Part-2

TOC Notes

Regular Expression and Languages

Notes by :- @jpwebdevelopers

→ Regular Expression is a method to represent a language.
(Regular language को represent करता)

* Regular Language :- The language accepted by some regular expression are referred to as Regular Language.
→ FSM accepts करता
→ Type-3 Grammar

→ It can also be described as a sequence of pattern that defines a string.

Regular Expression is a valid expression of string with operators like :-

→ Kleen's closure a^*
→ Positive kleen's closure a^+
→ concatenation $a \cdot b$
→ union $a+b$

over an alphabet of any number of symbol.

Primitive Regular Expression
ϕ
ϵ / λ
↳ symbol.

Derived / Non primitive regular expression
Combinations derived from Primitivies.
$(a+b)^*$

Imp:- A Regular Expression is said to be valid, if it is derived from primitive Regular Expression like union, concatenation, kleen's closure, positive closure etc.

$$(a+b)^* = \Sigma^*$$

Language → finite, Infinite

finite: Regular ✓, FA ✓, RE ✓

Regular Expression	Regular Language	
✓ $r = \phi$	$L(r) = \{\} / \phi$	(No string)
✓ $r = \epsilon$	$L(r) = \{\epsilon\}$	(Length 0)
✓ $r = a$	$L(r) = \{a \mid a \in \Sigma\}$	
✓ $r = a+b$	$L(r) = \{a, b \mid a, b \in \Sigma\}$	(Length 1)
✓ $r = (a+b)(a+b)$	$L(r) = \{aa, ab, ba, bb\}$	(" 2)
✓ $r = (a+b)(a+b)(a+b)$	$L(r) = \{aaa, aab, aba \ldots\}$	(" 3) finite
✓ $r = a \cdot b$	$L(r) = \{ab\}$	

(Combination ले सकते हैं
But एक ही बार, क्योंकि Power नहीं है)

✓ $r = (ab+a) \cdot b$ $\qquad L(r) = \{abb, ab\}$

$\begin{pmatrix} ab \cdot b \\ a \cdot b \end{pmatrix}$

✓ $r = a^*$ $\qquad L(r) = \{\epsilon, a, aa, \ldots\}$

$r = (a+b)^+$ $\qquad L(r) = \{a^*, b^*\}$

$r = \epsilon$ $\quad L(r) = \{\epsilon\}$

$r = \phi$ $\quad L(r) = \{\}/\phi$

RE

Reg. Language

$a \atop \epsilon$ $\leftarrow a+\epsilon \rightarrow a/\epsilon$ ✓

$a/\phi \leftarrow a+\phi = a$ ✓

एक ऐसी string जो 0 length की है।

✓ $r = (a+b)^*$ $\quad L(r) = \Sigma^* = \{a^*, b^*, \Sigma\}$

↳ ab
↳ ba
↳ Σ.

} कोई order नहीं

$\boxed{(ab)^*}$
↳ इन ही symbol के कितने भी
→ ab, abab, ababab, ...

✓ $\boxed{r = (a+ba).(b+a)}$ $\quad L(r) = \{ab, aa, bab, baa\}$

a → b, a
ba → b, a

✓ $r = (a+\epsilon).(b+\phi)$ $\quad L(r) = \{ab, b\}$

→ $(a+\epsilon).b$

a → b
ε → b

Imp ✓ — Imp identities

same
$r = \Sigma^* \rightarrow \{\Sigma\}$
$r = \Sigma^+ \rightarrow \{\Sigma\}$
$r = \phi^* \rightarrow \{\Sigma\}$

different
$r = \phi^+ = \{\}$

$r^*.r^+ = r^+$

$(\epsilon + r^+).r^+$

$(r^*)^* = r^*$

$(r^+)^* = r^*$

Imp

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

[NOTE] :-

- Multiple regular expression can generate same language but one regular expression can not generate multiple language.

- Two regular expressions are equivalent if languages generated by them are same. For example:- $(a+b^*)^*$ and $(a+b)^*$ generate same language. Every string which is generated by $(a+b^*)^*$ is also generated by $(a+b)^*$ and vice-versa.

→ ϵ is a Regular Expression indicates the language containing an empty string.
$(L(\epsilon) = \{\epsilon\})$

→ ϕ is Regular Expression denotes an empty language.
$(L(\phi) = \{\})$

→ x is Regular Exp. , where $L = \{x\}$

→ R^* is a Regular Exp. Corresponding to the language $L(R^*)$ where $L(R^*) = (L(R))^*$

→ The operation '$*$' has highest precedence over concatenation which has precedence over $(+)$
$(a+(b(c^*))) = a+bc^*$