# Regular Expression

-

Hemant Kumar

July 28, 2020

# Outline

# Regular Expression

A language accepted by a finite automata are easily described by expression,called *Regular Expression*.

## Definition

Let $\Sigma$ be an input alphabet, the regular expression over $\Sigma$ can be defined as

(a) $\phi$ is empty set and $\varepsilon$ is empty string then its regular expression is $\{\ \}$ & $\{\ \varepsilon\ \}$ respectively i.e.
$\phi \longrightarrow \{\ \}$
$\varepsilon \longrightarrow \{\ \varepsilon\ \}$
$a \in \Sigma \longrightarrow \{\ a\ \}$ i.e. $\{$*Primitive regular expression* $\}$

(b) Let $r_1$ and $r_2$ be the primitive regular expressions and apply any of the operators such as union$(+)$, concatenation$(\cdot)$, and kleen closure(*) ;
$r_1 + r_2$, $r_1 \cdot r_2$, $r_1*$ is regular expressions.

(c) We can apply any number of time and we get regular expressions

# Example: Let $\Sigma = \{a, b\}$

1. Language accepting all combinition of a's over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{\varepsilon, a, aa, aaa, \ldots\} = a^*$

# Example: Let $\Sigma = \{a, b\}$

1. Language accepting all combinition of a's over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{\varepsilon, a, aa, aaa, \ldots\} = a^*$
2. Language accepting all combinition of a's except the null string over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{a, aa, aaa, \ldots\} = a^+$

# Example: Let $\Sigma = \{a, b\}$

1. Language accepting all combinition of a's over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{\varepsilon, a, aa, aaa, \ldots\} = a^*$
2. Language accepting all combinition of a's except the null string over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{a, aa, aaa, \ldots\} = a^+$
3. Language accepting 'a' is atleast one time over the input $\Sigma = \{a\}$.
   Solution: $R.E. = (\varepsilon + a)$

# Example: Let $\Sigma = \{a, b\}$

1. Language accepting all combinition of a's over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{\varepsilon, a, aa, aaa, \ldots\} = a^*$
2. Language accepting all combinition of a's except the null string over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{a, aa, aaa, \ldots\} = a^+$
3. Language accepting 'a' is atleast one time over the input $\Sigma = \{a\}$.
   Solution: $R.E. = (\varepsilon + a)$
4. Language Containing all the strings contains any number of a's and b's over input $\Sigma = \{a, b\}$.
   Solution: $R.E = (a + b)^*$

# Example: Let $\Sigma = \{a, b\}$

1. Language accepting all combinition of a's over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{\varepsilon, a, aa, aaa, \ldots\} = a^*$
2. Language accepting all combinition of a's except the null string over the input $\Sigma = \{a\}$.
   Solution: $R.E. = \{a, aa, aaa, \ldots\} = a^+$
3. Language accepting 'a' is atleast one time over the input $\Sigma = \{a\}$.
   Solution: $R.E. = (\varepsilon + a)$
4. Language Containing all the strings contains any number of a's and b's over input $\Sigma = \{a, b\}$.
   Solution: $R.E = (a + b)^*$
5. Language Containing all the strings contains any number of a's and b's except the null string over input $\Sigma = \{a, b\}$.
   Solution: $R.E = (a + b)^+$

## Example: Let $\Sigma = \{a, b\}$

1. Length of the string exactly two.
   $L_1 = \{aa, ab, ba, bb\}$
   Here comma represent union (i.e $+$) so
   $R.E. = aa + ab + ba + bb = a(a + b) + b(a + b) = (a + b)(a + b)$

## Example: Let $\Sigma = \{a, b\}$

1. Length of the string exactly two.
   $L_1 = \{aa, ab, ba, bb\}$
   Here comma represent union (i.e $+$) so
   $R.E. = aa + ab + ba + bb = a(a + b) + b(a + b) = (a + b)(a + b)$
2. Length of the string atleast two.
   $L_1 = \{aa, ab, ba, bb, aaa, aba, bbb, \ldots\}$
   $R.E. = (a + b)(a + b)(a + b)^*$

# Example: Let $\Sigma = \{a, b\}$

1. Length of the string exactly two.
   $L_1 = \{aa, ab, ba, bb\}$
   Here comma represent union (i.e $+$) so
   $R.E. = aa + ab + ba + bb = a(a + b) + b(a + b) = (a + b)(a + b)$
2. Length of the string atleast two.
   $L_1 = \{aa, ab, ba, bb, aaa, aba, bbb, \ldots\}$
   $R.E. = (a + b)(a + b)(a + b)^*$
3. Length of the string atmost two.
   $L_1 = \{\varepsilon, a, b, aa, ab, ba, bb\}$
   $R.E. = (a + b + \varepsilon)(a + b + \varepsilon)$

# Example: Let $\Sigma = \{a, b\}$

1. Length of the string exactly two.
   $L_1 = \{aa, ab, ba, bb\}$
   Here comma represent union (i.e $+$) so
   $R.E. = aa + ab + ba + bb = a(a+b) + b(a+b) = (a+b)(a+b)$
2. Length of the string atleast two.
   $L_1 = \{aa, ab, ba, bb, aaa, aba, bbb, \ldots\}$
   $R.E. = (a+b)(a+b)(a+b)^*$
3. Length of the string atmost two.
   $L_1 = \{\varepsilon, a, b, aa, ab, ba, bb\}$
   $R.E. = (a+b+\varepsilon)(a+b+\varepsilon)$
4. Length of the string are even.
   $L_1 = \{\varepsilon, aa, ab, ba, bb, aaaa, aaba, abbb, \ldots\}$
   $R.E. = (a+b)^{2n} = ((a+b)(a+b))^*$

## Example: Let $\Sigma = \{a, b\}$

1. Length of the string exactly two.
   $L_1 = \{aa, ab, ba, bb\}$
   Here comma represent union (i.e $+$) so
   $R.E. = aa + ab + ba + bb = a(a + b) + b(a + b) = (a + b)(a + b)$
2. Length of the string atleast two.
   $L_1 = \{aa, ab, ba, bb, aaa, aba, bbb, \dots\}$
   $R.E. = (a + b)(a + b)(a + b)^*$
3. Length of the string atmost two.
   $L_1 = \{\varepsilon, a, b, aa, ab, ba, bb\}$
   $R.E. = (a + b + \varepsilon)(a + b + \varepsilon)$
4. Length of the string are even.
   $L_1 = \{\varepsilon, aa, ab, ba, bb, aaaa, aaba, abbb, \dots\}$
   $R.E. = (a + b)^{2n} = ((a + b)(a + b))^*$
5. Length of the string is odd.
   $L_1 = \{a, b, aaa, aab, bba, abb, aba, bbb, \dots\}$
   $R.E. = ((a + b)(a + b))^*(a + b)$

# Identities of Regular Expression

Let R be the regular expression

$\Rightarrow \phi + R = R + \phi = R$

$\Rightarrow \phi \cdot R = R \cdot \phi = \phi$

$\Rightarrow \varepsilon \cdot R = R \cdot \varepsilon = R$

$\Rightarrow \varepsilon^* = \varepsilon$

$\Rightarrow \phi^* = \varepsilon$

$\Rightarrow \varepsilon + RR^* = R^*R + \varepsilon = R^*$

$\Rightarrow (PQ)^*P = P(QP)^*$

Let a and b be the regular expression then

$(a+b)^* = (a^* + b^*)^* = (a^*.b^*)^*$

# Laws for Regular expression

## Commutative Law

Let r and s be a regular expression then

$$r + s = s + r \text{ for union}$$
$$r \cdot s = s \cdot r \text{ for Concatenation}$$

## Associative Law

Let r, s and p be a regular expression then

$$(r + s) + p = r + (s + p) \text{ for union}$$
$$(r \cdot s) \cdot = r \cdot (s \cdot p) \text{ for Concatenation}$$

## Distributive Law

Let r, s and p be a regular expression then

$$p \cdot (r + s) = p \cdot r + p \cdot s = (p + r) \cdot s$$

# Laws for Regular expression

## Idempotent Law

Let r be a regular expression then

$$r + r = r \text{ for union}$$

## Law of closure

Let r be a regular expression then

1. $(r^*)^* = r^*$
2. $\phi^* = \varepsilon$
3. $\varepsilon^* = \varepsilon$
4. $r^* = r^+ + \varepsilon$

# Inequlatites in Regular Expression

Let r,s and p be a regular expression then

1. $(r.s^*) \neq (r.s)^*$
2. $(r+s)^* \neq r+s^*$
3. $(r+s)^* \neq r^*+s^*$
4. $(r+s)^* \neq r^*.s^*$

# Arden's Theorem

## Arden's Theorem

Let $P$ and $Q$ be the two regular expression over the input alphabet $\Sigma$, if P does not contain $\varepsilon$ then the equation

$$R = Q + RP; \qquad \textit{where } R \textit{ is regular expression}$$

has a unique solution, i.e.

$$R = QP^*$$

**Example:** Prove that :

$$(1 + 00^*1) + (1 + 00^*1)(0 + 10^*1)^*(0 + 10^*1) = 0^*1(0 + 10^*1)^*$$

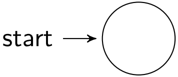
**Solution:** Let P $=(1 + 00^*1)$ and Q$=(0 + 10^*1)$

$$\begin{aligned} L.H.S &= (1 + 00^*1) + (1 + 00^*1)(0 + 10^*1)^*(0 + 10^*1) \\ &= P + PS^*S = P(\varepsilon + S^*S) = PS^* \qquad \{Replace\ the\ value\ with\ P\ and\ Q\} \\ &= (1 + 00^*1) \cdot (0 + 10^*1)^* \qquad \{Substitute\ the\ P\ and\ Q\} \\ &= (\varepsilon + 00^*)1 \cdot (0 + 10^*1)^* \\ &= 0^*1 \cdot (0 + 10^*1)^* \\ &= R.H.S \end{aligned}$$

# Regular Expression to Finite Automata

Let a and b be the regular expression. Some basic finite automata for primitive regular expression.

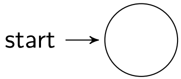
1. *Finite automata for empty string* ($\phi$)



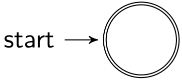
# Regular Expression to Finite Automata

Let a and b be the regular expression. Some basic finite automata for primitive regular expression.

1. *Finite automata for empty string* ($\phi$)



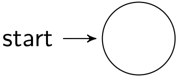

2. *Finite automata for null string* ($\varepsilon$)



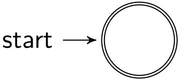
# Regular Expression to Finite Automata

Let a and b be the regular expression. Some basic finite automata for primitive regular expression.

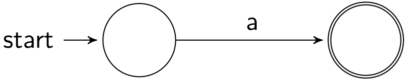
1. *Finite automata for empty string* ($\phi$)




2. *Finite automata for null string* ($\varepsilon$)



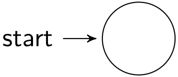

3. *For regular expression* $'a'$



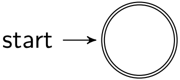
# Regular Expression to Finite Automata

Let a and b be the regular expression. Some basic finite automata for primitive regular expression.

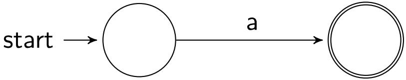
1. *Finite automata for empty string* ($\phi$)



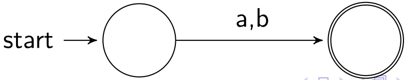

2. *Finite automata for null string* ($\varepsilon$)




3. *For regular expression* $'a'$

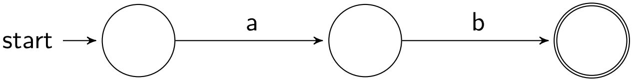



4. *For regular expression* $'a+b'$



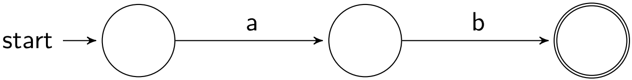
# Regular Expression to Finite Automata

5. *For regular expression* $'a.b'$

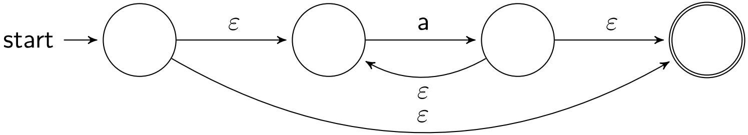


# Regular Expression to Finite Automata

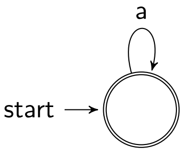
5. *For regular expression $'a.b'$*




6. *For regular expression $a^*$*

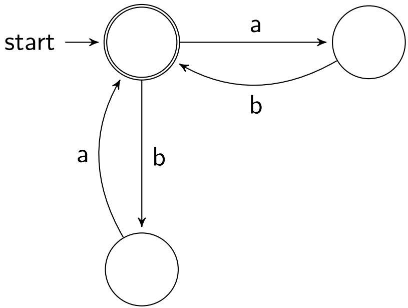



OR



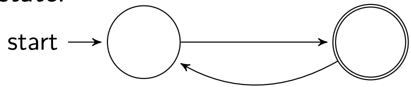
# Regular Expression to Finite Automata

**Example 1.:** Construct finite automata for $(ab + ba)^*$
**Solution:**

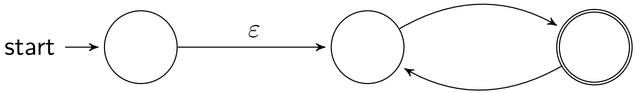


# FA to RE Using State Elemination Method

1. Initial state should not have any incoming edge, if it is then initial state.

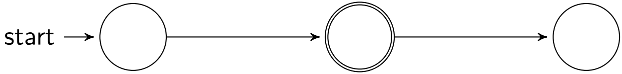



convert to

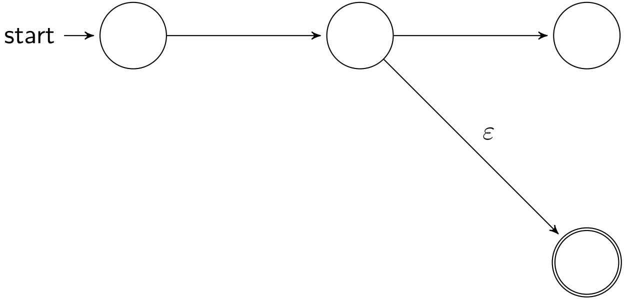


# FA to RE Using State Elemination Method

2. In final state should not have any outgoing edges. If it is then create new final state with $\varepsilon$ and make it non final state.
   a. If one Final State



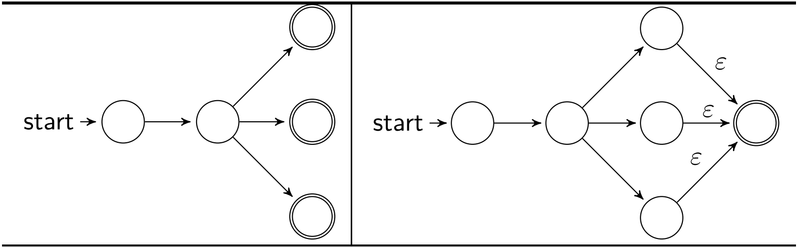

convert into



# FA to RE Using State Elemination Method

2. In final state should not have any outgoing edges. If it is then create new final state with $\varepsilon$ and make it non final state.

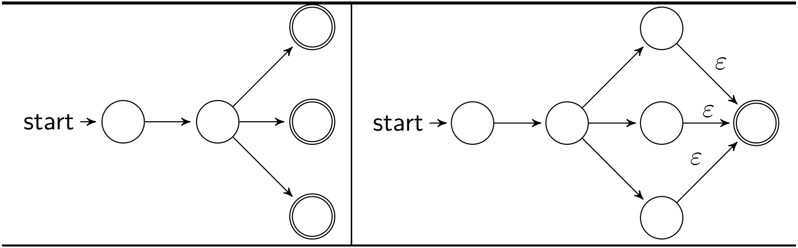
   b. If DFA contain more than one final state then we add one new final state with $\varepsilon$ moves and make final state to non-final state.



# FA to RE Using State Elemination Method

2. In final state should not have any outgoing edges. If it is then create new final state with $\varepsilon$ and make it non final state.

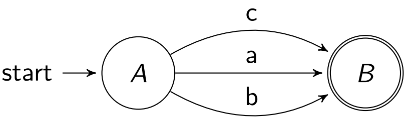
   b. If DFA contain more than one final state then we add one new final state with $\varepsilon$ moves and make final state to non-final state.



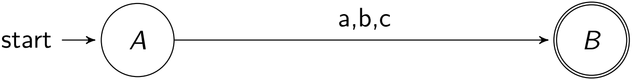

3. After that, other than the final and initial state, eliminate the remaining state one by one.

# Examples

**Example 1.** Findout the regular expression for the given finite automata.



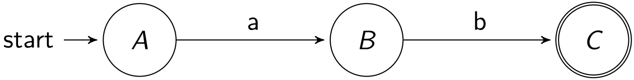

**Solution:** first to simplify the FA



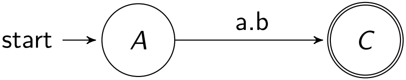

Here comma represent the union so the regular expression is
$R.E. = a + b + c$

# Examples

**Example 2.** Findout the regular expression for the given finite automata.



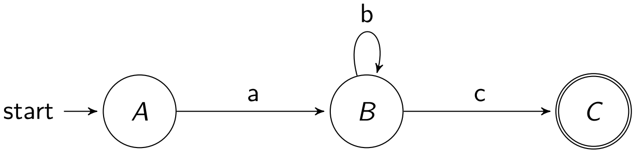

**Solution:** first to simplify the FA



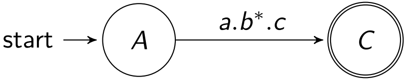

Here dot represent the concatenation so the regular expression is
$R.E. = a.b$

# Examples

**Example 3.** Findout the regular expression for the given finite automata.



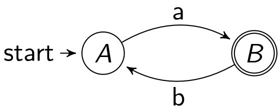

**Solution:** first to simplify the FA using eliminate state B



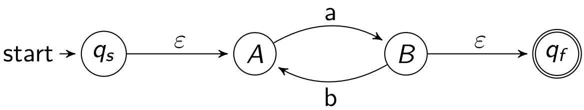

so the regular expression is
$R.E. = a.b^*.c$

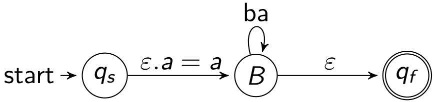
# Examples

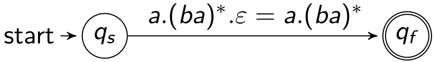
**Example 4.** Findout the regular expression for the given finite automata.




**Solution:** Create new state which is initial state $q_s$ and final state $q_f$ because initial state A have incoming edge and final state B have outgoing edge




the regular expression is
$R.E. = a.(ba)^*$