Finite Automata Theory and Formal Languages

(Week 2, Lecture 1)

Finite Automaton (FA)

- A collection of the followings:
  - Finite number of states, having one initial and some (maybe none) final states.
  - Finite set of input letters (Σ) from which input strings are formed.
  - Finite set of transitions i.e. for each state and for each input letter there is a transition showing how to move from one state to another.

Finite Automaton (FA)

$\Sigma = \{a,b\}$
States: x, y, z where x is an initial state and z is final state.
Transitions:

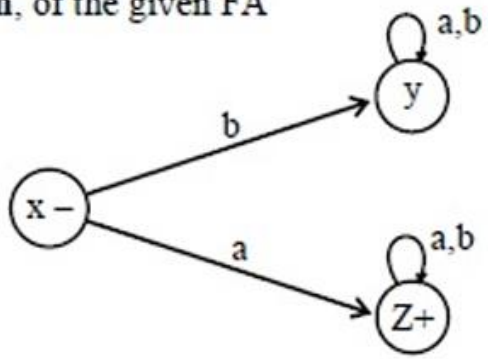
1. At state x reading a, go to state z
2. At state x reading b, go to state y
3. At state y reading a, b go to state y
4. At state z reading a, b go to state z

These transitions can be expressed by the following table called transition table

Old States	New States	
	Reading a	Reading b
x -	z	y
y	y	y
z +	z	z

Finite Automaton (FA)

It may be noted that the information of an FA, given in the previous table, can also be depicted by the following diagram, called the **transition diagram**, of the given FA



Remark

The above transition diagram is an FA accepting the language of strings, defined over $\Sigma = \{a, b\}$, **starting with a**. It may be noted that this language may be expressed by the regular expression $a(a + b)^*$

Finite Automaton (FA)

$\Sigma = \{a,b\}$

States: x, y, where x is both initial and final state.

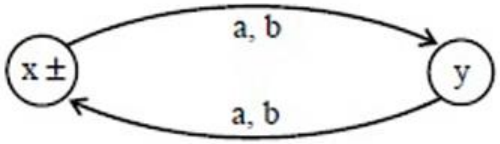
Transitions:

1. At state x reading a or b go to state y.
2. At state y reading a or b go to state x.

These transitions can be expressed by the following transition table

Old States	New States	
	Reading a	Reading b
x ±	y	y
y	x	x

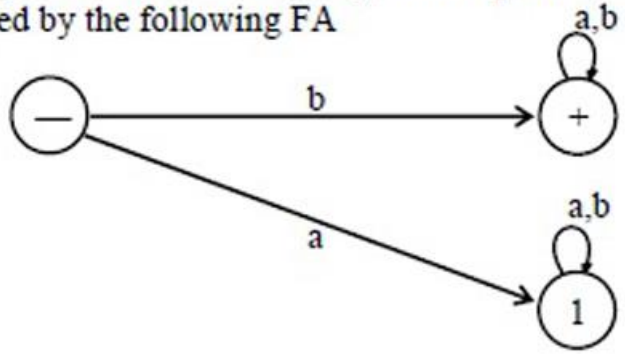
It may be noted that the above transition table may be depicted by the following transition diagram.



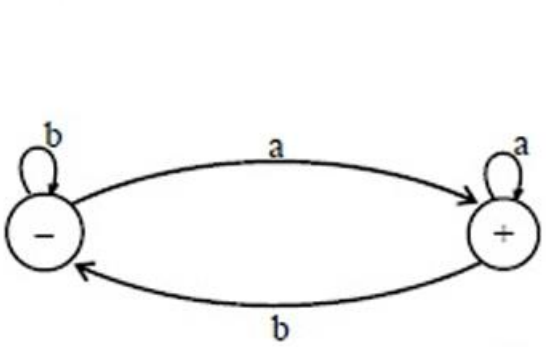
The above transition diagram is an FA accepting the language of strings, defined over $\Sigma=\{a, b\}$ of **even length**. It may be noted that this language may be expressed by the regular expression $((a+ b) (a + b))^*$

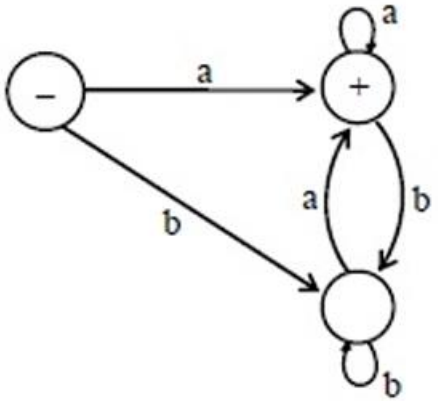
Finite Automaton (FA)

Consider the language L of strings, defined over Σ={a, b}, **starting with b.** The language L may be expressed by RE $b(a + b)^*$, may be accepted by the following FA



Consider the language L of strings, defined over Σ={a, b}, **ending in a.** The language L may be expressed by RE $(a+b)^* a$.



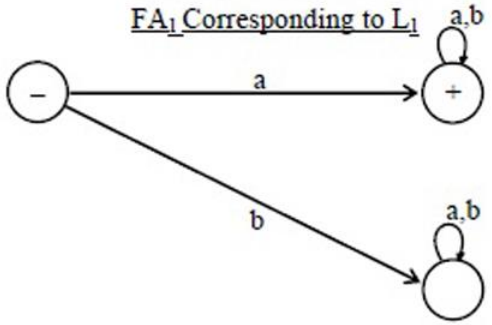


Finite Automaton (FA)

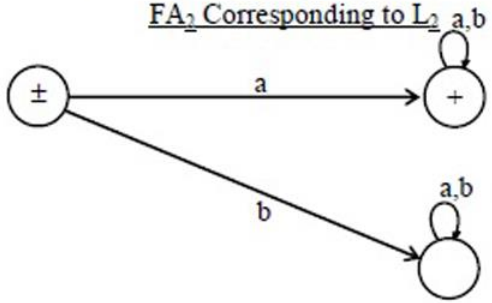
- Note:
  - The corresponding to a given language there may be more than one FA accepting that language, but for a given FA there is a unique language accepted by that FA.

Finite Automaton (FA)

It is to be noted that given the languages L_1 and L_2 ,where
L_1 = The language of strings, defined over $\Sigma = \{a, b\}$, **beginning with a.**
L_2 = The language of strings, defined over $\Sigma = \{a, b\}$, **not beginning with b**
The Λ does not belong to L_1 while it does belong to L_2 . This fact may be depicted by the corresponding transition diagrams of L_1 and L_2.



The language L_1 may be expressed by the regular expression $a(a + b)^*$



The language L_2 may be expressed by the regular expression $a(a + b)^* + \Lambda$

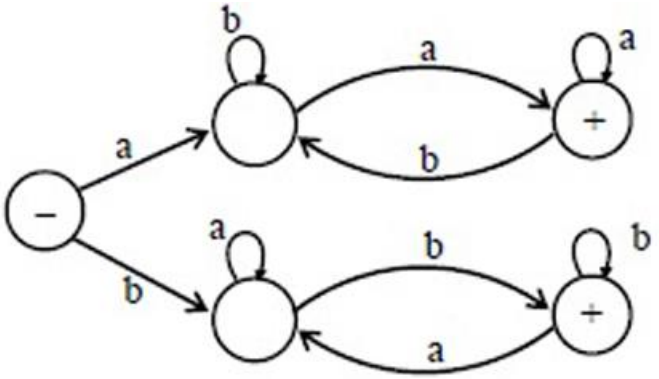
Finite Automaton (FA)

Consider the Language L of Strings of **length two or more**, defined over $\Sigma = \{a, b\}$, **beginning with and ending in same letters.**

The language L may be expressed by the following regular expression $a(a + b)^*a + b(a + b)^*b$

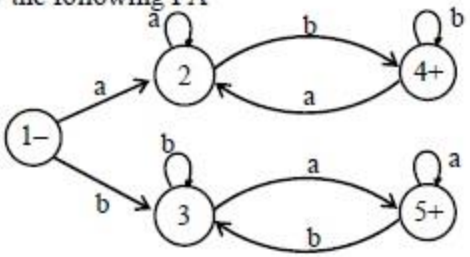
It is to be noted that if the condition on the length of string is not imposed in the above language then **the strings a and b will then belong to the language.**

This language L may be accepted by the FA as shown aside

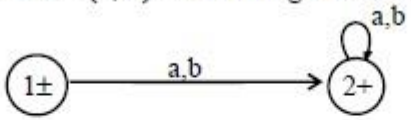


Finite Automaton (FA)

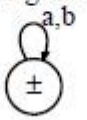
Consider the Language L of Strings, defined over $\Sigma = \{a, b\}$, **beginning with and ending in different letters.**
The language L may be expressed by the following regular expression $a(a + b)^*b + b(a + b)^*a$
This language may be accepted by the following FA



Consider the Language L, defined over $\Sigma = \{a, b\}$ of **all strings including Λ**. The language L may be accepted by the following FA



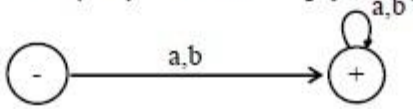
The language L may also be accepted by the following FA



The language L may be expressed by the regular expression $(a + b)^*$

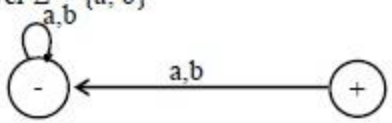
Finite Automaton (FA)

Consider the Language L , defined over $\Sigma = \{a, b\}$ of **all non empty strings**. The language L may be accepted by the following FA



The above language may be expressed by the regular expression $(a + b)^+$

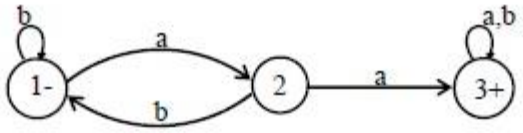
Consider the following FA, defined over $\Sigma = \{a, b\}$



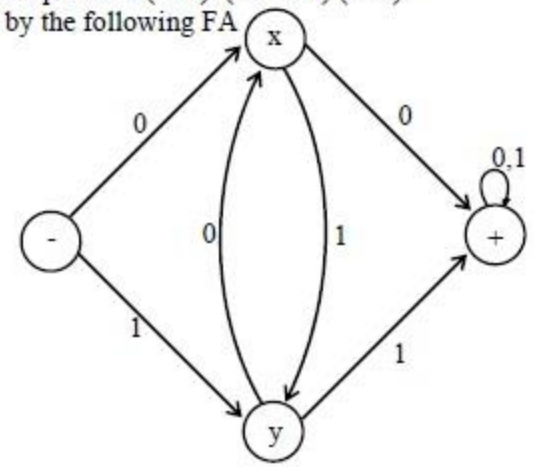
It is to be noted that the above FA **does not accept any string,** even it does not accept the null string; as there is no path starting from initial state and ending in final state.

Finite Automaton (FA)

Consider the Language L of strings , defined over Σ = {a, b}, **containing double a.**
The language L may be expressed by the regular expression $(a+b)^*$ (aa) $(a+b)^*$. This language may be accepted by the following FA.



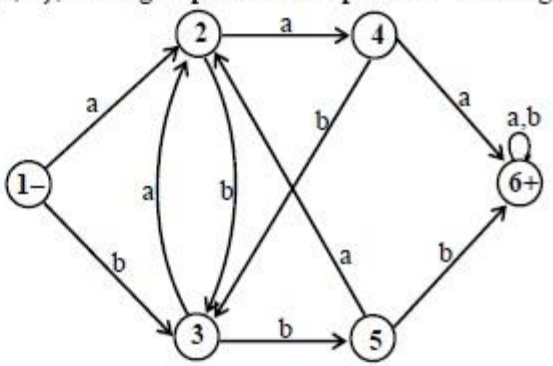
Consider the language L of strings, defined over Σ={0, 1}, **having double 0's or double 1's,** The language L may be expressed by the regular expression $(0+1)^*$ (00 + 11) $(0+1)^*$
This language may be accepted by the following FA



Finite Automaton (FA)

Consider the language L of strings, defined over Σ={a, b}, **having triple a's or triple b's.** The language L may be expressed by RE $(a+b)^*(aaa + bbb)(a+b)^*$

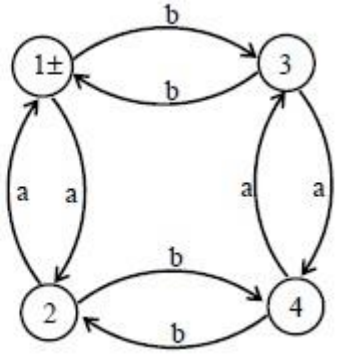
This language may be accepted by the FA as shown aside



Consider the **EVEN-EVEN** language, defined over Σ = {a, b}. As discussed earlier that **EVEN-EVEN** language can be expressed by the regular expression

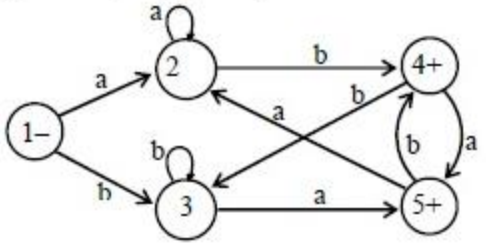
$$(aa+bb+(ab+ba)(aa+bb)^*(ab+ba))^*$$

EVEN-EVEN language may be accepted by the FA as shown aside

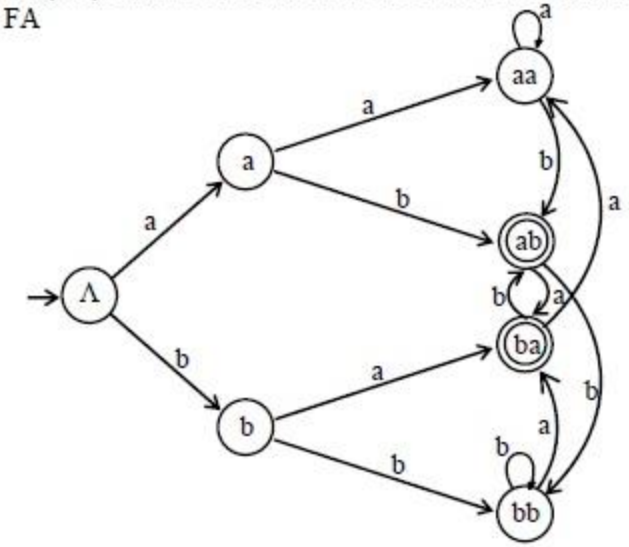


FA – Alternative Technique

Consider the language L = {w belongs to {a,b}*: length(w) ≥2 and w neither ends in **aa** nor **bb**}.
The language L may be expressed by the regular expression $(a+b)^*(ab+ba)$
This language may be accepted by

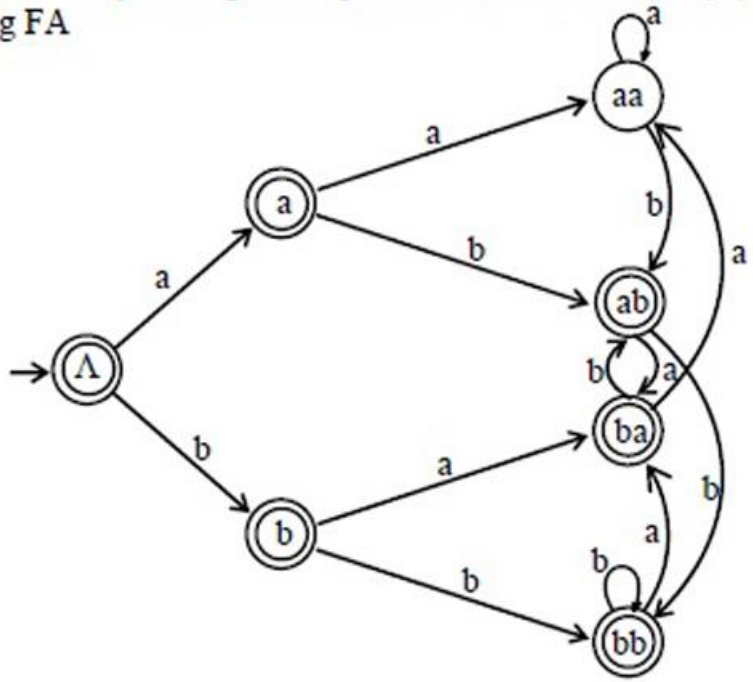


Note
It is to be noted that building an FA corresponding to the language L, discussed in the previous example, seems to be quite difficult, but the same can be done using tree structure along with the technique discussed in the book *Introduction to Languages and Theory of Computation*, by J. C. Martin
so that the strings ending in aa, ab, ba and bb should end in the states labeled as aa, ab, ba and bb, respectively; as shown in the following FA



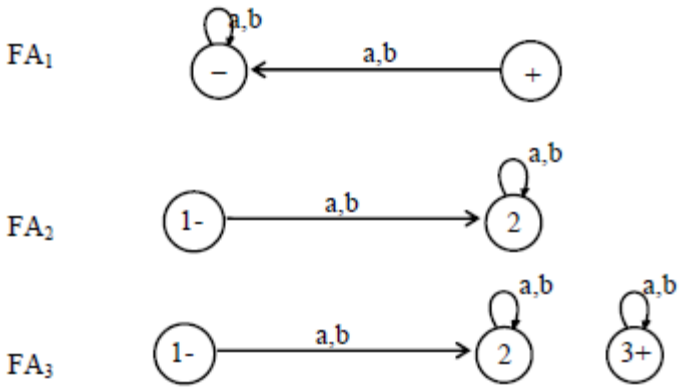
FA – Alternative Technique

Consider the language FA corresponding to r_1+r_2 can be determined as
L = {w belongs to {a,b}*: w does not end in **aa**}.
The language L may be expressed by the regular expression $\Lambda + a + b + (a+b)^*(ab+ba+bb)$. This language may be accepted by the following FA



Equivalent FAs

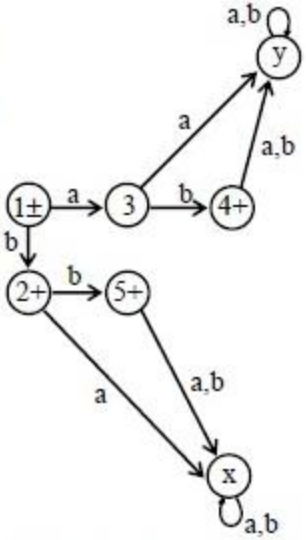
- Note:
  - It is to be noted that two FAs are said to be equivalent, if they accept the same language.

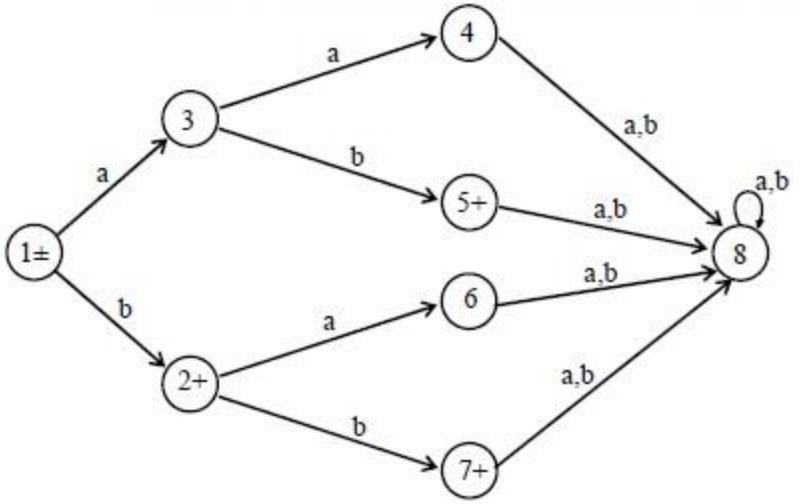


FAs for Finite Languages

Consider the language
L = {Λ, b, ab, bb}, defined over Σ = {a, b}, expressed by Λ + b + ab + bb OR Λ + b (Λ + a + b).
The language L may be accepted by the FA as shown aside

It is to be noted that the states x and y are called **Dead States, Waste Baskets or Davey John Lockers**, as the moment one enters these states there is no way to leave it.





It is to be noted that to build an FA accepting the language having less number of strings, the tree structure may also help in this regard, which can be observed in the following transition diagram for the Language L, discussed in the above example

FAs for Finite Languages

Consider the language
L = {aa, bab, aabb, bbba}, defined over Σ ={a, b},
expressed by aa + bab + aabb + bbba
OR aa (Λ + bb) + b (ab + bba)
The above language may be accepted by the FA as shown aside