Date : September 10, 2020 AFLL

Last class :-

Equivalence of RE and FA

1) RE → FA (Thompson construction method)

2) FA → RE (State Elimination method)

Today's class :-

✓ 1) Solve one more question on FA → RE

✓ 2) Equivalence of two RE

✓ 3) Regular Grammar

✓ 3.1) Construction

3.2) Equivalence of RG and FA

FA → RG

RG → FA is pending !

Equivalence of two Regex

Lang

finite infinite

$R_1 = R_2$?

Equivalent $L(R_1) = L(R_2)$

Formal way

Thompson const

subset-const

Table filling

R_1 → NFA_1 → DFA_1 → Minimal DFA

R_2 → NFA_2 → DFA_2 → Minimal DFA

Informal way, faster way

$R_1 \neq R_2$?

find out a string recognized only by one of the regex

→ observe your regex

Example :- formal way

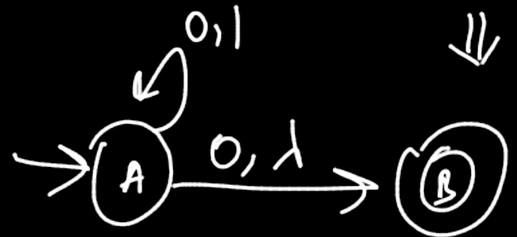
$R_1 = (0+1)^* (0+\lambda)$

$= (0+1)^*$

Same

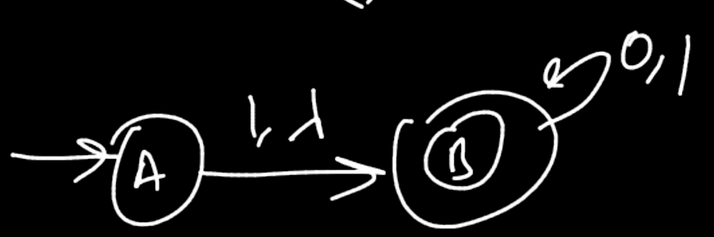
$R_2 = (1+\lambda)(1+0)^*$

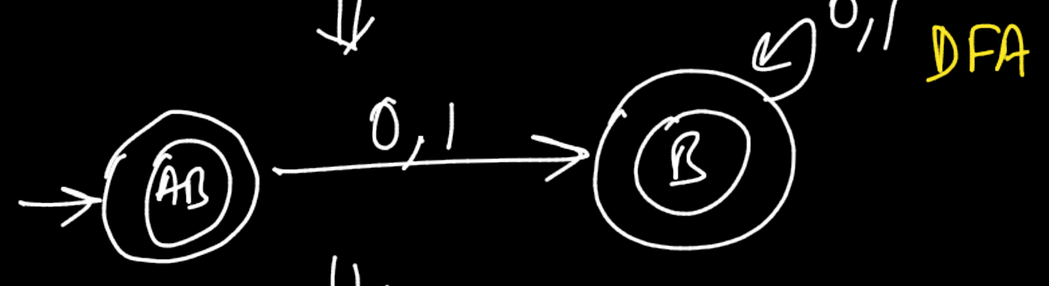
$(0+1)^*$



Equivalent

(Same)





Example :

$R_1 = (0+\lambda)\,(11^*0)^*\,(1+\lambda)$

$R_2 = (1+\lambda)\,(011^*)\,(0+\lambda)$

Not equivalent

$110 \rightarrow R_1$

$\rightarrow \times R_2$

$011 \rightarrow \times R_1$

$\rightarrow R_2$

Minimal String R_1

$\{\lambda\}$

Minimal String R_2

$\{011\}$

Regular Grammar

Grammar - captures syntax

Basis of all the Programming lang (PL's

:, *, +

⇒ if (a > b) { a = 0; }

Cond → Keyword - - - - - -

(terminal)

(Input Symbols in your Σ)

Σ = { ASCII Char set }

Formal Definition

Grammar is a 4 tuple

$G = (V, T, P, S)$

V = Set of Variables / Non-Terminals (expanded)

A, B, C, Cond,

$S \in V$

T = Terminals (Σ)

P = Set of production rules $\alpha \rightarrow \beta$

$\alpha, \beta \in (V \cup T)^*$

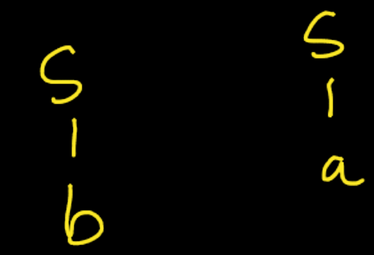
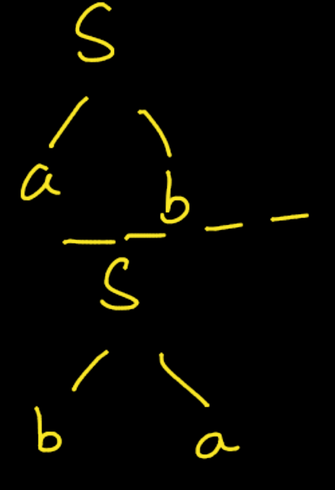
Regular grammar $V \rightarrow \beta$

S = Start Symbol / Variable of your G

Construct Regular Grammar for the given language Descriptions:

	Language	Regex	Regular Grammar	Parse tree
1)	$\{a\}$	a	$S \to a$	S | a
2)	$\{ab\}$	ab	$S \to ab$	S / \\ a b
3)	$\{aaaa\}$	$aaaa$	$S \to \underline{a}\underline{a}\underline{a}\underline{a}$	S / | \ \\ a a a a

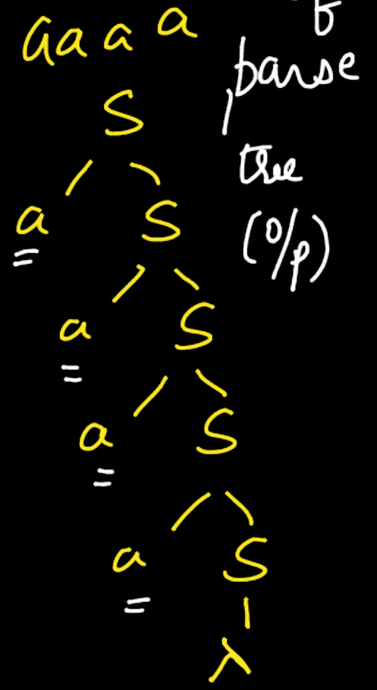
Construct Regular Grammar for the given language descriptions

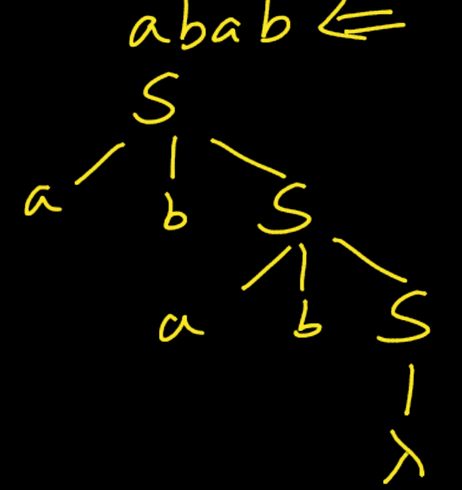
Language	Regex	Regular Grammar	Parse tree
4) $\{a, b\}$	$a + b$	$S \to a$ $S \to b$ or $S \to a \mid b$	
5) $\{ab, ba\}$	$ab + ba$	$S \to ab \mid ba$ or $S \to ab$ $S \to ba$	

Construct Regular Grammar for the given language descriptions:

Language	Regex	Regular Grammar
6) $\{\lambda, a, aa, aaa \ldots\ldots\}$	a^*	$S \to aS \mid \lambda$ or $S \to aS$, $S \to \lambda$ or $S \to Sa \mid \lambda$
7) $\{\lambda, ab, abab, ababab, \ldots\ldots\}$	$(ab)^*$	$S \to abS \mid \lambda$ or $S \to Sab \mid \lambda$

yield of parse tree (%p)





Construct Regular Grammar for the given language descriptions:

Language	Regex	Regular Grammar
8) $\{w, w \in \{a,b\}^*\}$	$(a+b)^*$	$S \to aS \mid bS \mid \lambda$ ✓

$S \to aS \mid Sb \mid \lambda$

↓

Not the right way

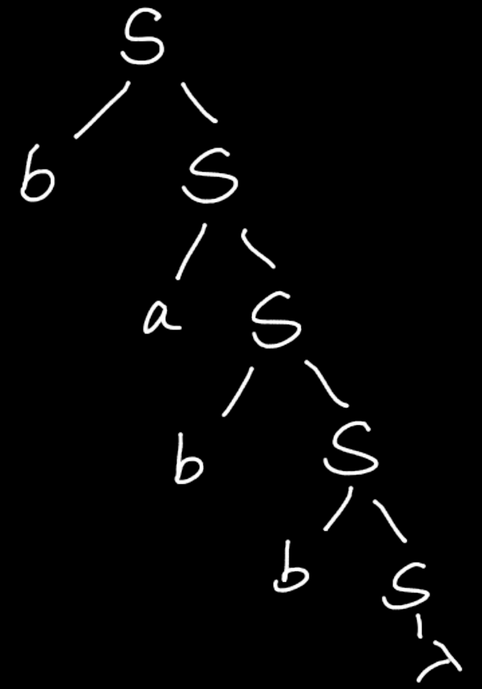
RG

All rules LLG

All rules RLG

You cannot have a mix

$= baabb \in L(G)$



All rules Left linear form

or

$S \to Sa \mid Sb \mid \lambda$

Construct Regular grammar for the given language :-

Language	Regex	Regular Grammar
q) $\{w, w \in \{ab, ba\}^*\}$	$(ab + ba)^*$	$S \rightarrow abS \mid baS \mid \lambda$

Construct Regular grammar for the given language

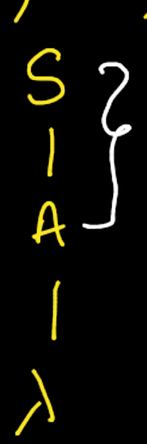
10) $a^* b^*$

$S \to aS \mid A$

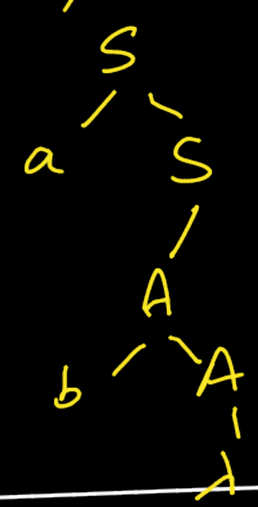
$A \to bA \mid \lambda$

$L = \{ \lambda, a, b, aaa\ldots, bbbb, aa\ldots bb\ldots \}$

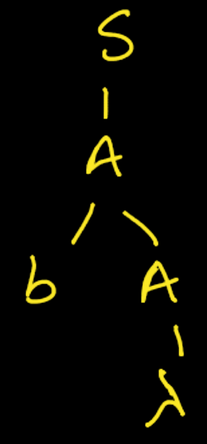
$\omega = \lambda$



$\omega = ab$



$\omega = b$



---

11) $a^n b^m$, $n, m \geq 1$

minimal string

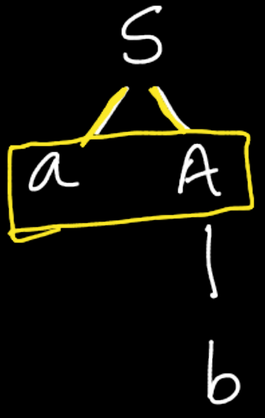
$\{ ab, a\ldots.b\ldots. \}$

G_1

$S \to aS \mid aA$

$A \to bA \mid b$

or



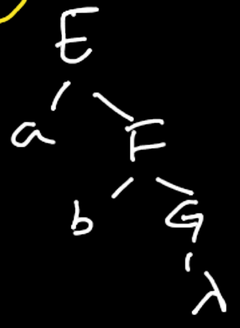
or

G_2

$E \to aF$

$F \to aF \mid bG$

$G \to bG \mid \lambda$



Construct Regular grammar for the following language:

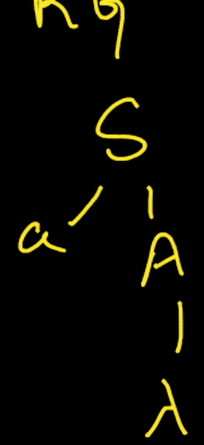
Language over $\{a, b\}^*$ where the string must contain atleast one 'a'.

RE

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

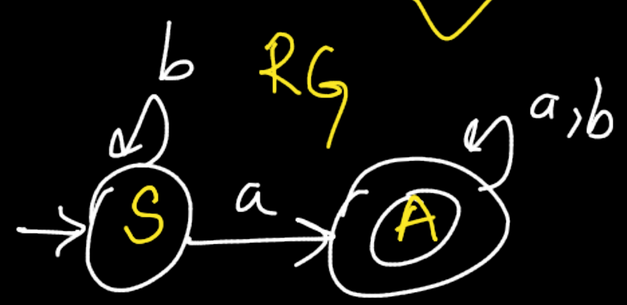
RG

$$S \to bS \mid aA$$
$$A \to aA \mid bA \mid \lambda$$



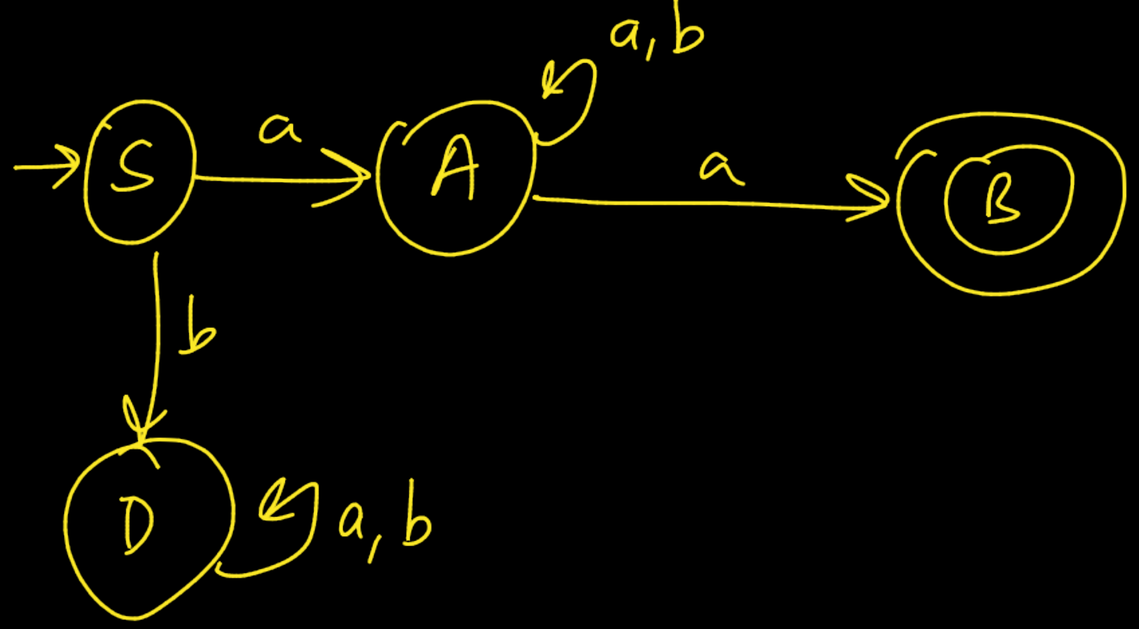
FA → RG

all final states generate λ



$$S \to bS \mid aA$$
$$A \to aA \mid bA \mid \lambda$$

Convert given Finite automata to Regular Grammar :-

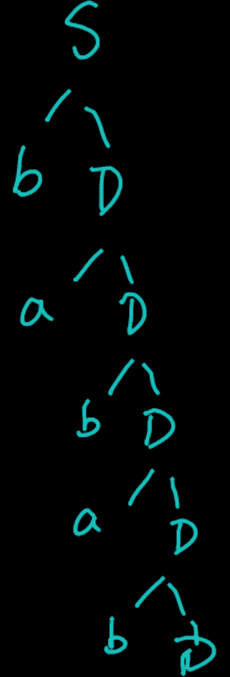


$S \rightarrow aA \mid bD$

$A \rightarrow aA \mid bA \mid aB$

$B \rightarrow \lambda$

$D \rightarrow aD \mid bD$



Ignore the Dead State !

D :- Useless symbol

Although its reachable from S but it doesn't generate anything (end)