Regular Expressions-2

- Equivalence of Regular Expression and Finite Automata,
- Reduction of Regular Expression to ε – NFA,
- Conversion of DFA to Regular Expression

Equivalence of RE's and Finite Automata

- We need to show that for every RE, there is an automaton that accepts the same language.
  - Pick the most powerful flexible type: the ϵ-NFA, easier to construct.
- And we need to show that for every automaton, there is a RE defining its language.
  - Pick the most restrictive type: the DFA.

Converting a RE to an ϵ-NFA

- To convert RE to FA, the simplest one is to convert RE to ϵ-NFA .

- Then the ϵ-NFA can be converted into any other FA (NFA and DFA).

- The theorem explained after this slide describes the method for conversion of RE to FA.

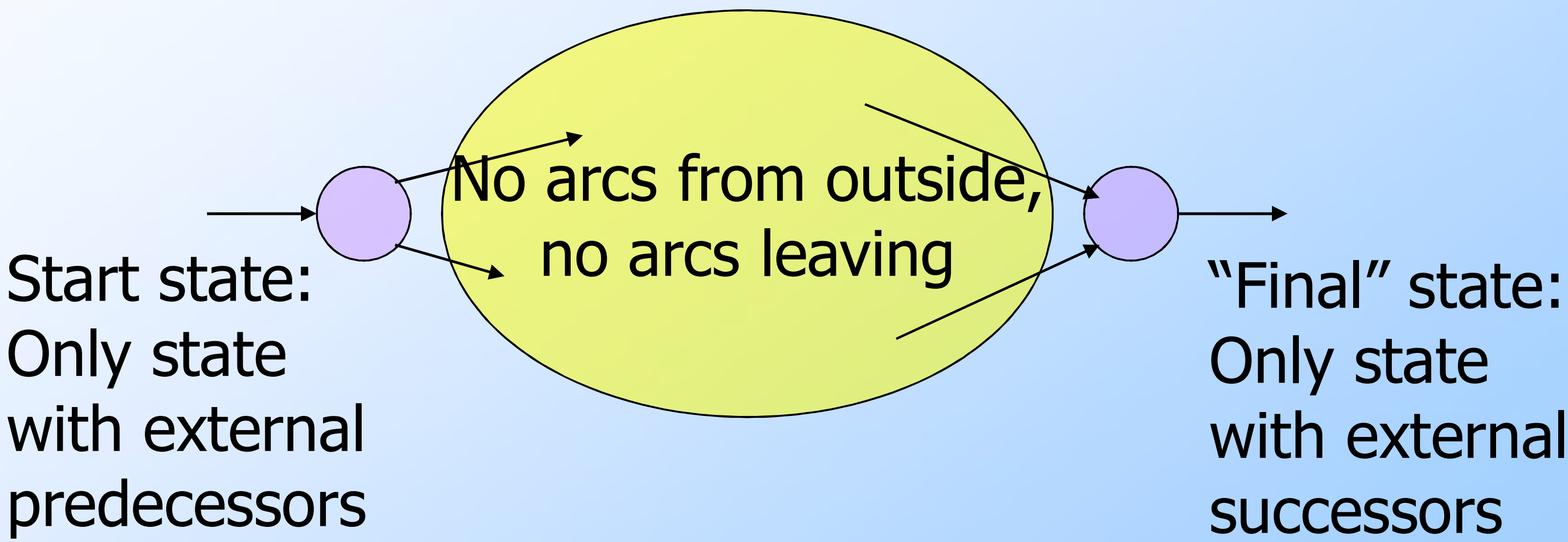
Converting a RE to an ϵ-NFA

Theorem: For any regular expression **r**, there is an ϵ-NFA that accepts the same language represented by **r.**

- **Proof:**
  - This theorem can be proved by the structural induction on the no of regular operators in regular expression.

Form of ϵ-NFA's Constructed

- We always construct an automaton of a special form as below to show structure of ϵ–NFA as below.



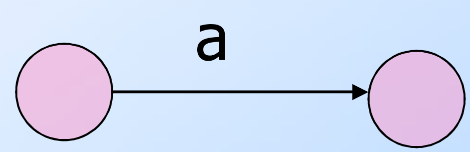
Converting a RE to an ϵ-NFA

- **Basis:** (no of operator is zero)
  - The regular Expression with no. of operator zero are: ϕ, ϵ , and **a** representing languages ϕ , {ϵ } and {a} respectively.
  - These are accepted by ϵ–NFA,s which we can show by the diagram

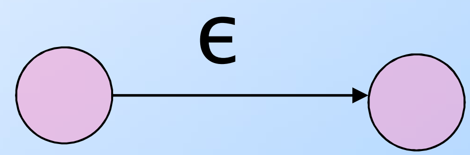
(Look at next slide)

RE to ϵ-NFA: Basis

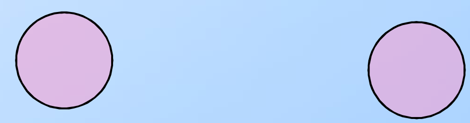
- Symbol **a**: r=**a**



- ϵ: r= ϵ

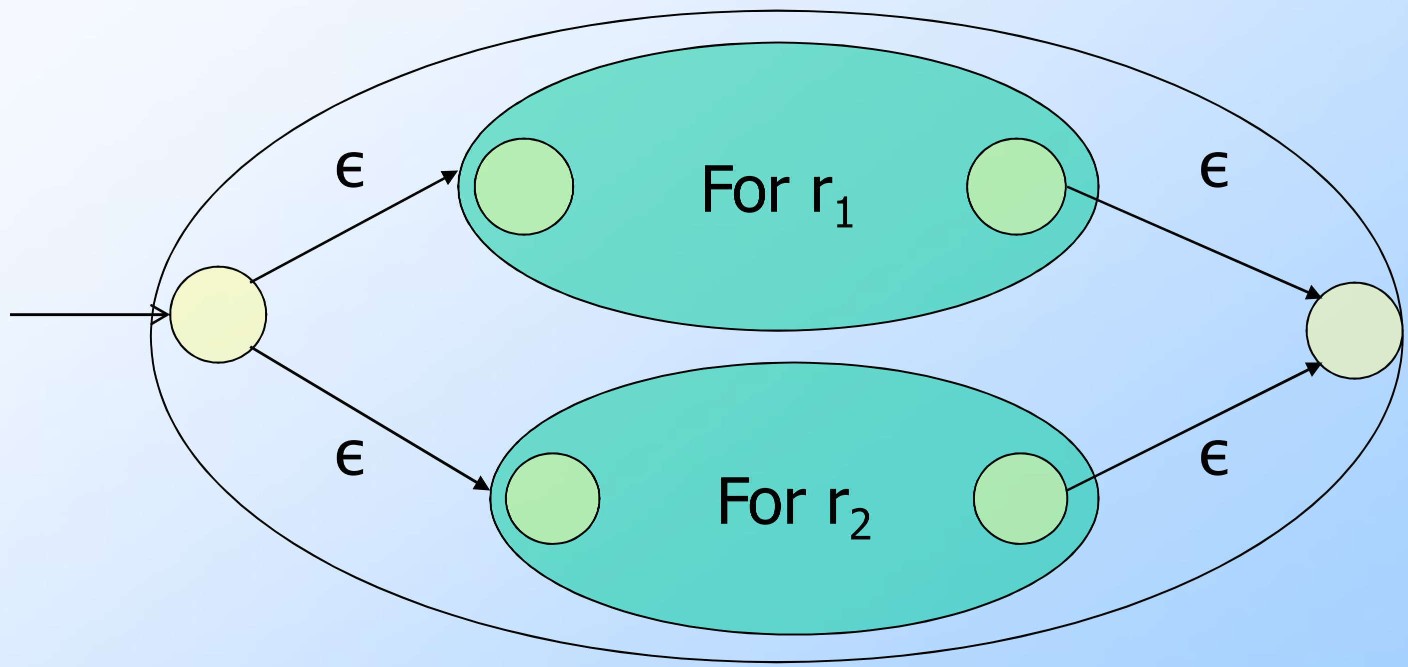


- ∅: r= ∅



RE to ϵ-NFA: Induction 1 – Union

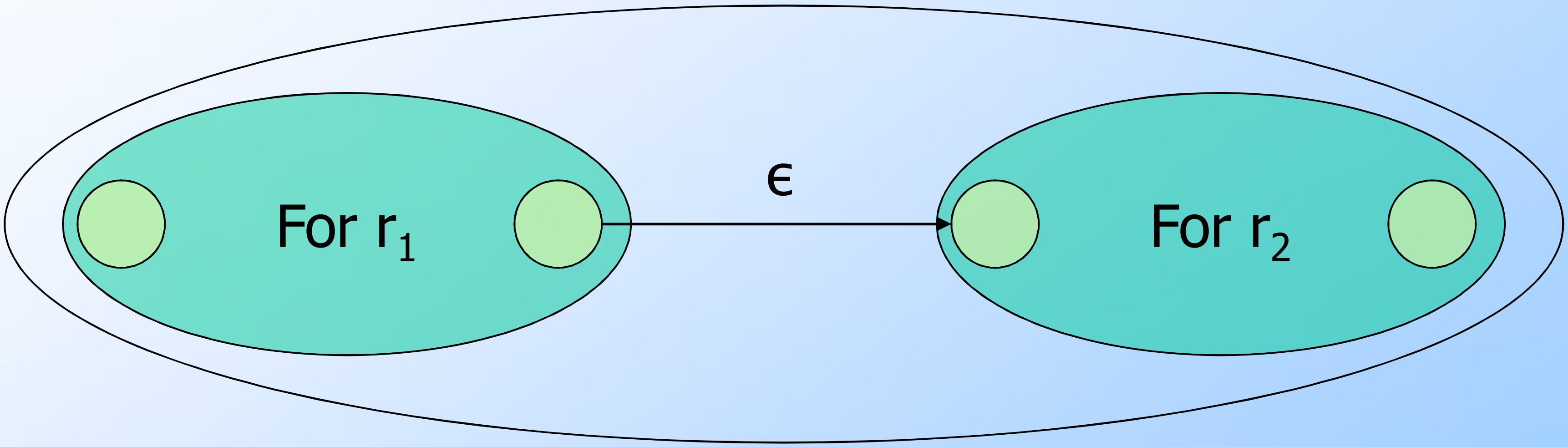
- **Let r is a regular expression such that $r=r_1+r_2$ and r_1 or r_2 have k operators and there are ϵ–NFA for them. So obiously r has at least k+1 operators.**
- **So ϵ–NFA for r can be constructed as below**



For $\mathbf{r_1 + r_2}$

RE to ϵ-NFA: Induction 2 – Concatenation

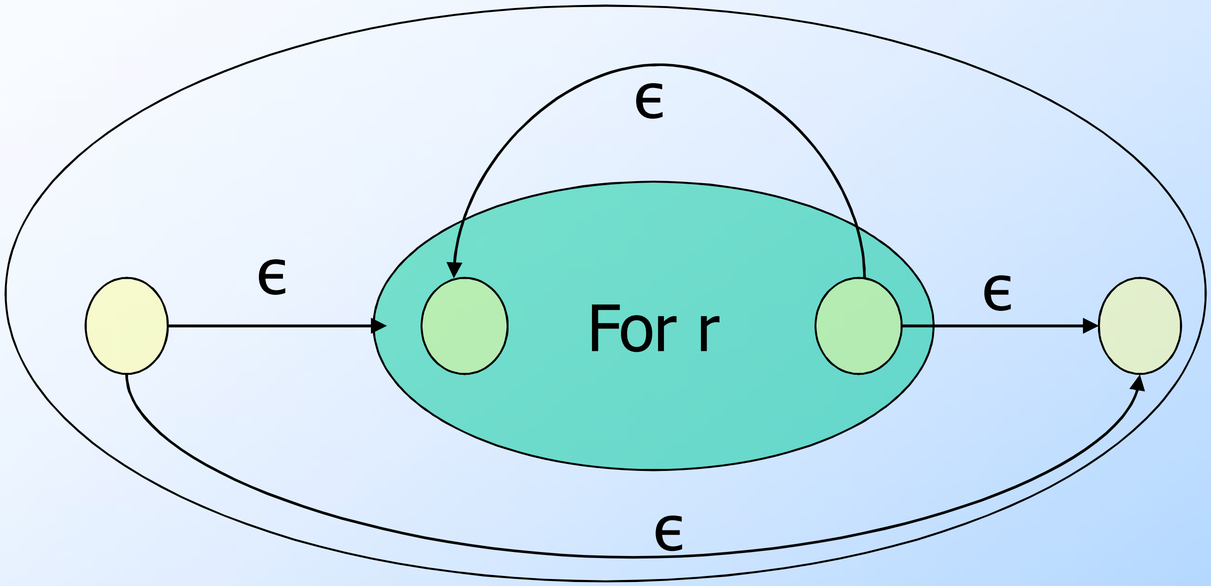
- Let r is a regular expression such that $r=r_1.r_2$ and r_1 or r_2 have k operators and there are ϵ–NFA for them. So obviously r has at least k+1 operators.
- So ϵ–NFA for r can be constructed as below



For r_1r_2

RE to ϵ-NFA: Induction 3 – Kleen Closure

- Let r is a regular expression with k operators and there is ϵ–NFA for r. So obiously r* has k+1 operators.
- So ϵ–NFA for r can be constructed as below



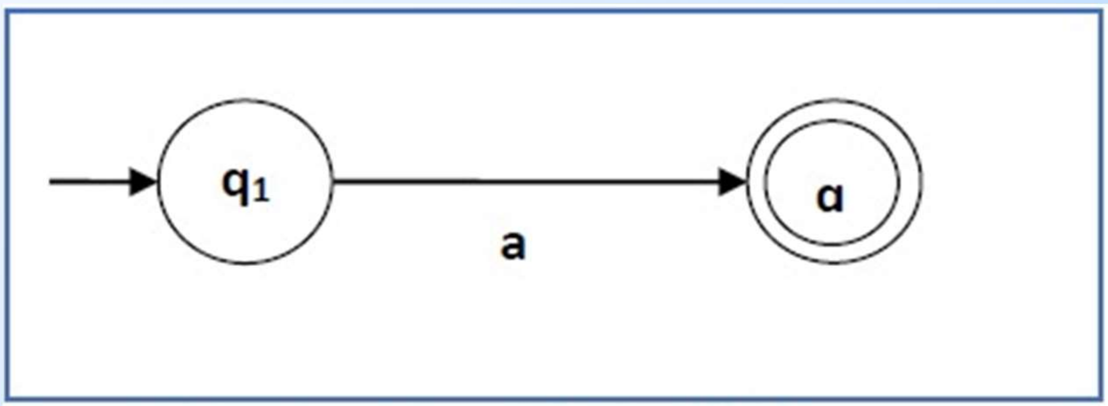
For r*

- **Clearly in induction 1,2 and 3 r has the operators k+1 or more operators, Hence we can construct ϵ-NFA accepting the any language described by regular expression.**
- **This completes the proof.**

RE to ϵ-NFA

For any given regular expression you can convert it in to FA as below:

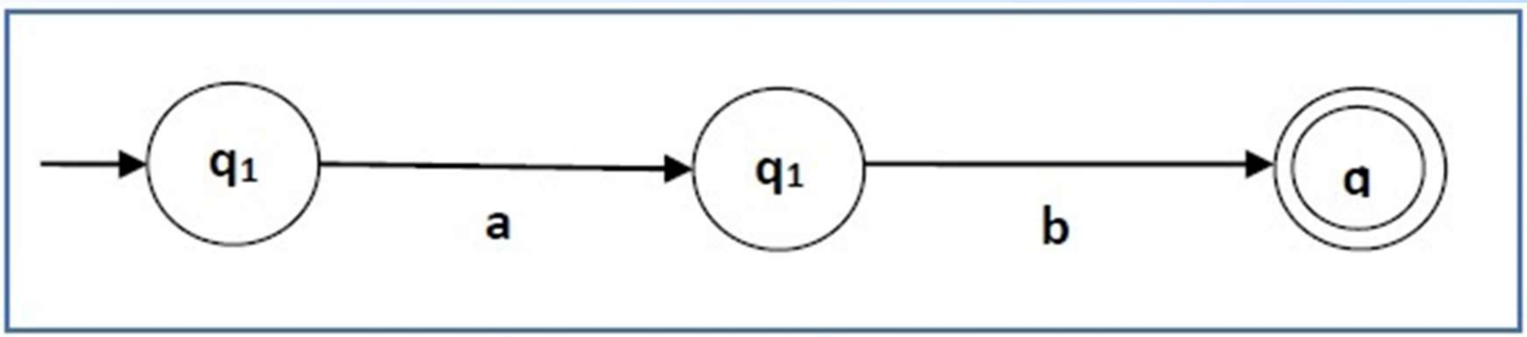
- ***Case 1*** – For a regular expression 'a', we can construct the following FA –**ϵ-NFA**



Finite automata for RE = a

RE to ϵ-NFA

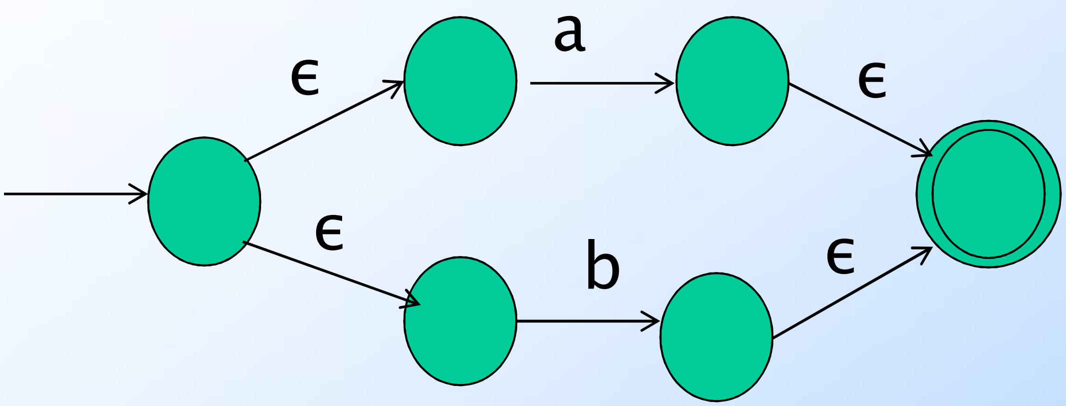
Case 2 – For a regular expression ‘ab’, we can construct the following FA –



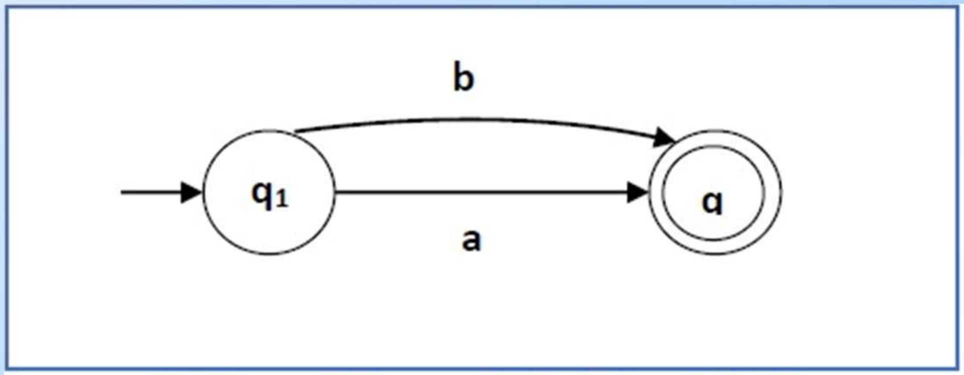
Finite automata for RE = ab

RE to ϵ-NFA

Case 3– For a regular expression 'a+b', we can construct the following FA –



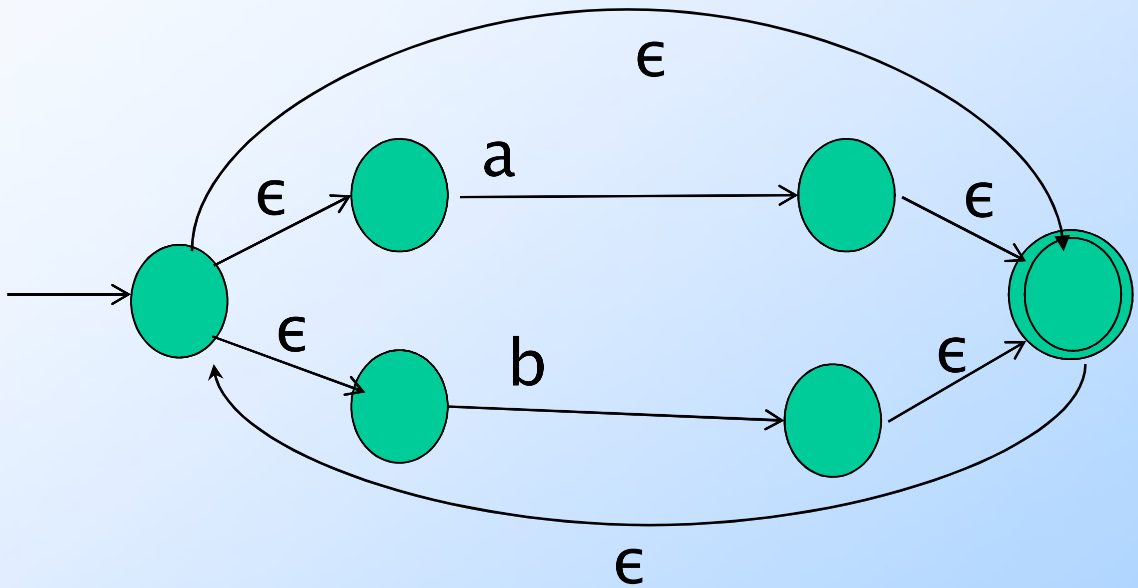
OR simply,



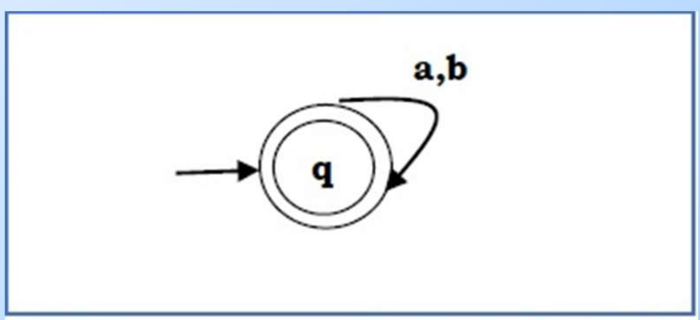
Finite automata for RE= (a+b)

RE to ϵ-NFA

- ***Case 4*** – For a regular expression (a+b)*, we can construct the following FA –



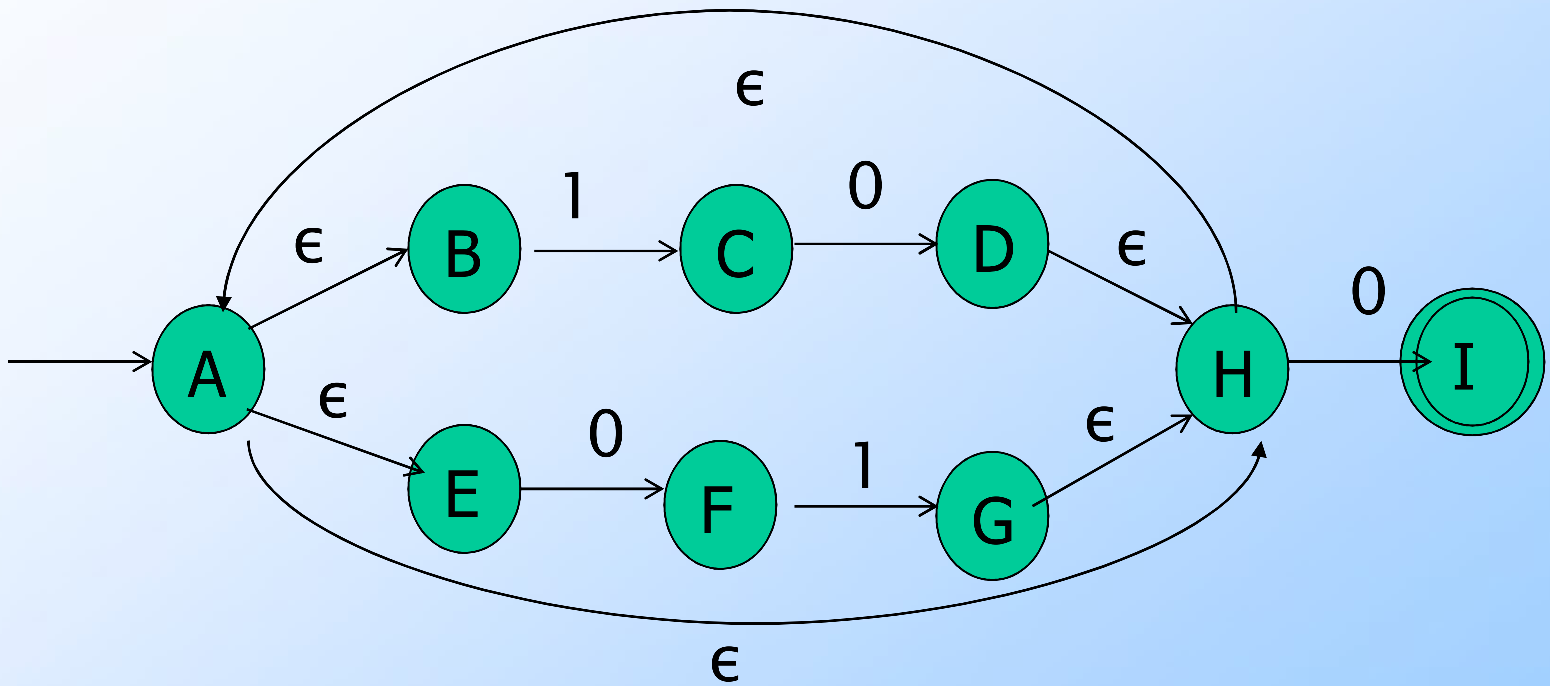
Or Simply,



Finite automata for RE= (a+b)*

RE to ϵ-NFA

- **Example:**
  - **Given RE as:** (10+01)*0 , the **ϵ-NFA** for the language of this RE is



RE to ϵ-NFA

- **Exercise**
  - **Construct the ϵ-NFA accepting languages described by following RE.**
    - (1+110)*0
    - (1+10+110)* 0
    - 1(01+10)*+0(11+10)*
    - 1(1+10)*+10(0+01)*
    - (010 + 00)*(10)*

DFA to RE

- In order to find out a regular expression of a Finite Automaton, we use Arden's Theorem along with the properties of regular expressions.
- ***Statement*** :
  - Let **P** and **Q** be two regular expressions.
  - If **P** does not contain empty string, then

    R = Q + RP has a unique solution that is **R = QP***
- **Proof**:-
  - R = Q + (Q + RP)P [After putting the value R = Q + RP]
  - $= Q + QP + RP^2$
  - When we put the value of **R** recursively again and again, we get the following equation −
  - $R = Q + QP + QP^2 + QP^3$..... Up to infinity
  - $R = Q (\varepsilon + P + P^2 + P^3 +$ …. Up to infinity)
  - R = QP* [As P* represents (ε + P + P2 + P3 + ….)]
- Hence, proved.

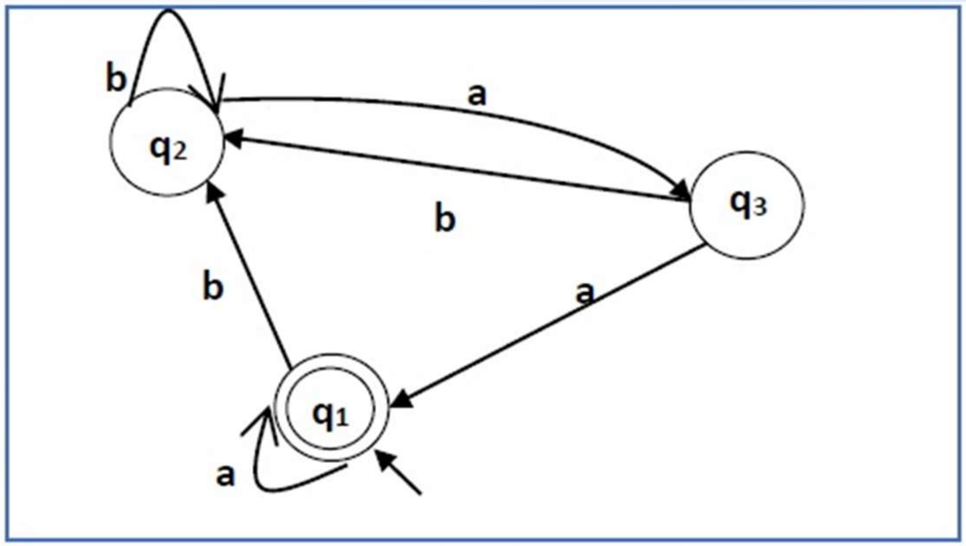
DFA to RE

Applying Arden's Theorem to convert FA to RE

- Method:
  - **Step 1** – Create equations as the following form for all the states of the DFA having n states with initial state q_1.
  - $q_1 = q_1R_{11} + q_2R_{21} + \dots + q_nR_{n1} + \varepsilon$
  - $q_2 = q_1R_{12} + q_2R_{22} + \dots + q_nR_{n2}$
  - ……………………………
  - $q_n = q_1R_{1n} + q_2R_{2n} + \dots + q_nR_{nn}$
  - $\mathbf{R_{ij}}$ represents the set of labels of edges from $\mathbf{q_i}$ to $\mathbf{q_j}$, if no such edge exists, then $\mathbf{R_{ij}} = \emptyset$
- **Step 2** – Solve these equations to get the equation for the final state in terms of $\mathbf{R_{ij}}$.

DFA-to-RE

- Construct a regular expression corresponding to the DFA given below



Solution :–

- Here the initial state and final state is $\mathbf{q_1}$.
- The equations for the three states q1, q2, and q3 are as follows –
- $q_1 = q_1a + q_3a + \varepsilon$ (ε move is because q1 is the initial state)
- $q_2 = q_1b + q_2b + q_3b$
- $q_3 = q_2a$

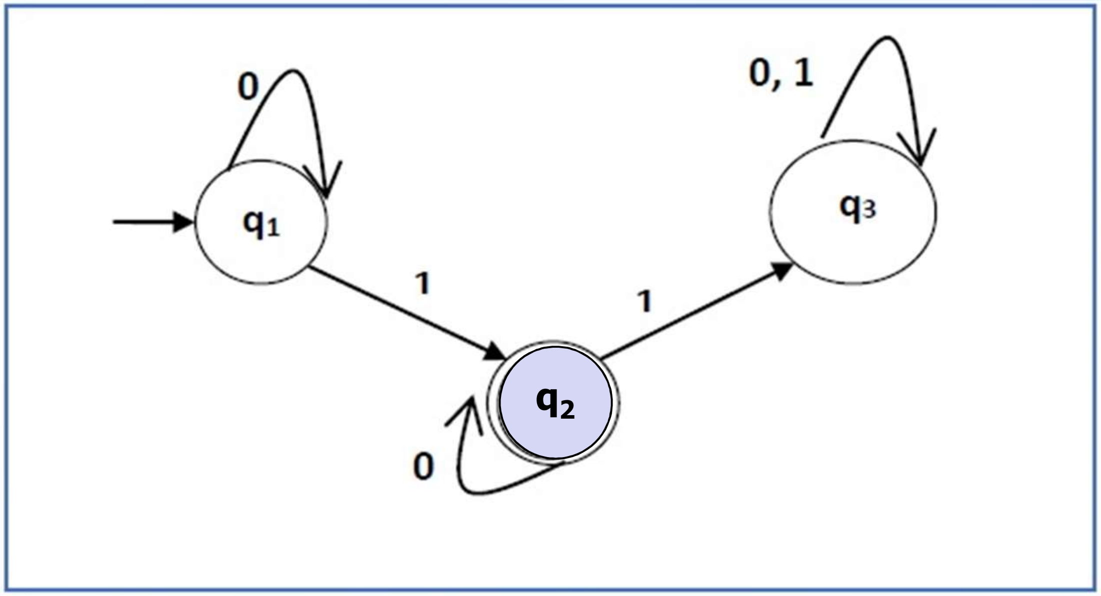
DFA-to-RE : Arden's Rule

Now, we will solve these three equations –

- $\mathbf{q_2 = q_1b + q_2b + q_3b}$
- $= q_1b + q_2b + (q_2a)b$ (Substituting value of q_3)
- $\mathbf{q_2} = q_1b + q_2(b + ab)$ (Arden rule, R=Q+RP ➔ R=QP*)
- $= q_1b\ (b + ab)^*$ (Applying Arden's Theorem)
- Now from, $\mathbf{q_1 = q_1a + q_3a + \varepsilon}$
- $= q_1a + \mathbf{q_2a}a + \varepsilon$ (Substituting value of q_3)
- $= q_1a + q_1b(b + ab^*)aa + \varepsilon$ (Substituting value of q_2)
- $\mathbf{q_1 = \varepsilon + q_1(a + b(b + ab)^*aa)}$
- $= \varepsilon\ (a + b(b + ab)^*aa)^*$ (Applying Arden's Rule)
- $= (a + b(b + ab)^*aa)^*$ (Remove **ε after concatinating)**
- Hence, the regular expression is $(a + b(b + ab)^*aa)^*$.

DFA-to-RE: Arden's Rule Example 2

- Construct a regular expression corresponding to the DFA given below



Solution :–

- Here the initial state is **q_1 and final state is q_2**.
- The equations for the three states q1, q2, and q3 are as follows –
- $q_1 = q_1 0 + \varepsilon$ (ε move is because q1 is the initial state)
- $q_2 = q_1 1 + q_2 0$
- $q_3 = q_2 1 + q_3 0 + q_3 1$

DFA-to-RE : Arden's Rule

Solution :–

- Here the initial state is **q_1 and final state is q_2**.
- The equations for the three states q1, q2, and q3 are as follows –
- $q_1 = q_1 0 + \varepsilon$ (ε move is because q1 is the initial state)
- $q_2 = q_1 1 + q_2 0$
- $q_3 = q_2 1 + q_3 0 + q_3 1$

Now, we will solve these three equations –

- **$q_1 = \varepsilon + q_1 0$**
  - $q_1 = \varepsilon 0^*$ [As per Arden's Theorem $R=Q+RP \Rightarrow R=QP^*$]
  - So, $q_1 = 0^*$ [As, $\varepsilon R = R$]
- Now, $q_2 = q_1 1 + q_2 0$
  - $q_2 = 0^*1 + q_2 0$ [Substituting for q_1]
- So, $q_2 = 0^*1(0)^*$ [By Arden's theorem]
- Hence, the regular expression is **0*10*.**