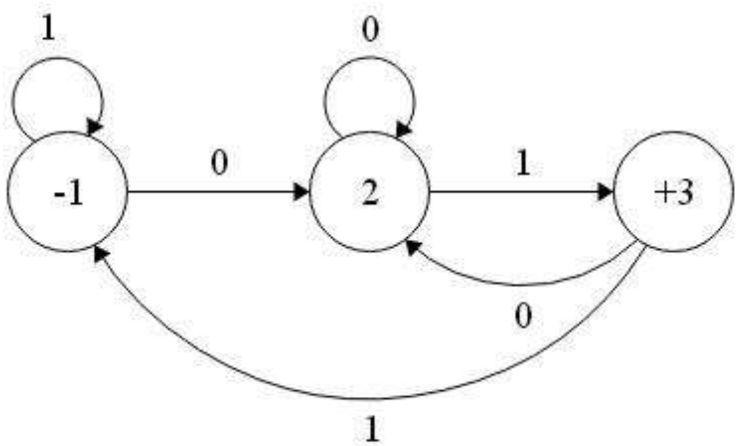
Examples # 19: Language which accept words of length >=2 and neither end with 'aa' nor with 'bb'

Compiler Construction (CS-636)

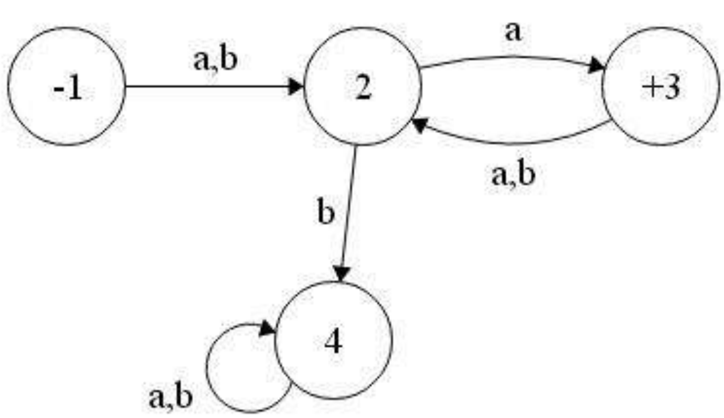
Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010



Examples # 20: Language which accept string that end with ‘01’



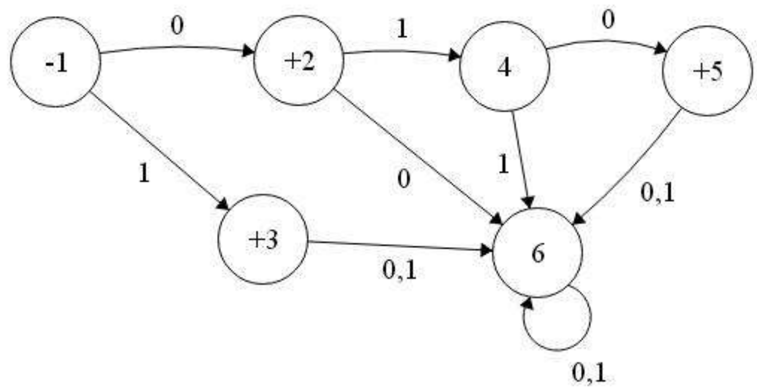
Examples # 21: Language which accept even length words and its even letter must be ‘a’



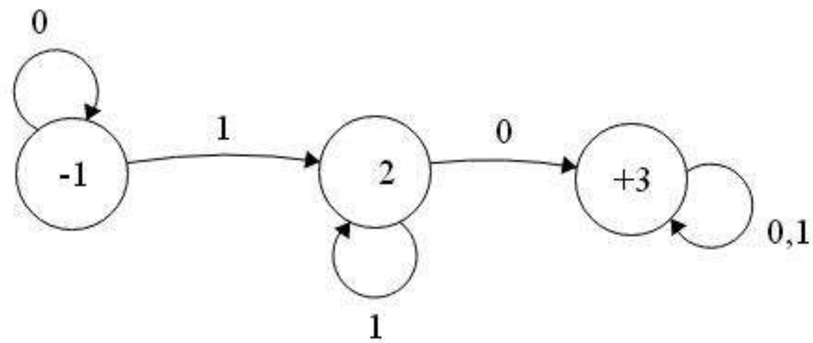
Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid **email id: naseer@biit.edu.pk Whatsapp# 0346-5100010**

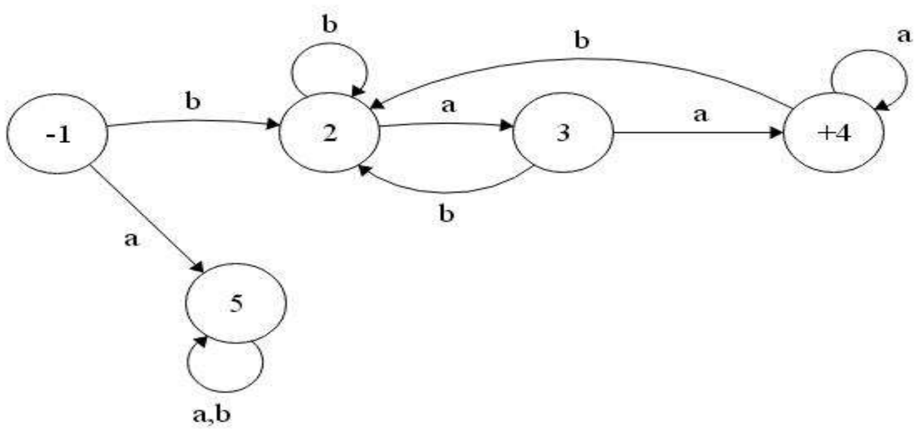
Examples # 22: FSA for L={0,1,010}



Examples # 23: Language which must contain ‘01’



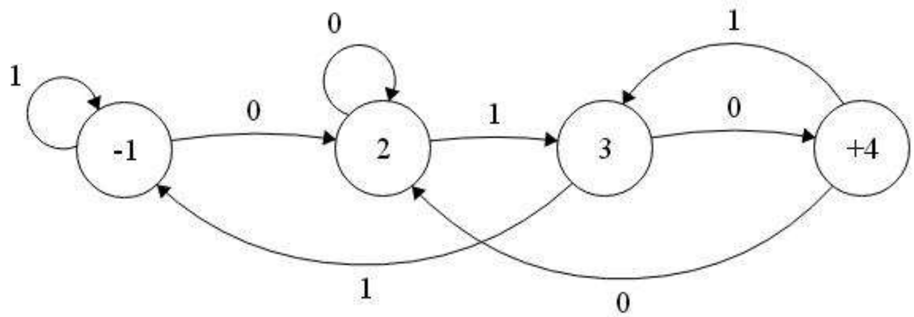
Examples # 24: Language which start with ‘b’ and end with ‘aa’ for length >=3



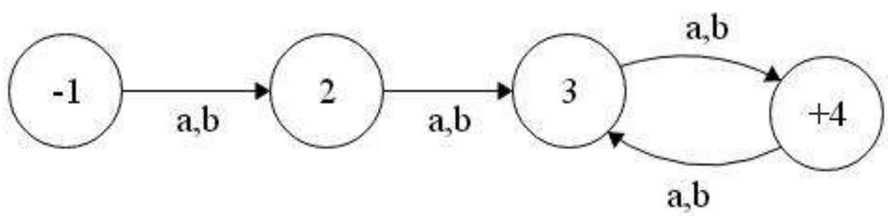
<u>Compiler Construction (CS-636)</u>

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

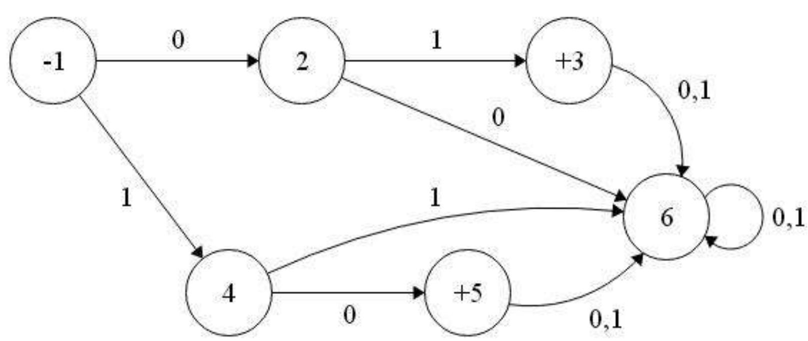
Examples # 25: Language which accepts string that end with '010'



Examples # 26: Language of odd length >1

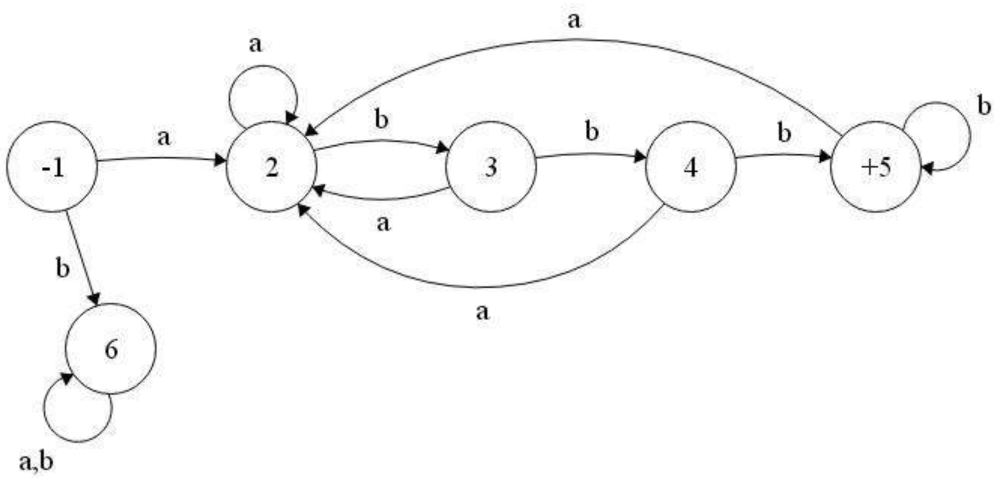


Examples # 27: Language which accept only '01' or '10'

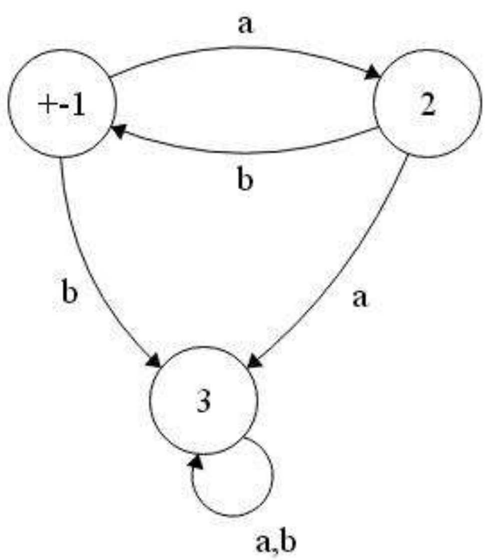


Examples # 28: Language which start with 'a' and end with 'bbb'

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010



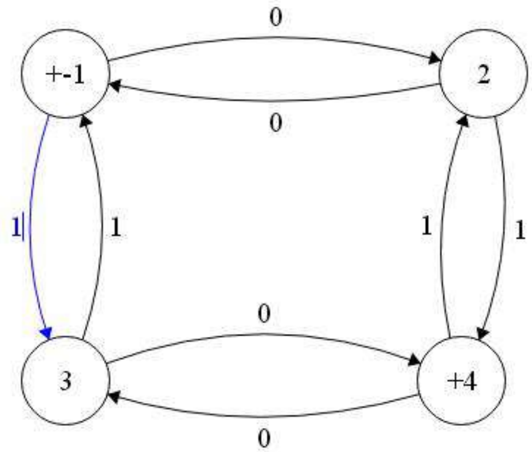
Examples # 29: Language for (ab)*



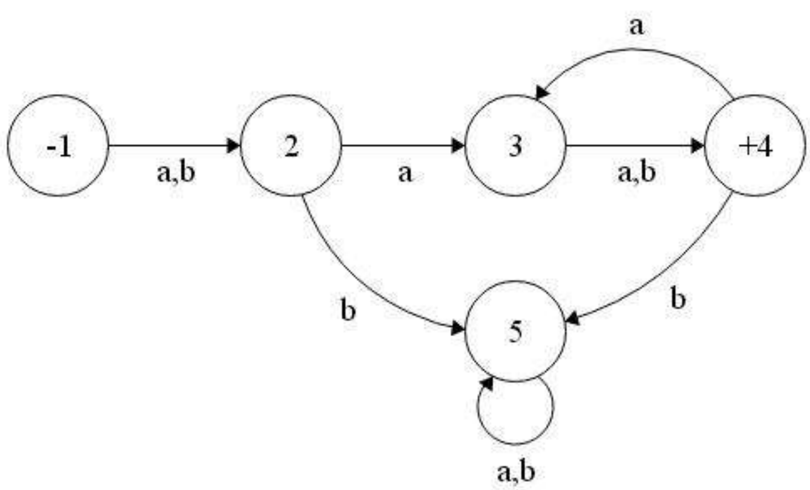
Examples # 30: Language which contain even number of 1's OR odd number of 0's

Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

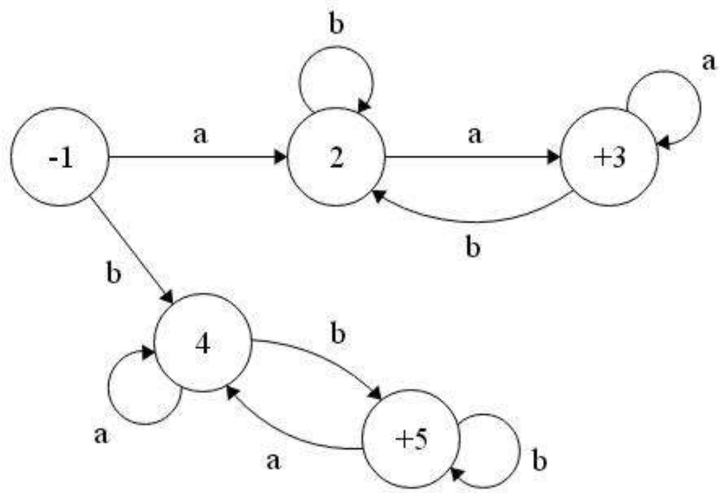


Examples # 31: Language which accept odd length but its even letter must be 'a'

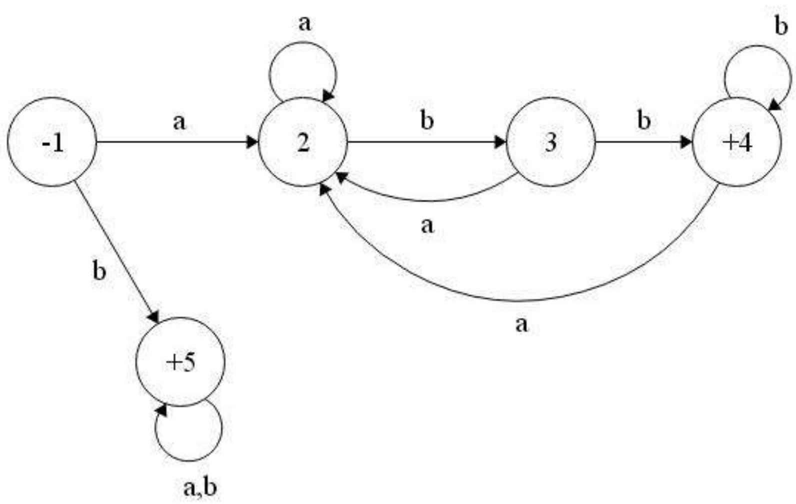


Examples # 32: Language which start and end with same letters and length >1

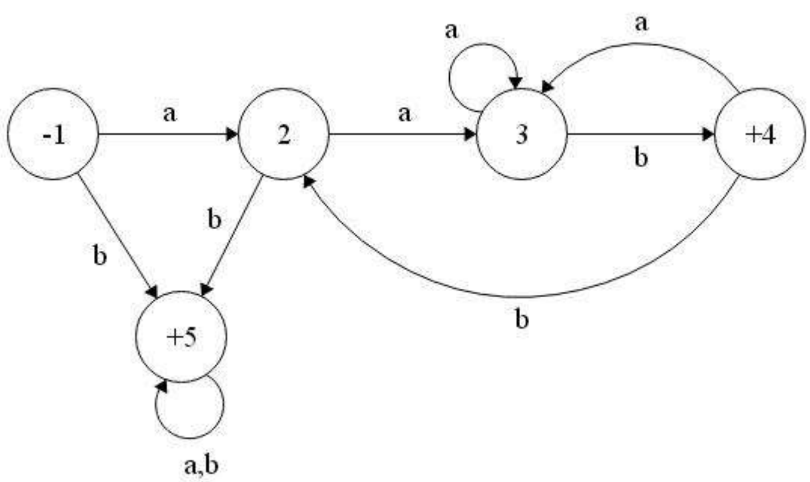
Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010



Examples # 33: Language which start with ‘a’ and end with ‘bb’

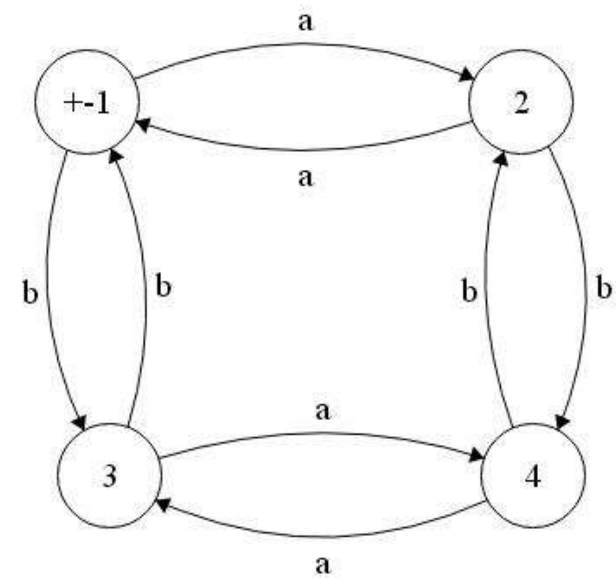


Examples # 34: Language of words which start with ‘aa’ and end with ‘ab’ and length >2

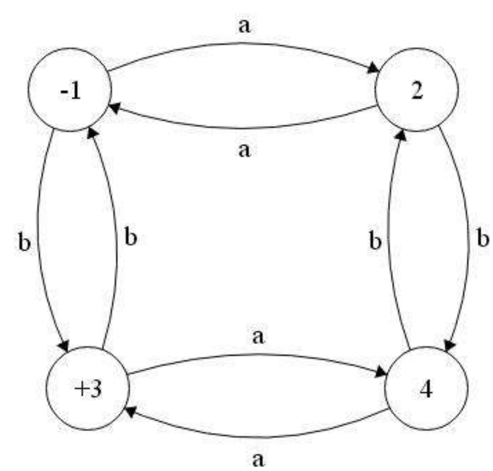


Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

Examples # 35: Language which contain EVEN number of a's AND EVEN number of b's

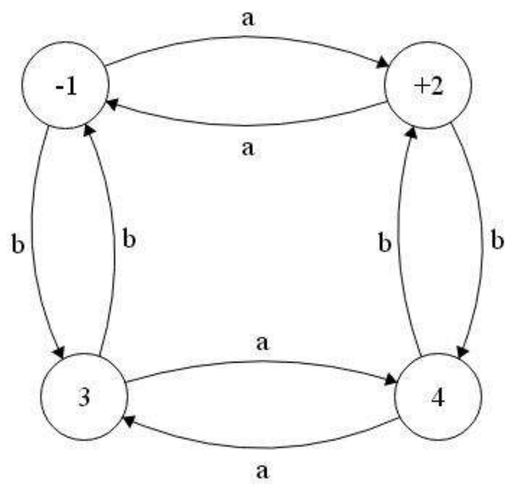


Examples # 36: Language which contain EVEN number of a's AND ODD number of b's

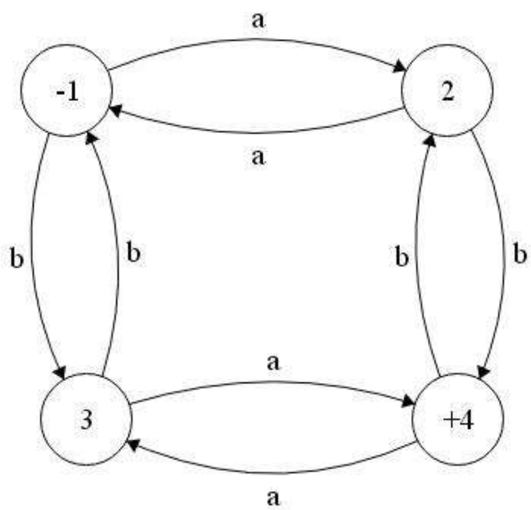


Examples # 37: Language which contain ODD number of a's AND EVEN number of b's

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

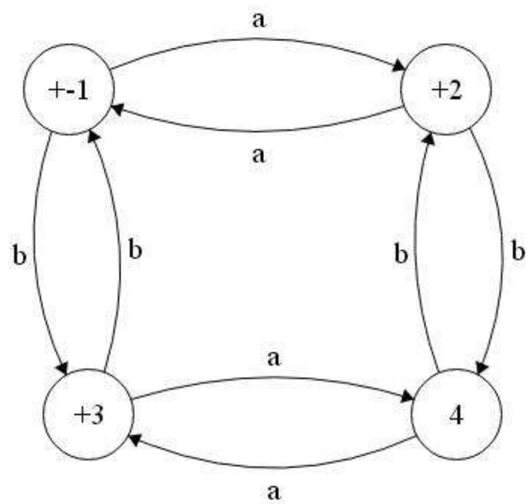


Examples # 38: Language which contain ODD number of a's AND ODD number of b's

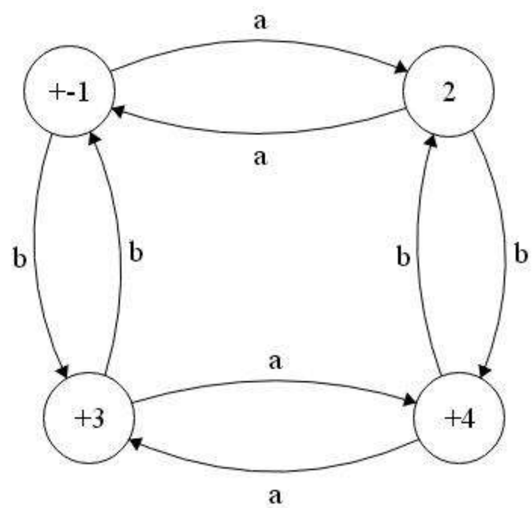


Examples # 39: Language which contain EVEN number of a's OR EVEN number of b's

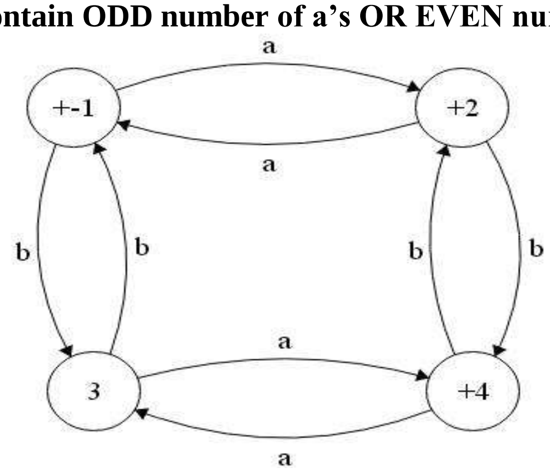
Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010



Examples # 40: Language which contain EVEN number of a's OR ODD number of b's

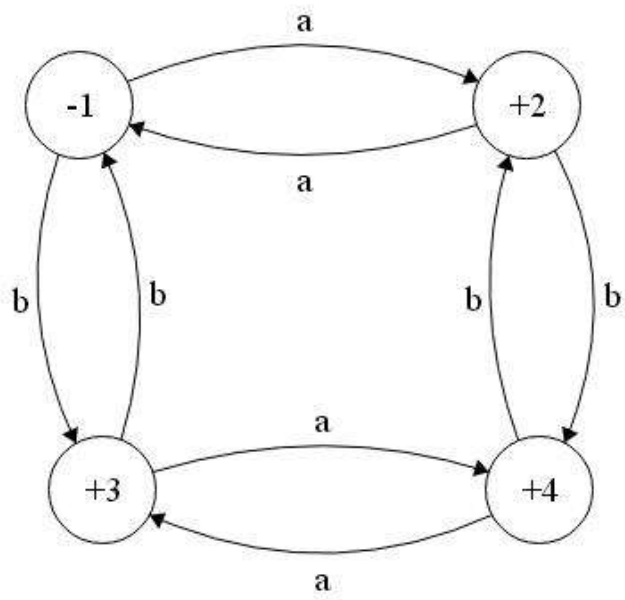


Examples # 41: Language which contain ODD number of a's OR EVEN number of b's

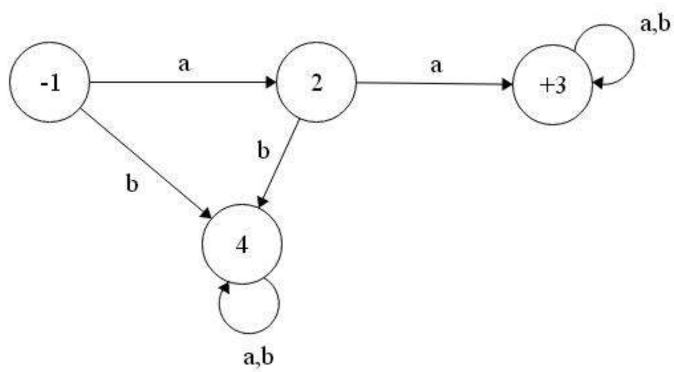


Examples # 42: Language which contain ODD number of a's OR ODD number of b's

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

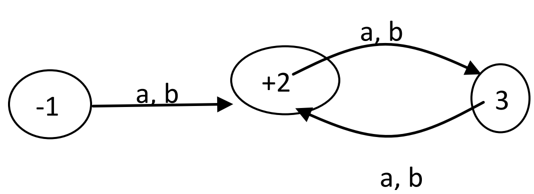


Examples # 43: Language which start with ‘aa’



Language of FSA: We can write a language from an FSA by analyzing it. To write a language for given FSA we have to keep our eyes on acceptance words of that particular FSA. It is elaborated with the help of examples below

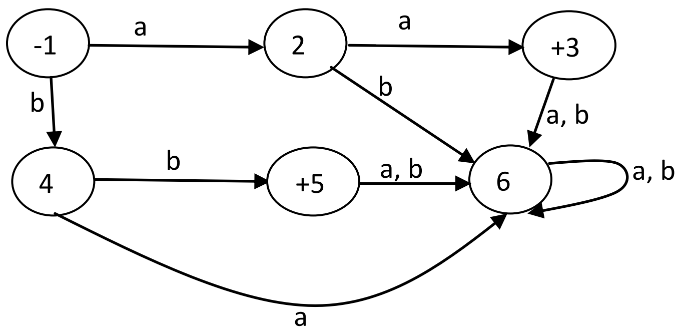
Example 1:



Language for the given FSA is : Language having words of odd length

i.e. (a + b)((a + b)(a + b))*

Dr. Naseer Ahmed Sajid **email id: naseer@biit.edu.pk Whatsapp# 0346-5100010**



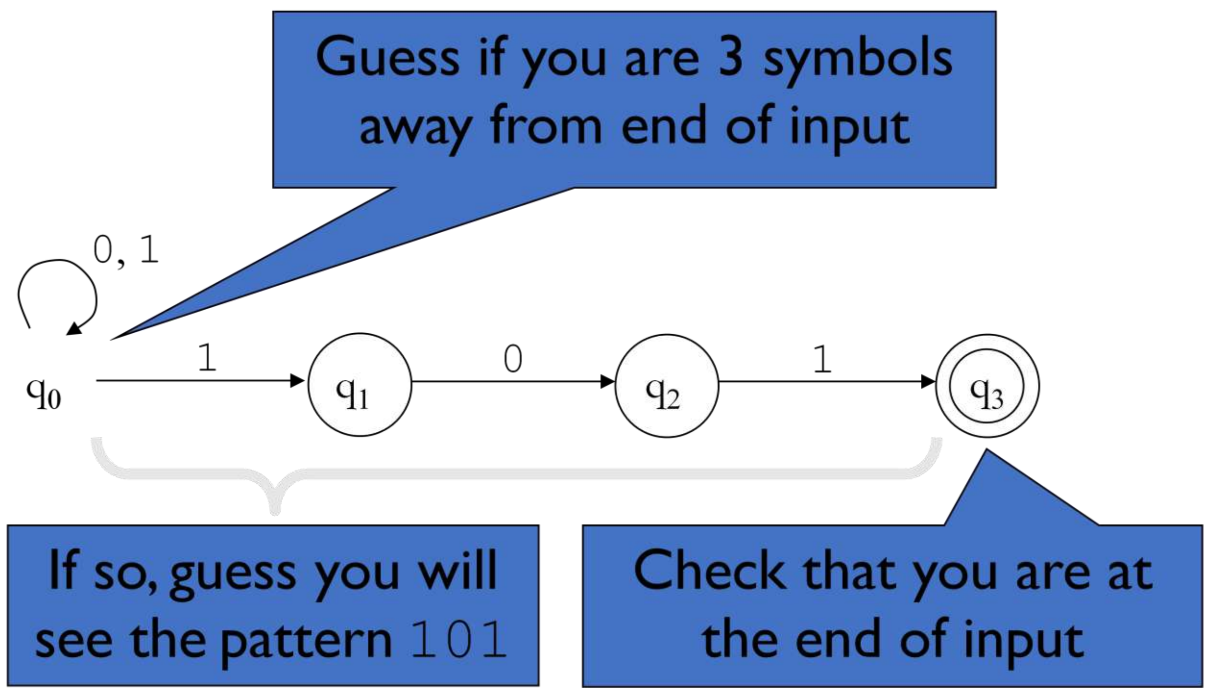
Language for the given FSA is : Language having two words aa and bb

i.e. aa + bb

Nondeterministic Finite Automata (NFA)

- Can be at multiple states at the same time
- Can go to multiple states on one alphabet
- May have no transitions on an alphabet (die)

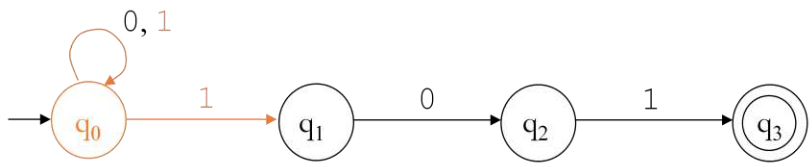
Meaning of NFA



Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid **email id: naseer@biit.edu.pk Whatsapp# 0346-5100010**

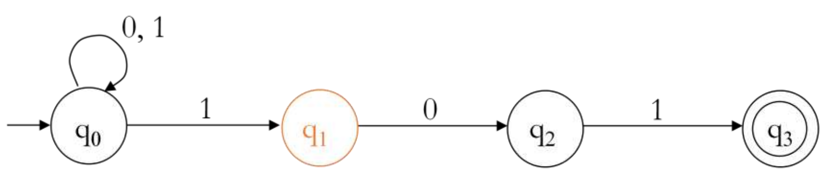
Choosing Where to Go



- State q_0 has two transitions labeled 1
- Upon reading 1, we have the choice of staying in q_0 or moving to q_1

The ability to make choices

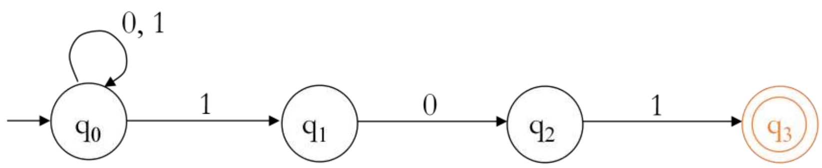
Case 1



- State q_1 has no transition labeled 1
- Upon reading 1 in q_1, we die; upon reading 0, we continue to q_2

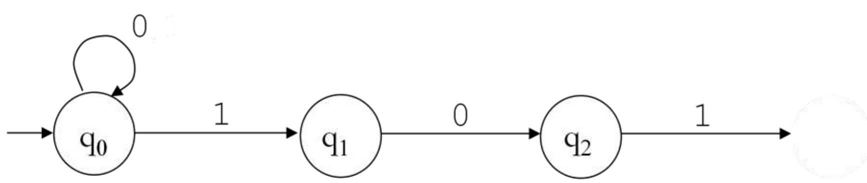
Case 2:

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010



- State q_3 has no transition going out
- Upon reading 0 or 1 in q_3, we die

How to run an NFA?



input: 0 1 1 0 1

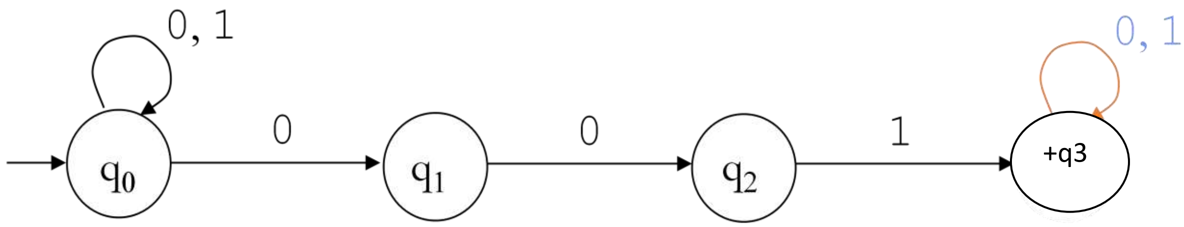
The NFA can have several active states at the same time

It accepts if at the end, one of its active states is final

Example 1:

Construct an NFA over alphabet {0, 1} that accepts those strings that contain the pattern 001 somewhere.

Dr. Naseer Ahmed Sajid email id: naseer@biit.edu.pk Whatsapp# 0346-5100010

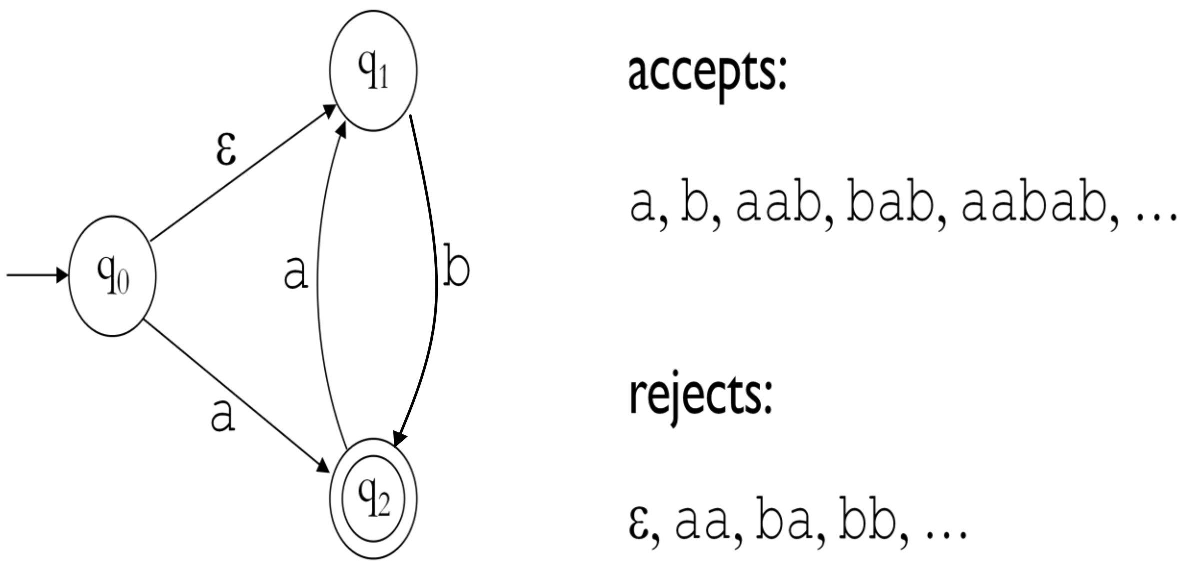


Language of NFA:

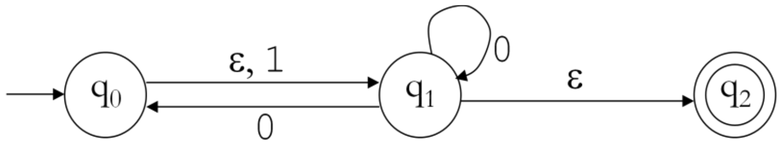
The NFA accepts string x ε $\sum^*$ if there is some path that, starting from q_0, leads to an accepting state as the string is read left to right.

The language of an NFA is the set of all strings that the NFA accepts.

ε – Transitions



Example 2:



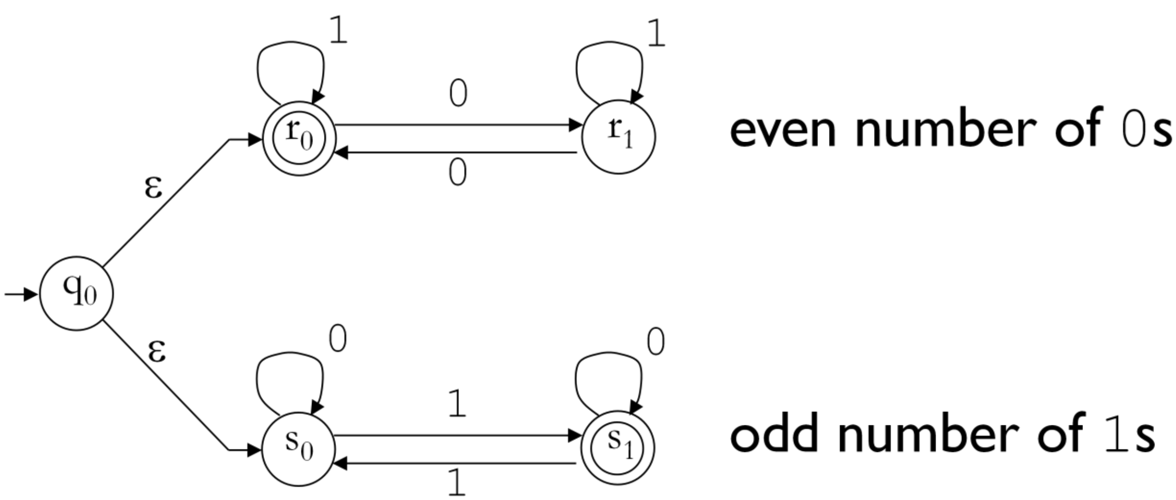
alphabet $\Sigma = \{0, 1\}$
states $Q = \{q_0, q_1, q_2\}$
initial state q_0
accepting states $F = \{q_2\}$

table of
transition function δ:

states	inputs: 0	1	ε
q_0	$\varnothing$	$\{q_1\}$	$\{q_1\}$
q_1	$\{q_0, q_1\}$	$\varnothing$	$\{q_2\}$
q_2	$\varnothing$	$\varnothing$	$\varnothing$

Example 3:

- Construct an NFA that accepts all strings with an even number of 0s or an odd number of 1s



Eliminating ε-transitions

Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid **email id: naseer@biit.edu.pk Whatsapp# 0346-5100010**

- q_0 can reach q_0 on 0 (q_0 -> q_1 -> q_0)
- q_0 can reach q_1 on 0 (q_0 -> q_1)
- q_1 can reach q_0 on 0 (q_1 -> q_0)
- q_1 can reach q_1 on 0 (q_1 -> q_1)
- The rest of the transitions remain unchanged

	0	**1**
q_0	q_0, q_1	q_2
q_1	q_0, q_1	
q_2	q_1	q_1, q_2

- q_0 is the initial state
- q_0 can reach an accepting state on ε (i.e. q_1)
- therefore q_0 is also an accepting state

Compiler Construction (CS-636)

Dr. Naseer Ahmed Sajid   email id: naseer@biit.edu.pk Whatsapp# 0346-5100010