Automata Formal Languages & Logic

Preet Kanwal

Department of Computer Science & Engineering

Automata Formal Languages & Logic

Unit 2

Preet Kanwal

Department of Computer Science & Engineering

Regular Grammar

- Formal Definition of a Grammar
- Terminology : Sentence , Sentential Form
- Examples on how to construct a Regular Grammar for a given L

Grammar

- **V - Variables**: expr, var, S, A, B…, w,x,y
- **T - Terminals**: a, b, c …, 0, 1, … id, num
- **P - Production Rules**: α or β = (V U T)*
  $\alpha \rightarrow \beta$
- **S - Start Symbol**: L(G) =
  $\{w \mid S \Rightarrow w, \text{ where } w \in \Sigma^*\}$

Sentence (or a String)

A sentence is made up only of terminals.

Example:

For the Grammar S->aS | a | λ

Sentence (or string) is { a or λ or aa or aaaa)

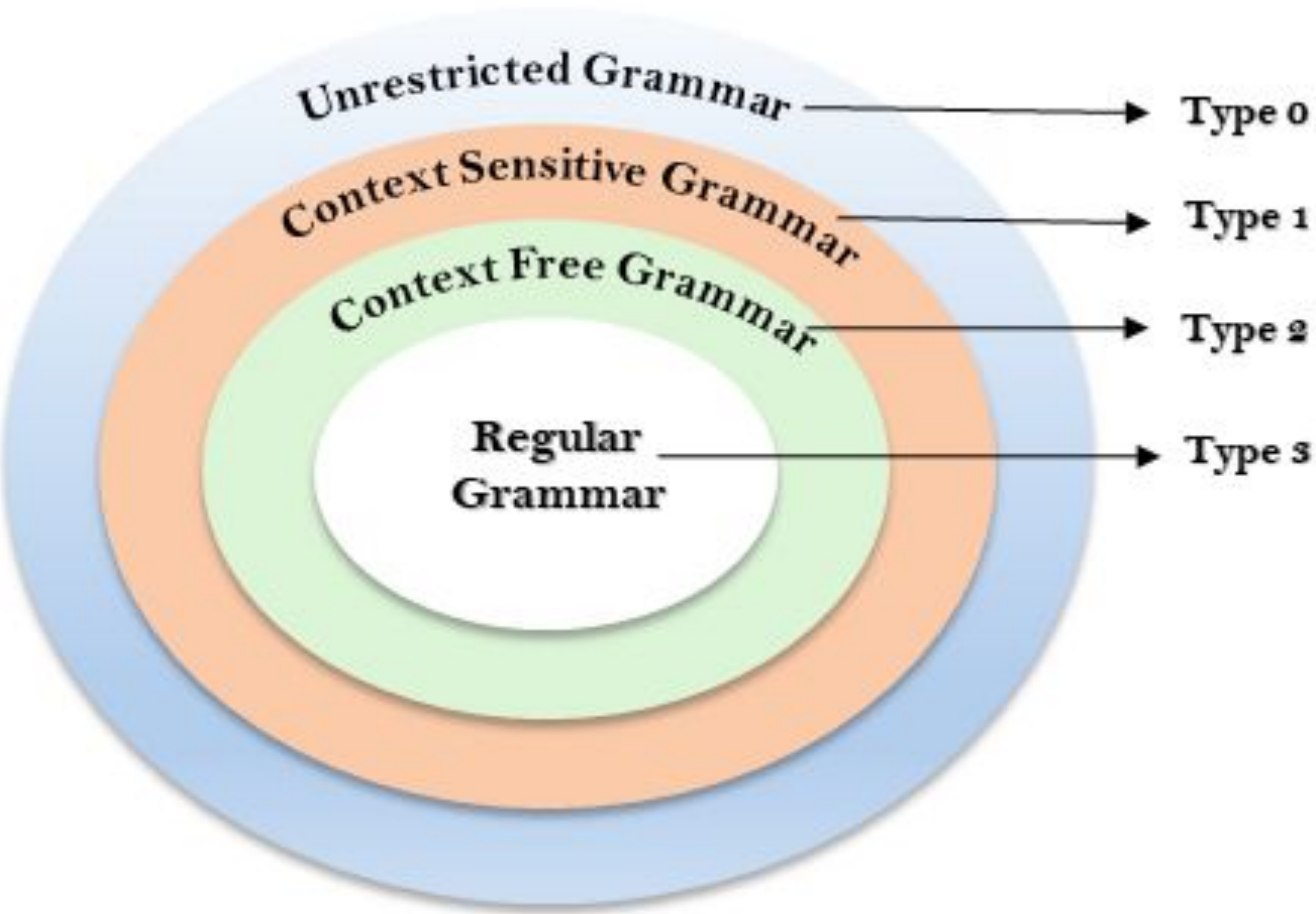
Sentential Form

A sentential form is made up over set of terminals and non-terminals (VUT)*

Example:

For the Grammar S->aS | a |λ

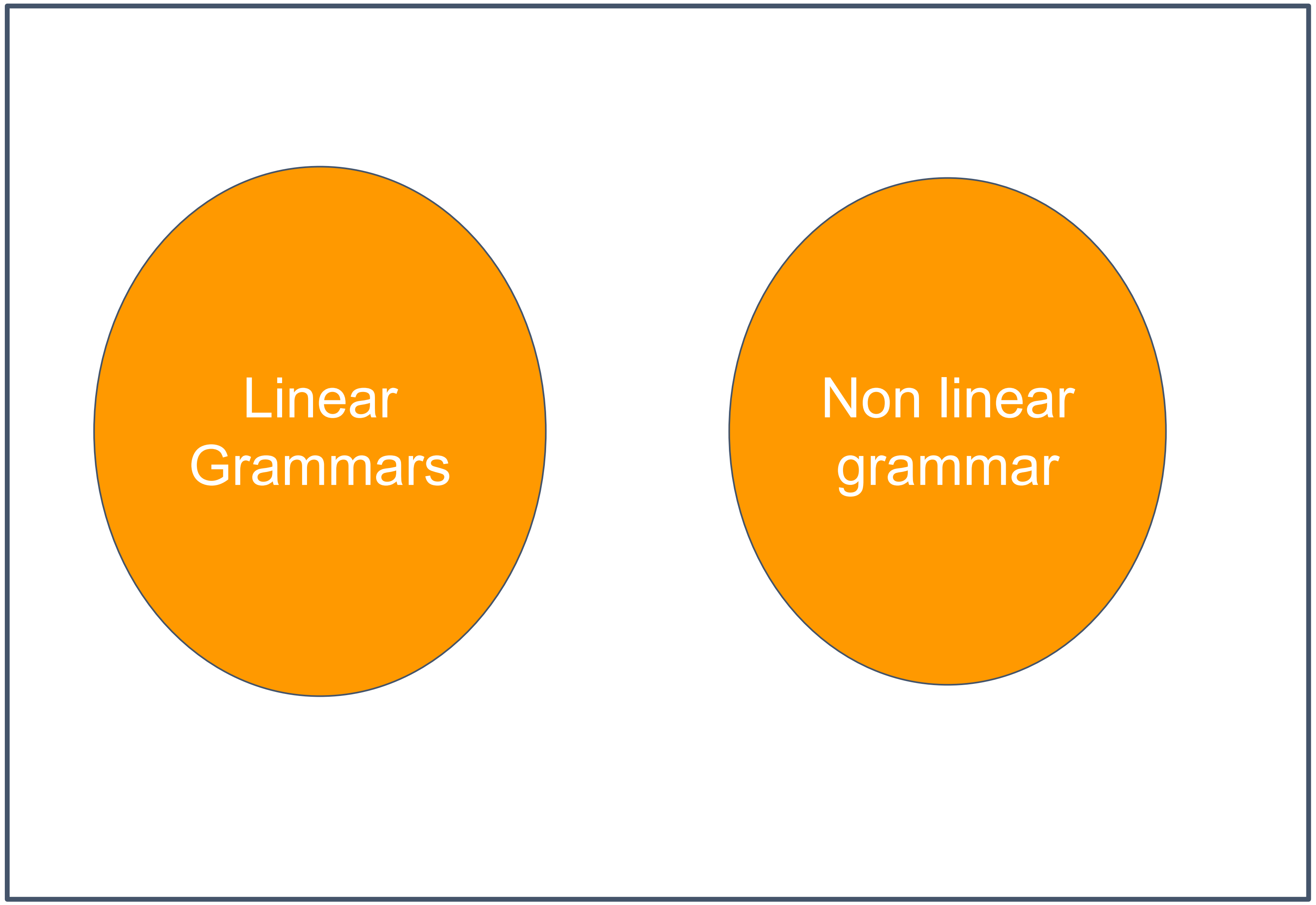
Sentential form is { a or λ or aS or aa or or aaS aaaa)

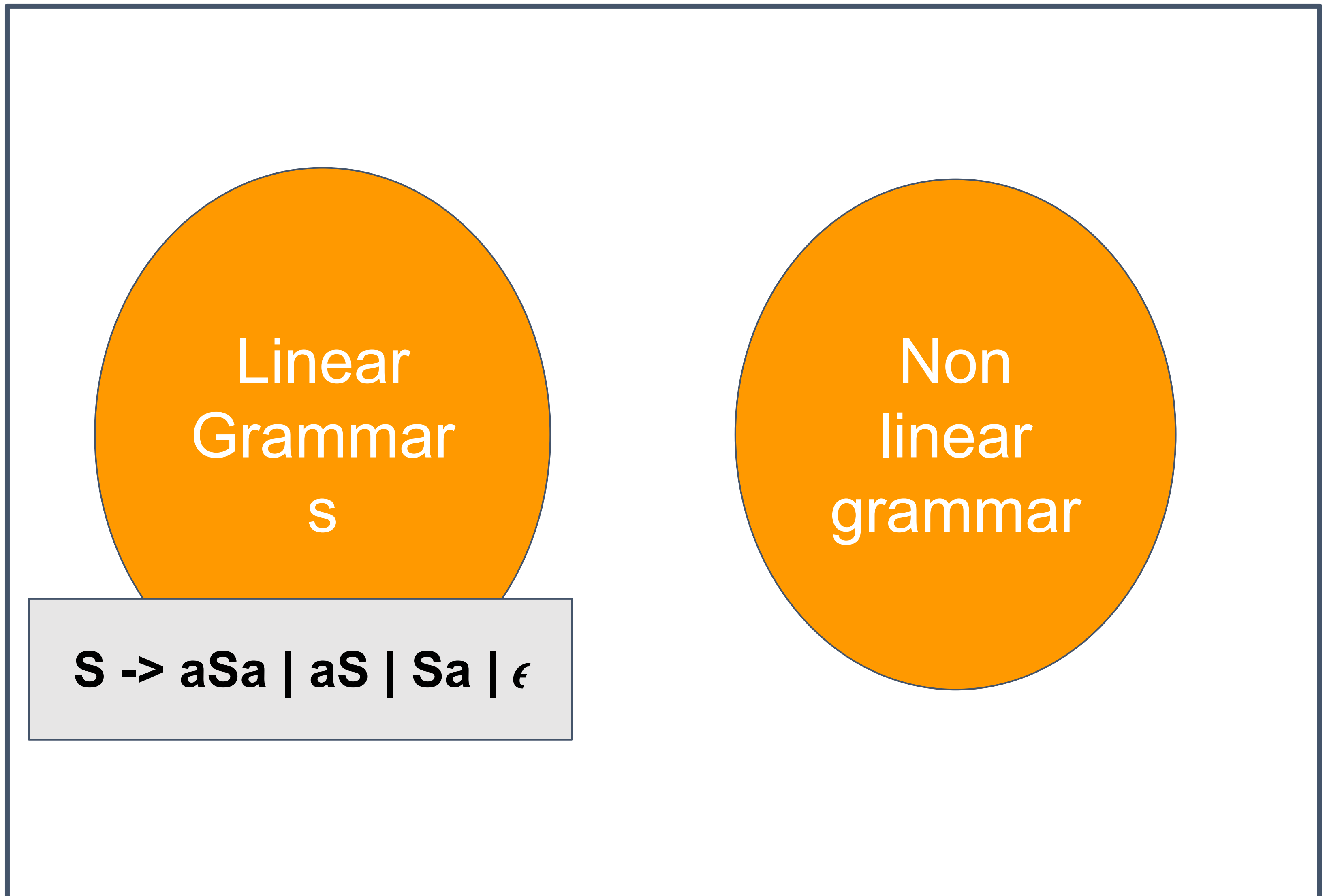


Chomsky Hierarchy

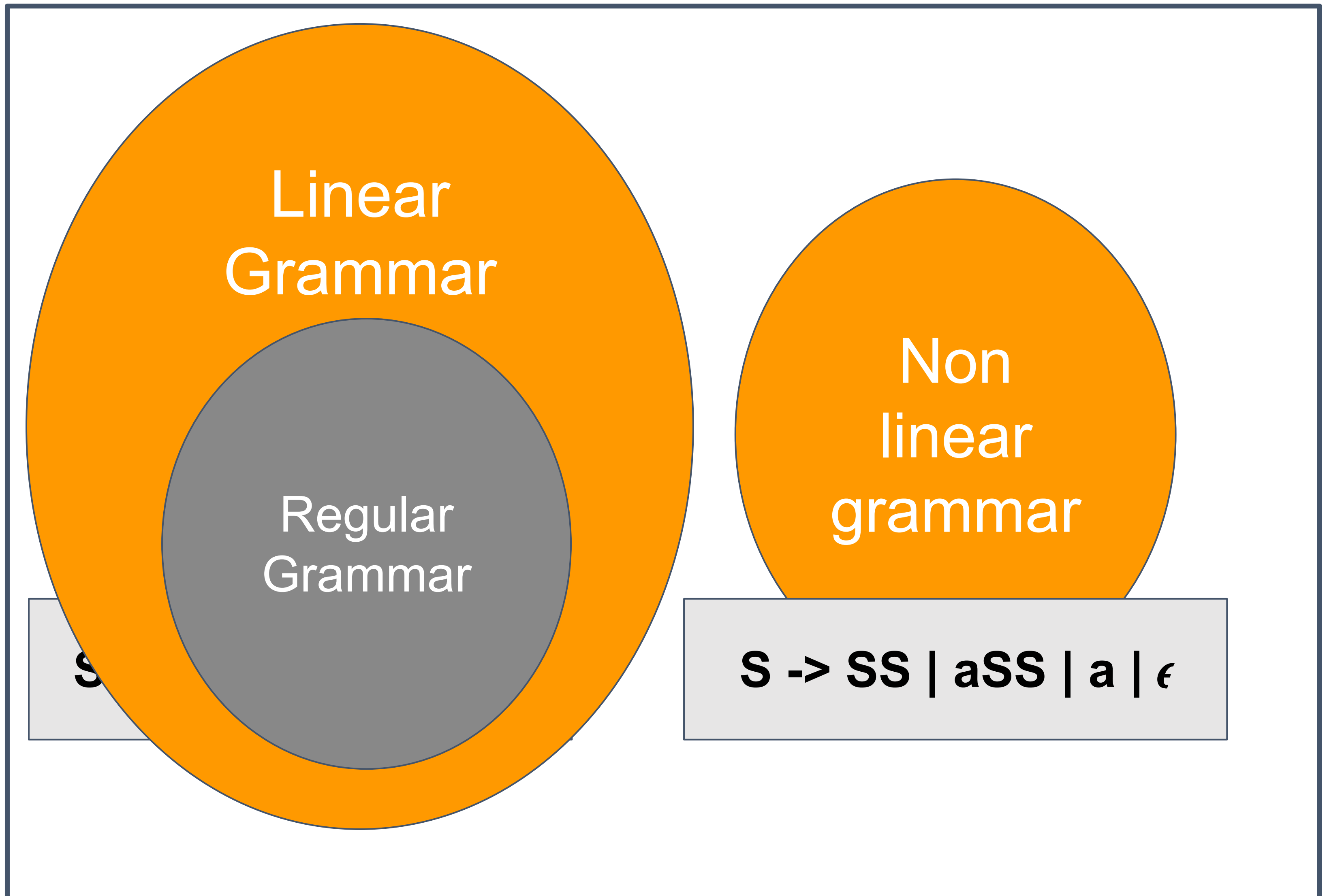
Noam Chomsky

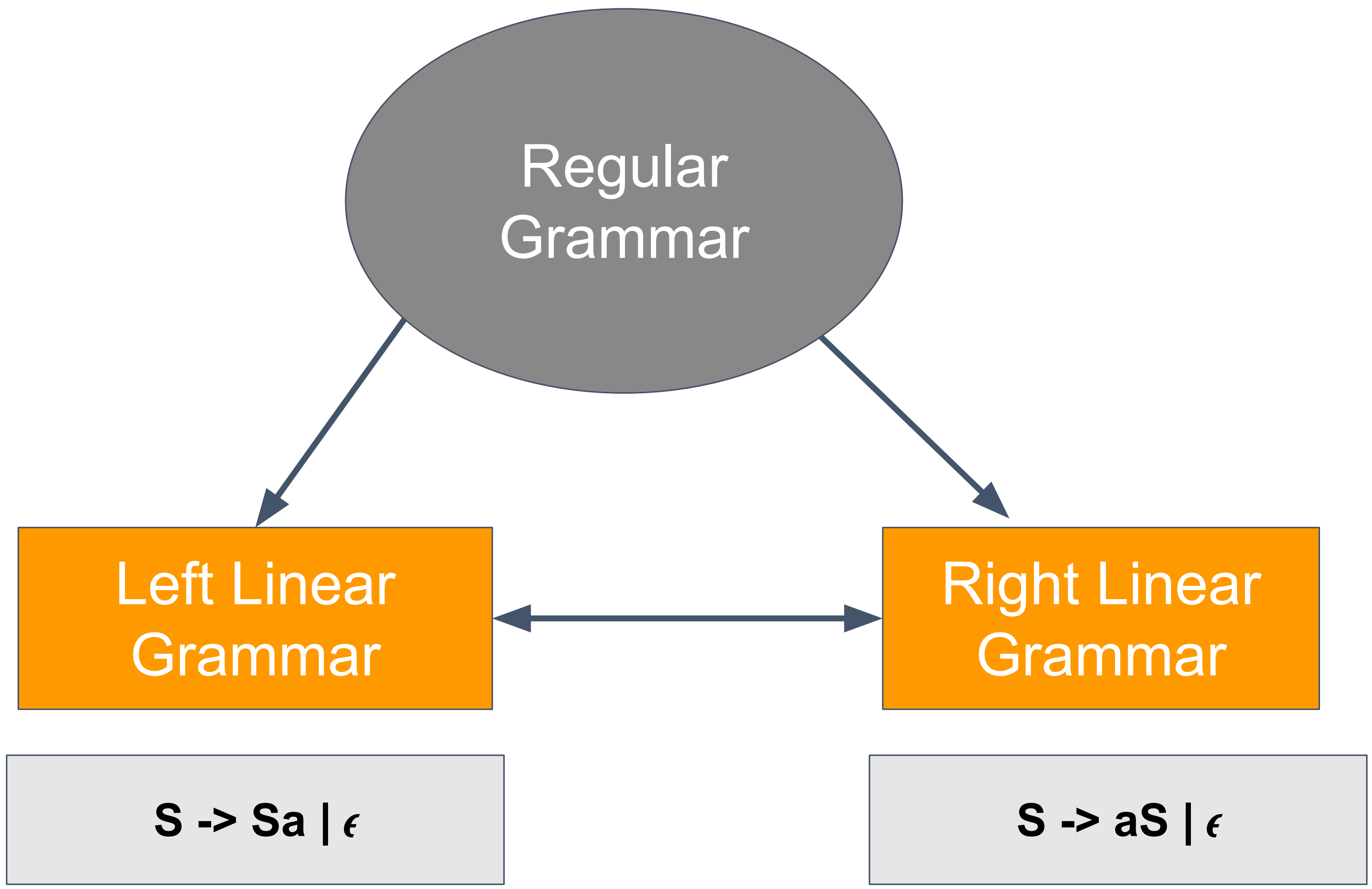
https://www.javatpoint.com/automata-chomsky-hierarchy


Linear
Grammars
Non linear
grammar



Linear Grammar s
Non linear grammar
S -> aSa | aS | Sa | ϵ


Linear Grammar
Regular Grammar
Non linear grammar
S -> SS | aSS | a | ϵ



Regular Grammar
Left Linear Grammar
Right Linear Grammar
S -> Sa | ϵ
S -> aS | ϵ

Construct a regular grammar for a given language description

Construct a regular grammar for the given language

Language	RE	RG
L = {a}	a	S -> a

Construct a regular grammar for the given language

Language	RE	RG
L = {a, b}	a+b	S -> a | b or S -> a S -> b

Construct a regular grammar for the given language

Language	RE	RG
L = {ab}	ab	S -> ab

Construct a regular grammar for the language $L=\{a^n|n>=0\}$

Language	RE	RG
L = {λ,a,aa,aaa,...}	a*	S -> aS|λ

Construct a regular grammar for the language $L=\{a^n|n>=1\}$

Language	RE	RG
L = {a,aa,aaa,...}	a^+	S -> aS | a

Construct a regular grammar for the regular expression (a+b)*

Language	RE	RG
L = {λ,a,b,abb,baaaab,.......} L={any number of 'as and b's}	(a+b)*	S -> aS | bS | λ

PES UNIVERSITY ONLINE

Construct a regular grammar for the regular expression $(a+b)^+$

Language	RE	RG
L = {λ,a,b,abb,baaaab,.......} L={any number of 'as and b's}	$(a+b)^+$	S -> aS | bS | a | b

Construct a regular grammar for the regular expression $(ab)^*$

Language	RE	RG
L = {λ,a,b,ab,ababab,…...} L={any number of ab's}	$(ab)^*$	S -> abS | λ

Construct a regular grammar for the language $L=\{a^m b^n \mid n,m>=0\}$

Language	RE	RG
L={λ,a,b,bb,abb,......} L={any number of a's followed by any number of b's}	a*b*	S -> A|B A:any number of a's B:any number of b's A->aA|λ B->bB|λ

Construct a regular grammar for the given language with even number of a's. $L=\{a^{2n} \mid n>=0\}$

Language	RE	RG
L={λ, aa, aaaa, aaaaaa ….	(aa)*	S -> aaS | λ

Construct a regular grammar for the given language with odd number of a's. $L=\{a^{2n+1} \mid n>=0\}$

Language	RE	RG
L={a, aaa, aaaaa, aaaaaaa ….	(aa)*a	S -> aaS | a

Construct a regular grammar for the given language with number of a's as multiples of 4. $L=\{a^{4n} \mid n>=0\}$

Language	RE	RG
L={λ, aaaa, aaaaaaaa, aaaaaaaaaaaa, ….	(aaaa)*	S -> aaaaS | λ

Convert finite automata to regular grammar for the language to accept at least one ‘a’ over the alphabet Σ={a,b}

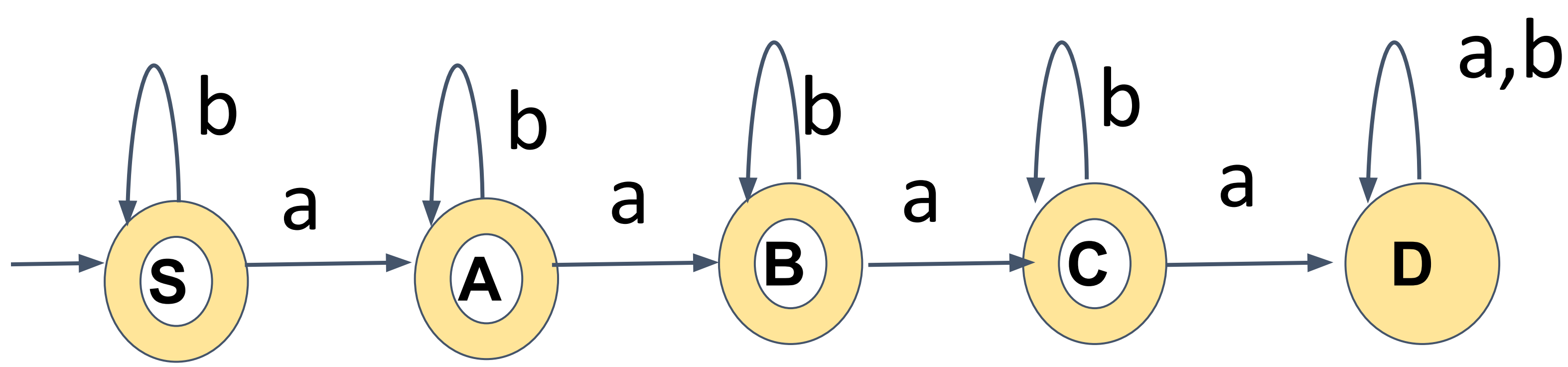
Language	RE	RG
L={a, bb... a, ba any # of a’s & b’s	b* a (a+b)*	S -> bS | aA A -> aA | bA | λ


Regular Grammar
Finite Automata

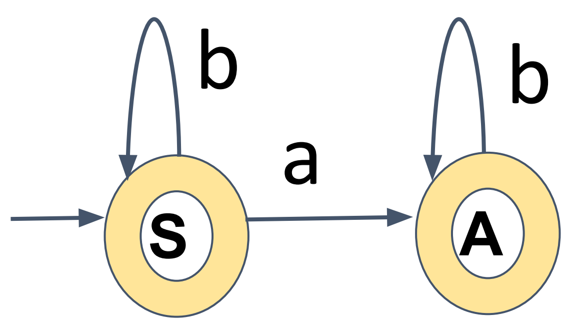
Let’s look at examples to convert :

1. Finite automata to regular grammar
2. Regular Grammar to Finite Automata

1.Convert finite automata to regular grammar for the language to accept at most 3 ‘a’s over the alphabet Σ={a,b}.



Let's look at each transition and perform the conversion

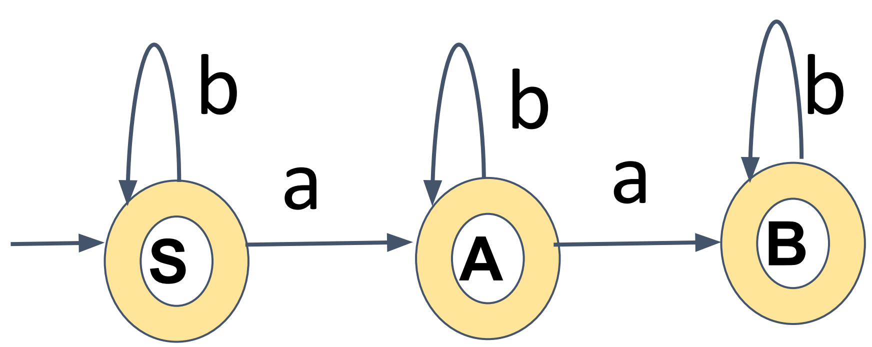


$S \rightarrow bS \mid aA \mid \lambda$

$A \rightarrow bA \mid \lambda$

λ on RHS indicates S and A are final states

Let's look at each transition and perform the conversion



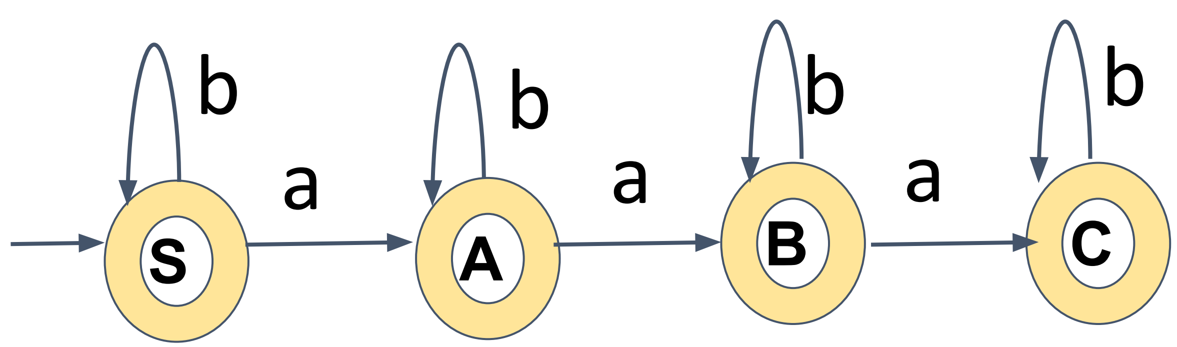
$S \rightarrow bS \mid aA \mid \lambda$

$A \rightarrow bA \mid aB \mid \lambda$

$B \rightarrow bB \mid aC \mid \lambda$

λ on RHS indicates S,A and B are final states

Let's look at each transition and perform the conversion



$S \rightarrow bS \mid aA \mid \lambda$

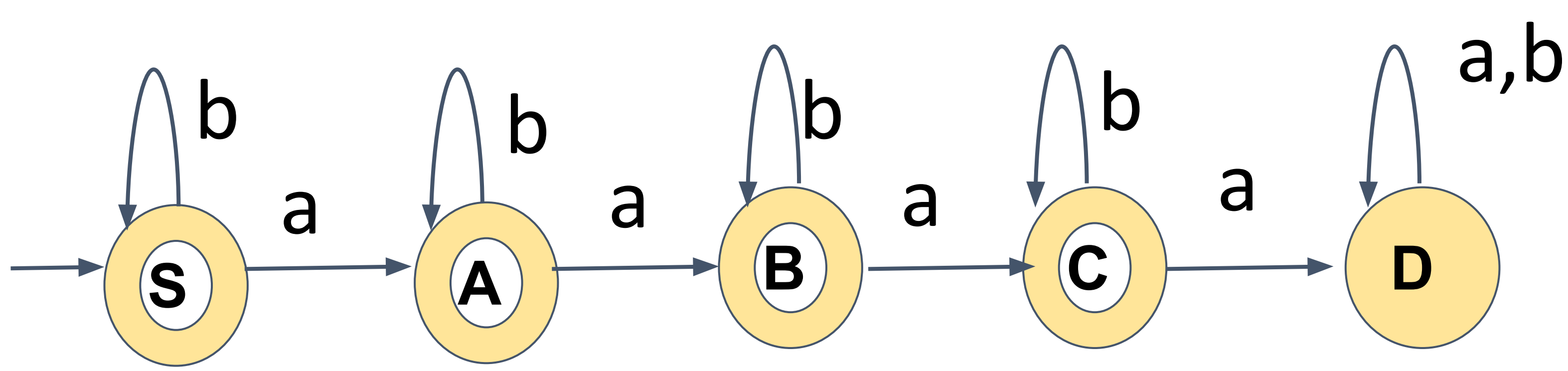
$A \rightarrow bA \mid aB \mid \lambda$

$B \rightarrow bB \mid aC \mid \lambda$

$C \rightarrow bC \mid aD \mid \lambda$

Let's look at each transition and perform the conversion

Since D is a dead state, we will not encode C-> aD and D -> aD | bD as D is a useless Non-terminal(that means it never terminates)



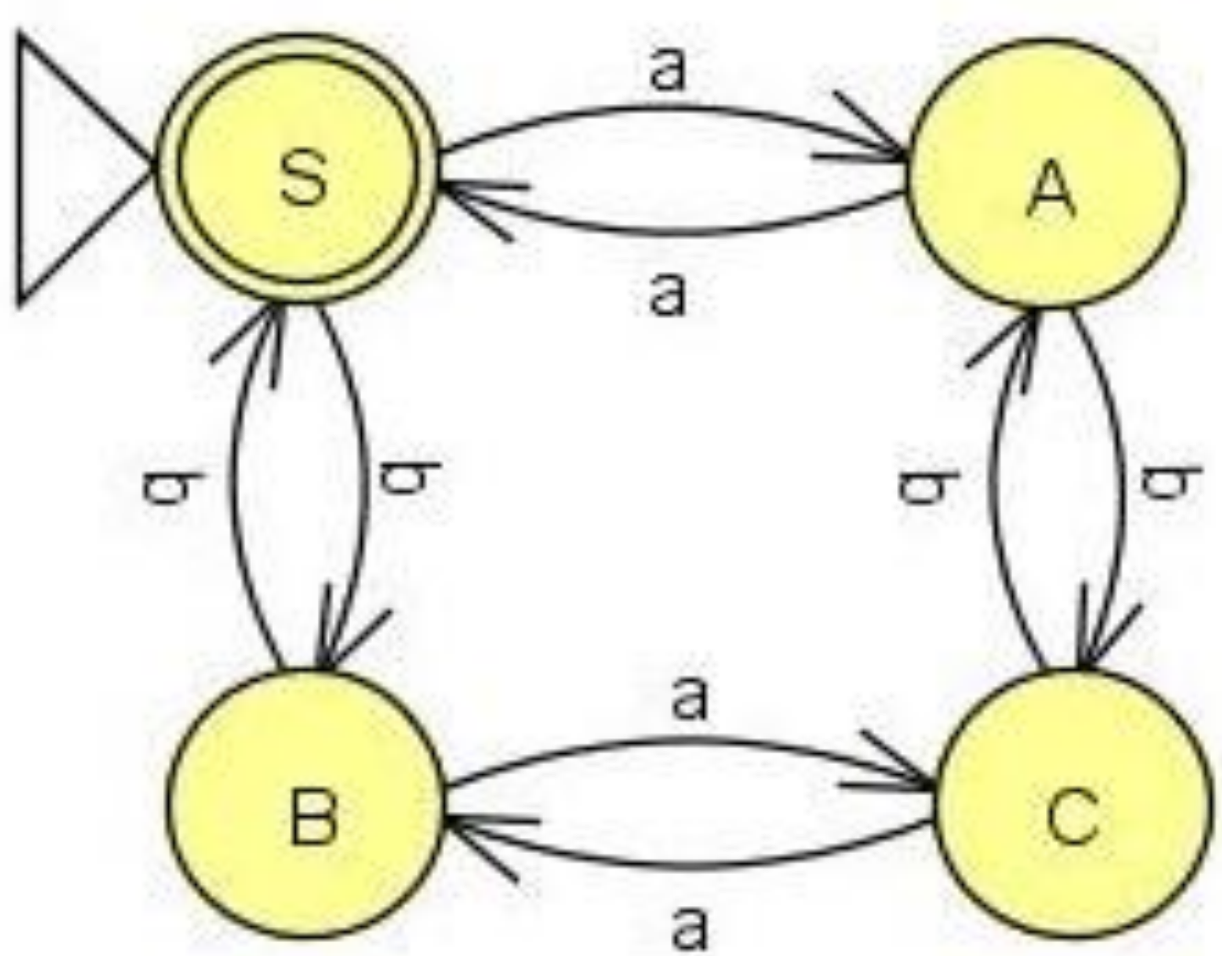
$S \rightarrow bS \mid aA \mid \lambda$

$A \rightarrow bA \mid aB \mid \lambda$

$B \rightarrow bB \mid aC \mid \lambda$

$C \rightarrow bC \mid \lambda$

2.Convert a given finite automata accepting $L=\{n_a(w) \bmod 2=0$ and $n_b(w) \bmod 2=0\}$ to regular grammar.



2.Convert a given finite automata accepting L={$n_a(w)$mod 2=0 and $n_b(w)$mod 2=0 } to regular grammar.

Start state of automata will be the start symbol of the grammar.
We start with S,S on seeing terminal a it moves to state A($S \longrightarrow aA$) and on seeing terminal b it moves to state B ($S \longrightarrow bB$).
Since S is also the final state,we introduce the production $S \longrightarrow \lambda$.
Grammar is

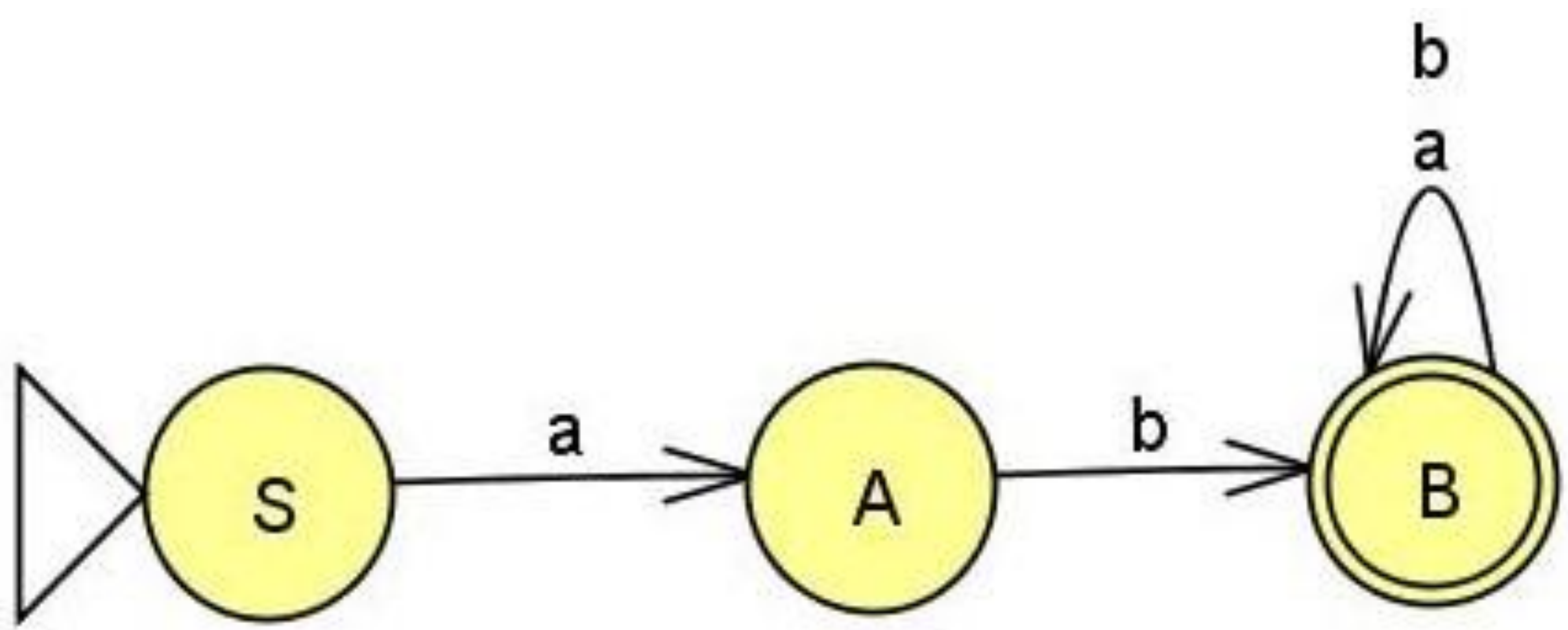
$S \longrightarrow aA|bB|\lambda$

$A \longrightarrow aS|bC$

$B \longrightarrow aC|bS$

$C \longrightarrow aB|bA$

3. Converting a given finite automata accepting L={abw,w∈{a,b}*} to regular grammar.

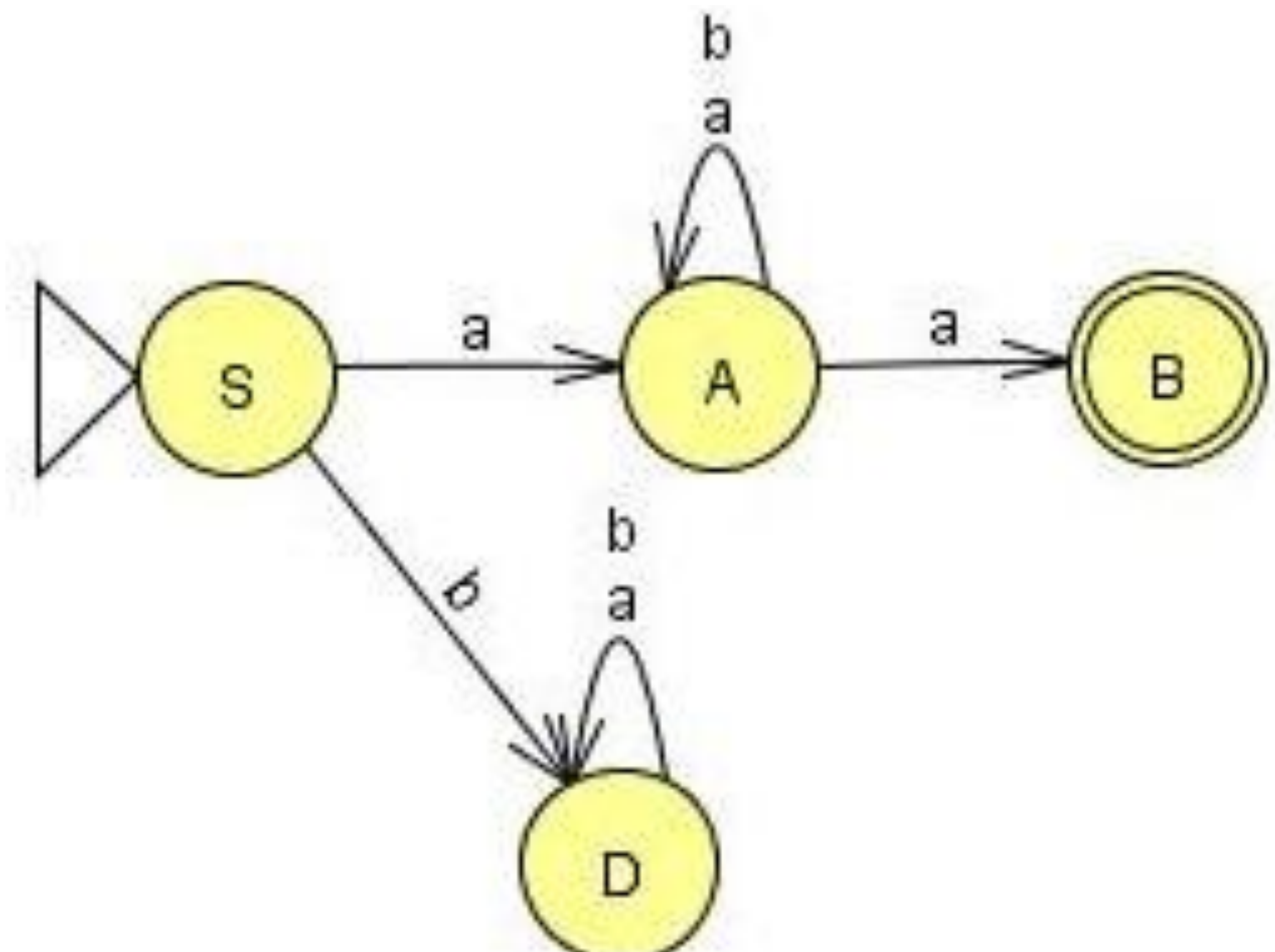


3. Converting a given finite automata accepting $L=\{abw, w \in \{a,b\}^*\}$ to regular grammar.

Grammar is,

$S \longrightarrow aA$
$A \longrightarrow bB$
$B \longrightarrow aB \mid bB \mid \lambda$

4. Converting given finite automata accepting L={awa,w∈{a,b}*} to regular grammar.



PES UNIVERSITY ONLINE

4. Converting given finite automata accepting $L=\{awa, w \in \{a,b\}^*\}$ to regular grammar.

So the grammar is,

$S \longrightarrow aA$
$A \longrightarrow aA|bA|aB$
$B \longrightarrow \lambda$

Converting Regular grammar to finite automata

Example 1 :

$A \longrightarrow aB|bA|b$

$B \longrightarrow aC|bB$

$C \longrightarrow aA|bC|a$

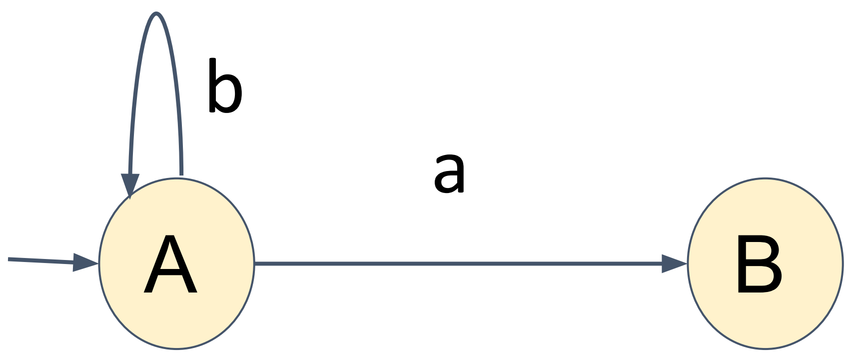
Variables->state

terminal symbols ->symbols of finite automata

Example 1 :

A⟶aB|bA|b
B⟶aC|bB
C⟶aA|bC|a

- Initial start symbol will be the start state



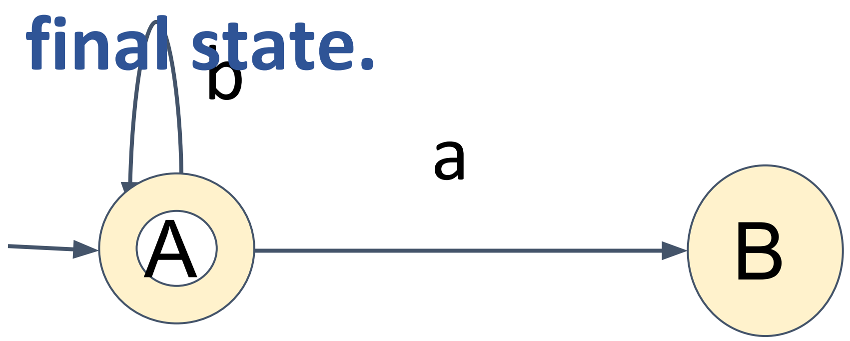
Example 1 :

$A \longrightarrow aB|bA|b$

$B \longrightarrow aC|bB$

$C \longrightarrow aA|bC|a$

- **Transition $A \longrightarrow bA$ transforms to $A \longrightarrow b$,when $A \longrightarrow \lambda$ $A \longrightarrow \lambda$ indicates A is a final state.**

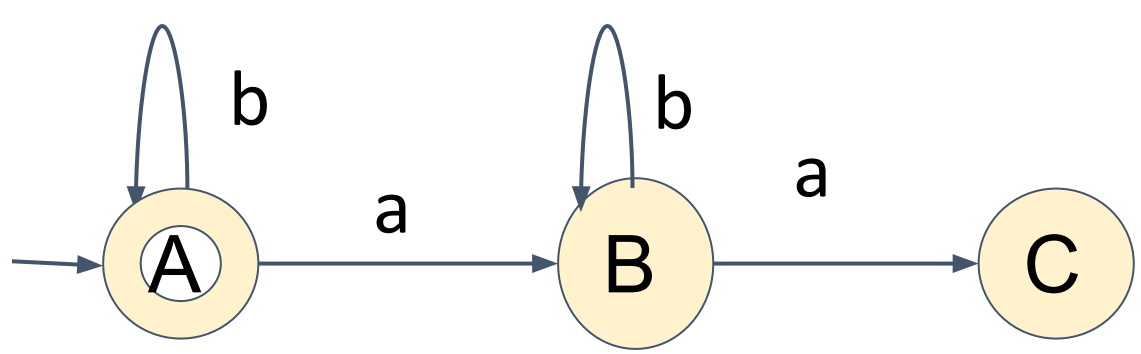


Example 1 :

$A \longrightarrow aB \mid bA \mid b$

$B \longrightarrow aC \mid bB$

$C \longrightarrow aA \mid bC \mid a$



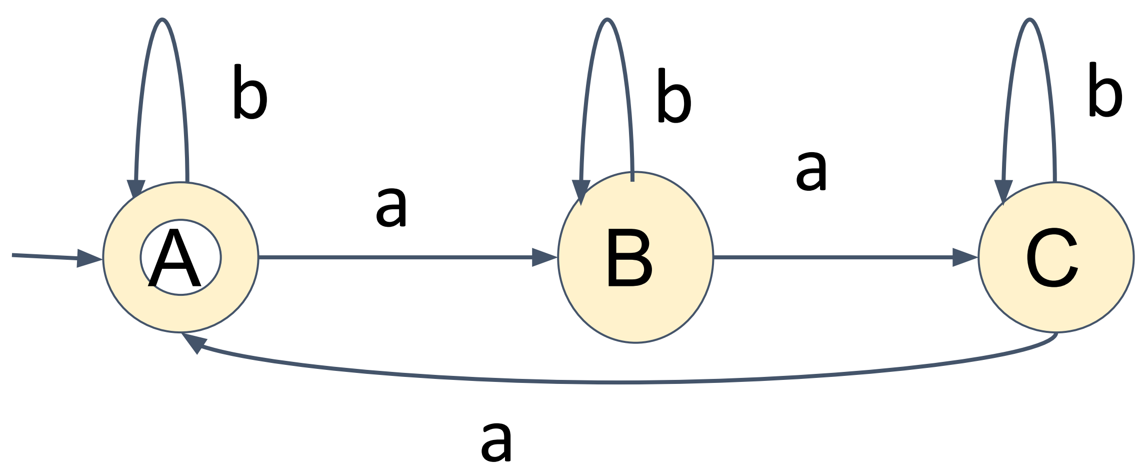
Example 1 :

$A \longrightarrow aB|bA|b$

$B \longrightarrow aC|bB$

$C \longrightarrow aA|bC|a$

- Transition $C \longrightarrow aA$ to $C \longrightarrow a$ when when $A \longrightarrow \lambda$ indicates A is a final state.



Example 2 :

$S \longrightarrow 01A$

$A \longrightarrow 10B$

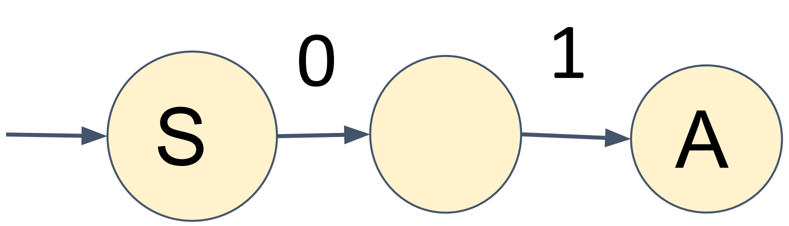
$B \longrightarrow 0A \mid 11$

Example 2 :

S⟶01A

A⟶10B

B⟶0A|11

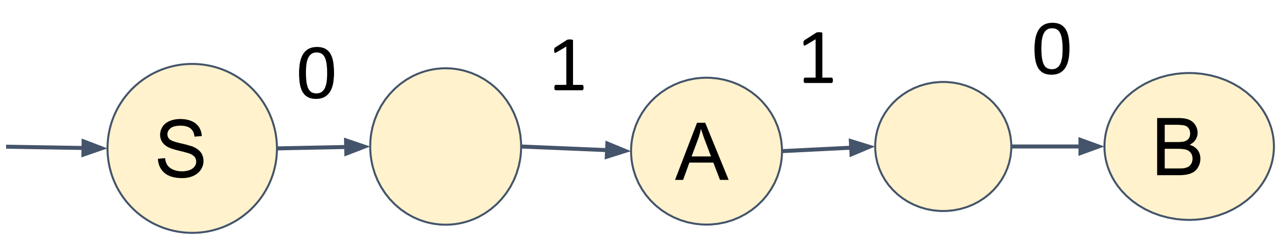


Example 2 :

$S \longrightarrow 01A$

$A \longrightarrow 10B$

$B \longrightarrow 0A \mid 11$



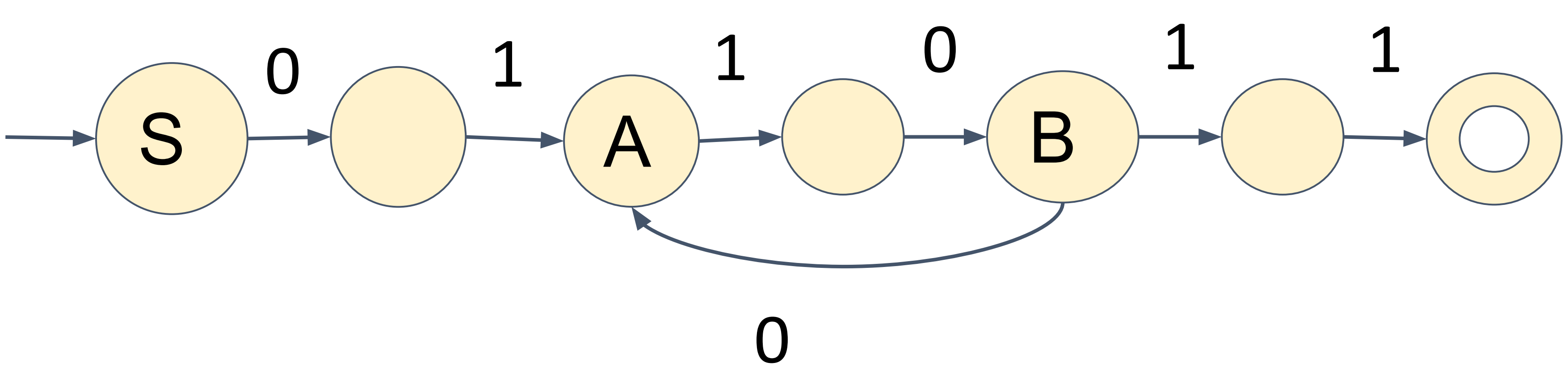
Example 2 :

$S \longrightarrow 01A$

$A \longrightarrow 10B$

$B \longrightarrow 0A|11$

$B \longrightarrow 11$ indicates on state B on consuming the input 11 we reach final state



Example 3 :

$S \longrightarrow bS \mid aA$

$A \longrightarrow aA \mid bA$

$A \longrightarrow aB$

$B \longrightarrow bbB$

$B \longrightarrow \lambda$

Example 3 :

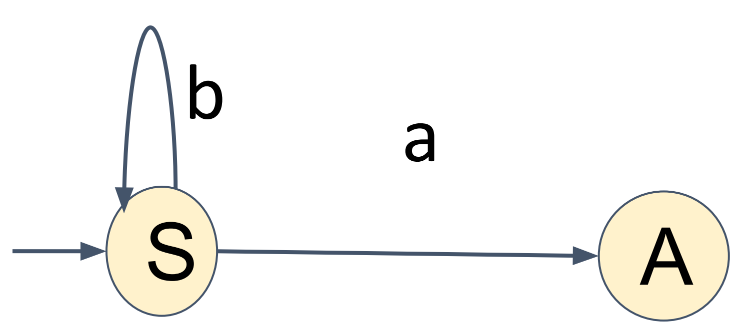
$S \longrightarrow bS \mid aA$

$A \longrightarrow aA \mid bA$

$A \longrightarrow aB$

$B \longrightarrow bbB$

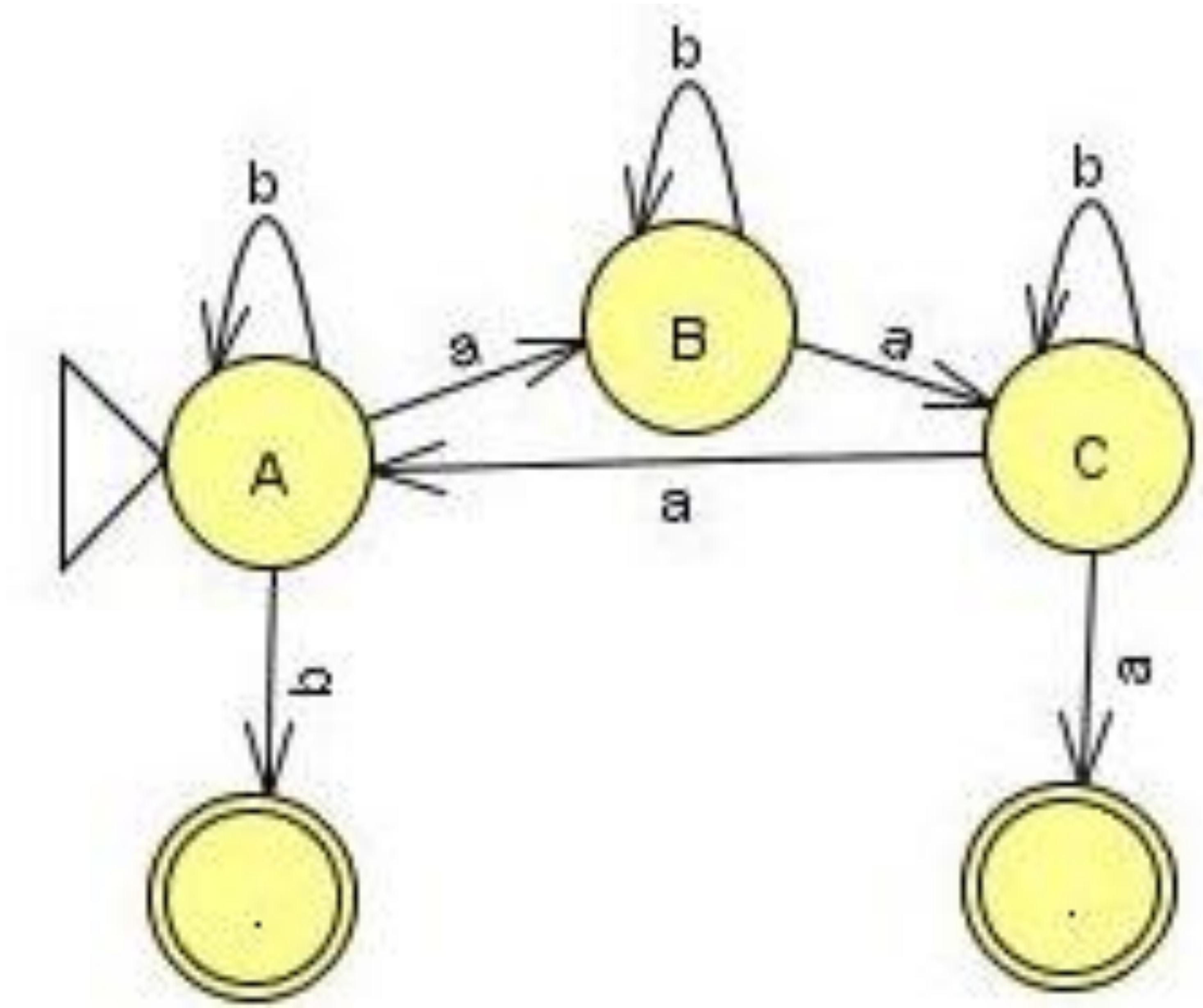
$B \longrightarrow \lambda$



Grammar

$S \rightarrow bS \mid aA$
$A \rightarrow aA \mid bA \mid aB$
$B \rightarrow bbB \mid \lambda$

Automata



Aspects of grammar

- Generative (Derivative)
- Analytical Aspect

Generate All Strings w ϵ L(G)

Check whether w ϵ L(G) ??

Parse Tree

What is a Tree?

Leaves

Branches

Root

Technically we represent a Tree upside down

Root-

Branches

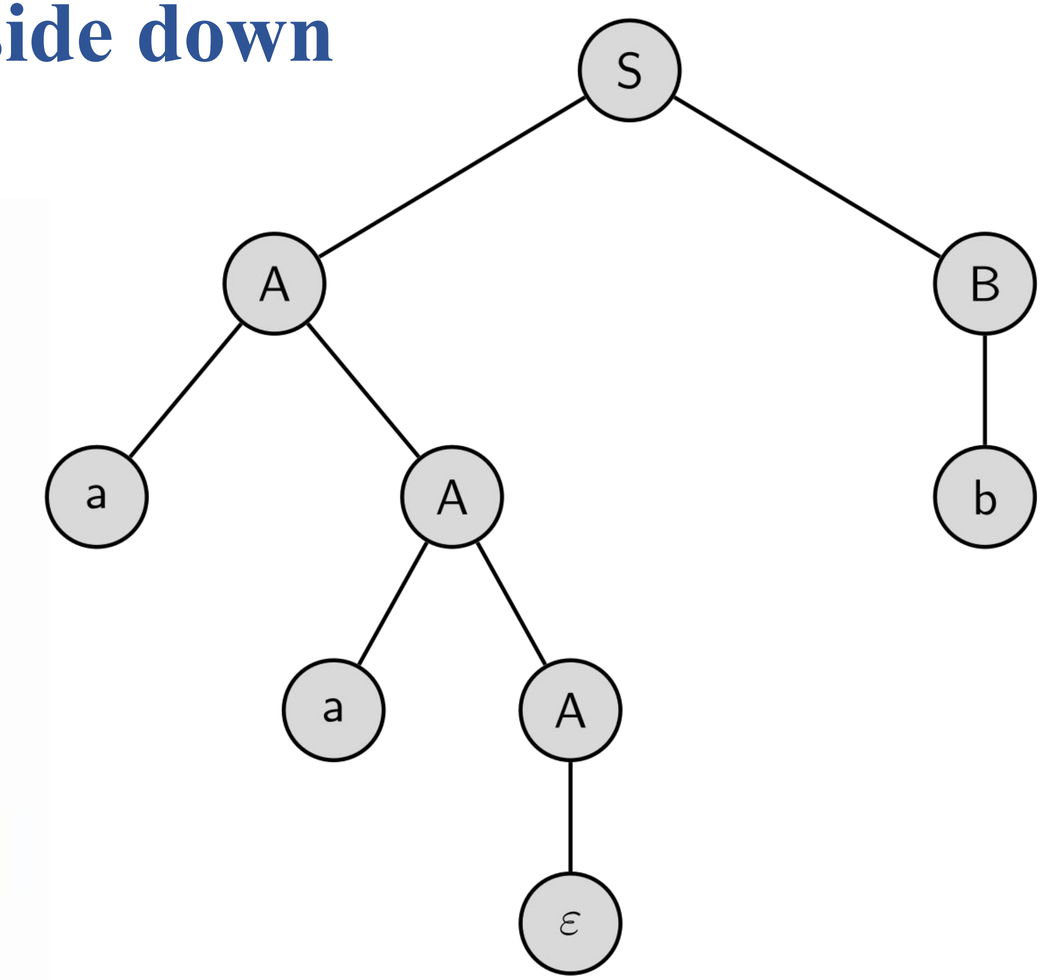
Leaves

Technically we represent a Tree upside down

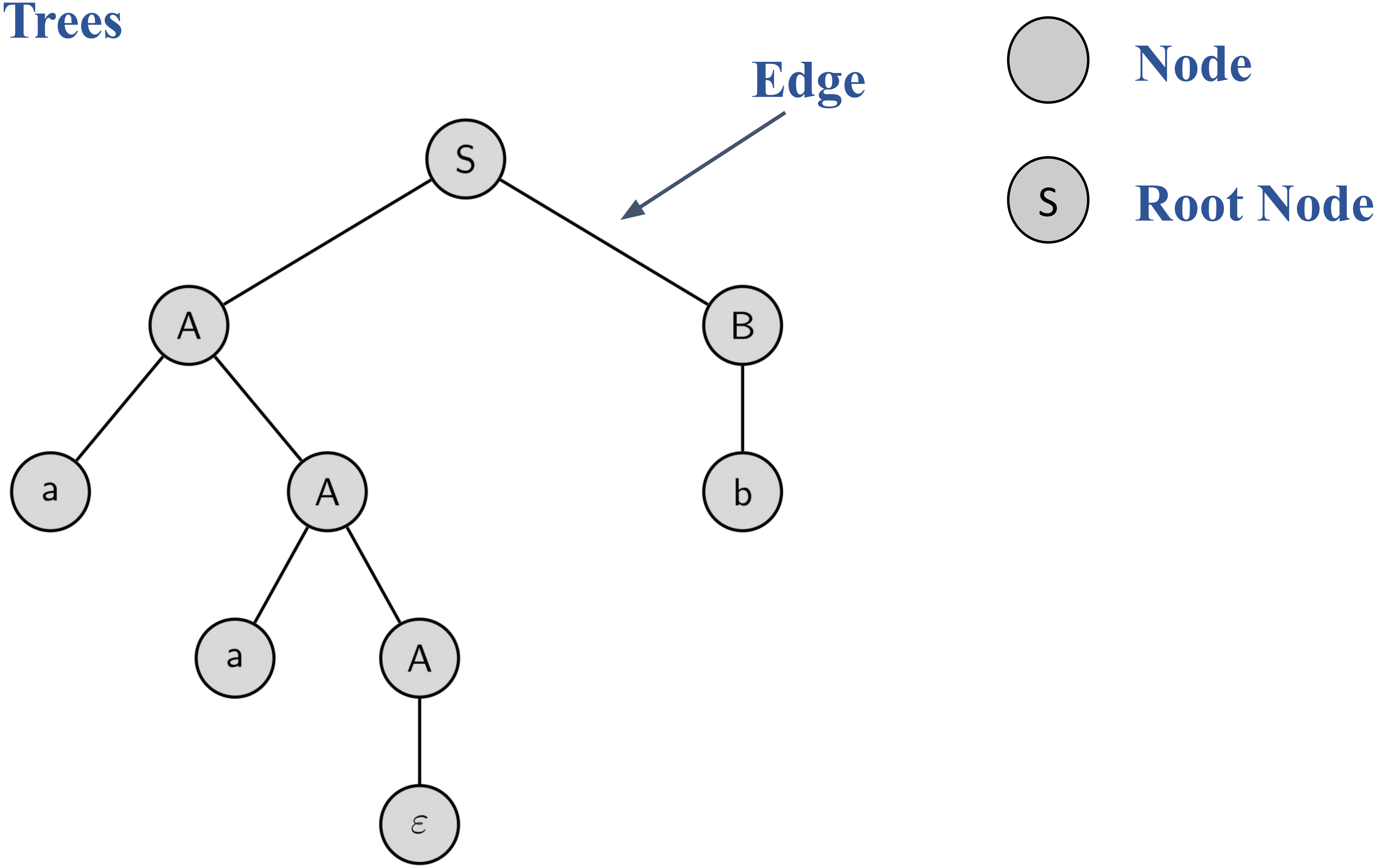
Root

Branches

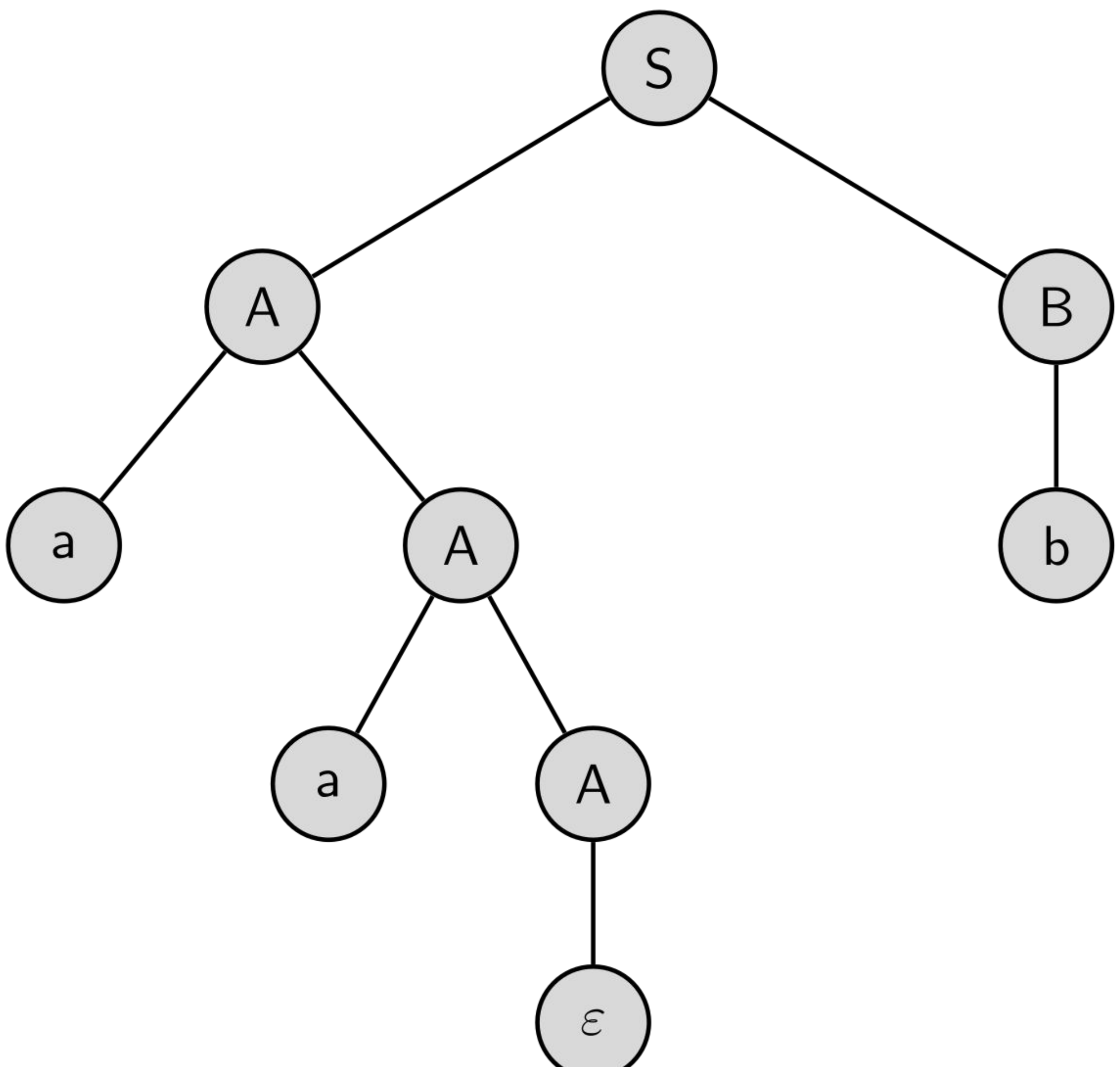
Leaves



Trees



Parse Tree / Derivation Tree



S Root Node (Start Symbol)

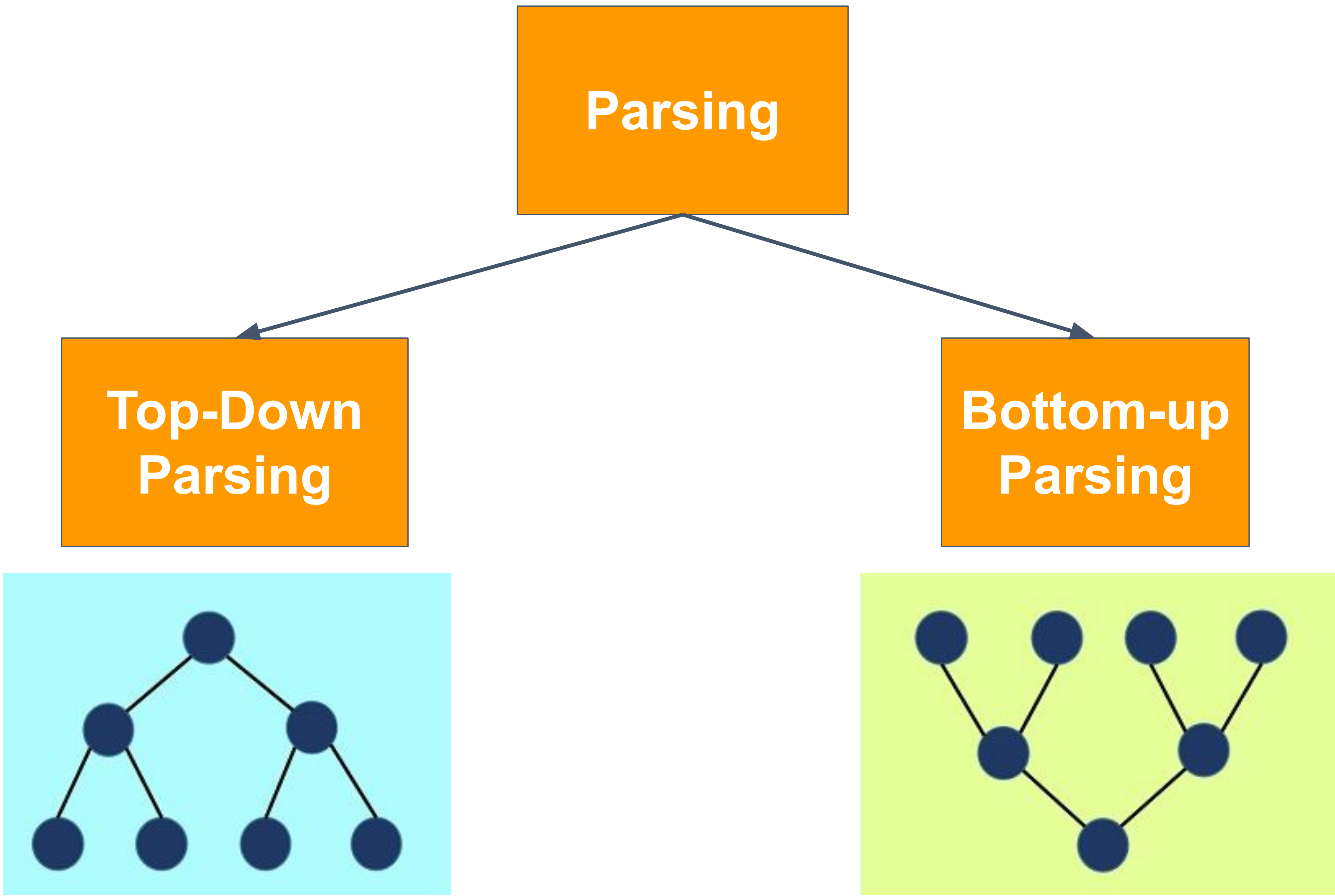
V Non-Terminals - Interior Nodes

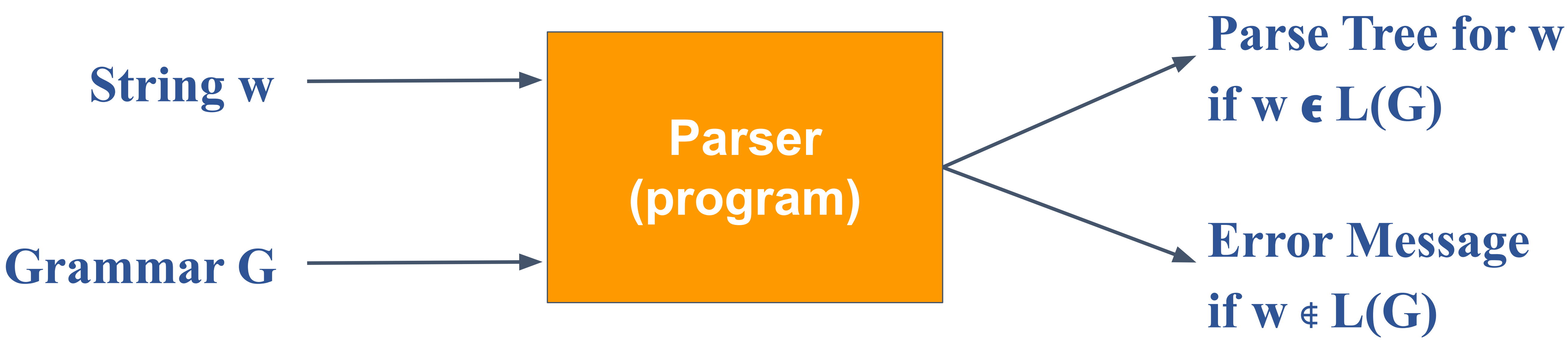
T Terminal Symbols - Leaf nodes

Yield of Parse Tree - aab

Parsing is the process of determining whether a String $w \in L(G)$??

Unit 2 - Types of Parsing



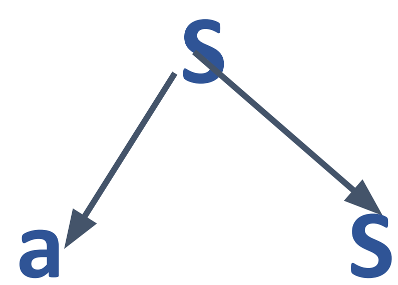

String w
Grammar G
Parser
(program)
Parse Tree for w
if w ϵ L(G)
Error Message
if w ∉ L(G)

Let us look at few examples and generate parse tree for a given String w and Grammar G.

Parse Tree

S Root node

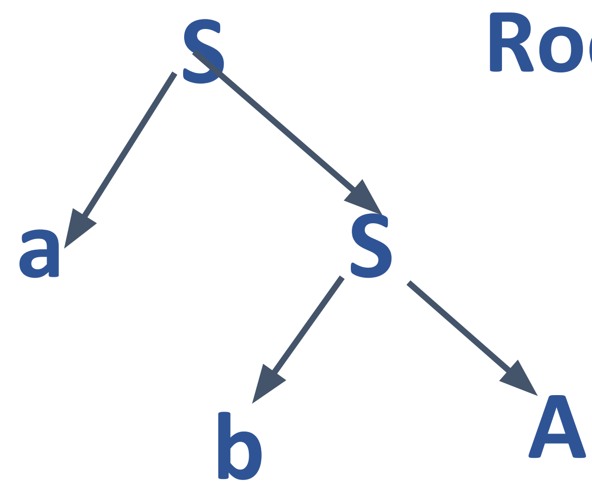
Parse Tree



Root node

S⟶aS

Parse Tree

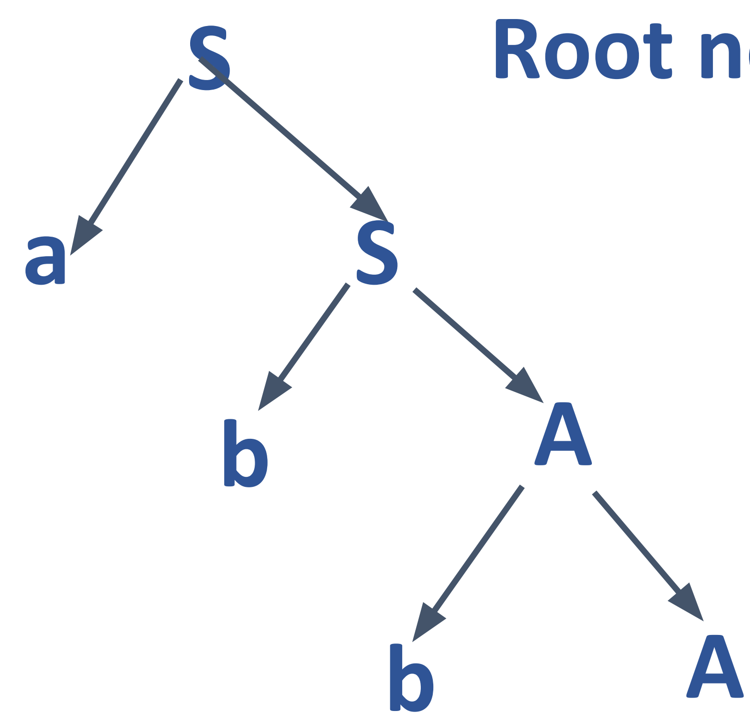


Root node

$S \rightarrow aS$

$S \rightarrow bA$

Parse Tree



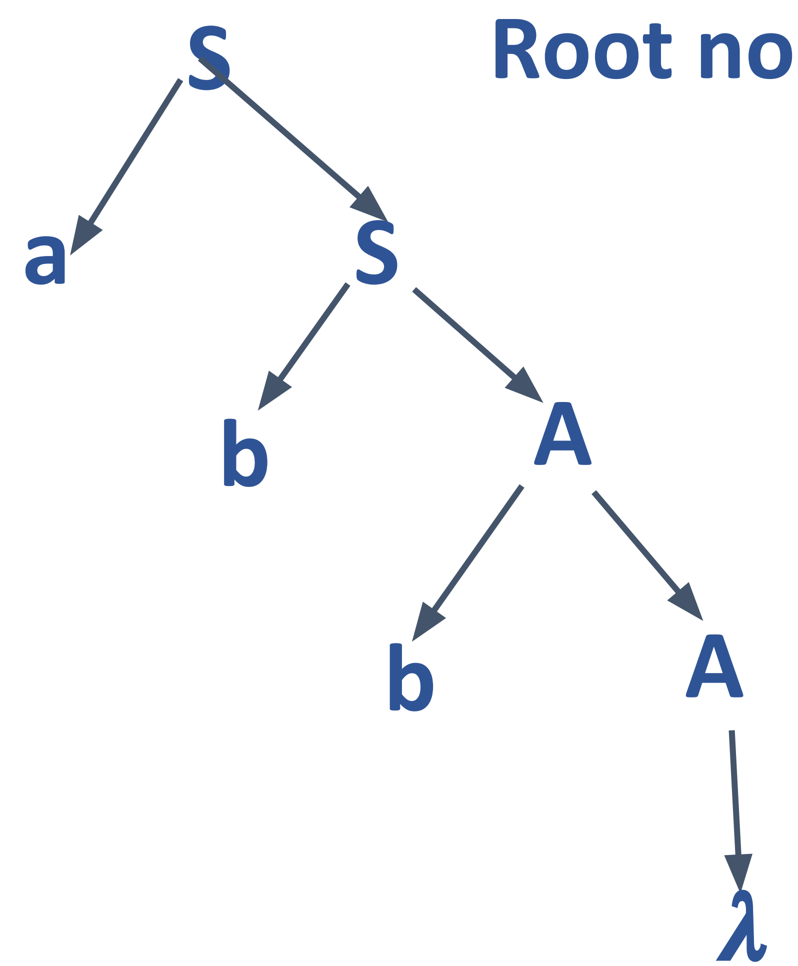
Root node

S⟶aS

S⟶bA

A⟶bA

Parse Tree



Root node

$S \longrightarrow aS$

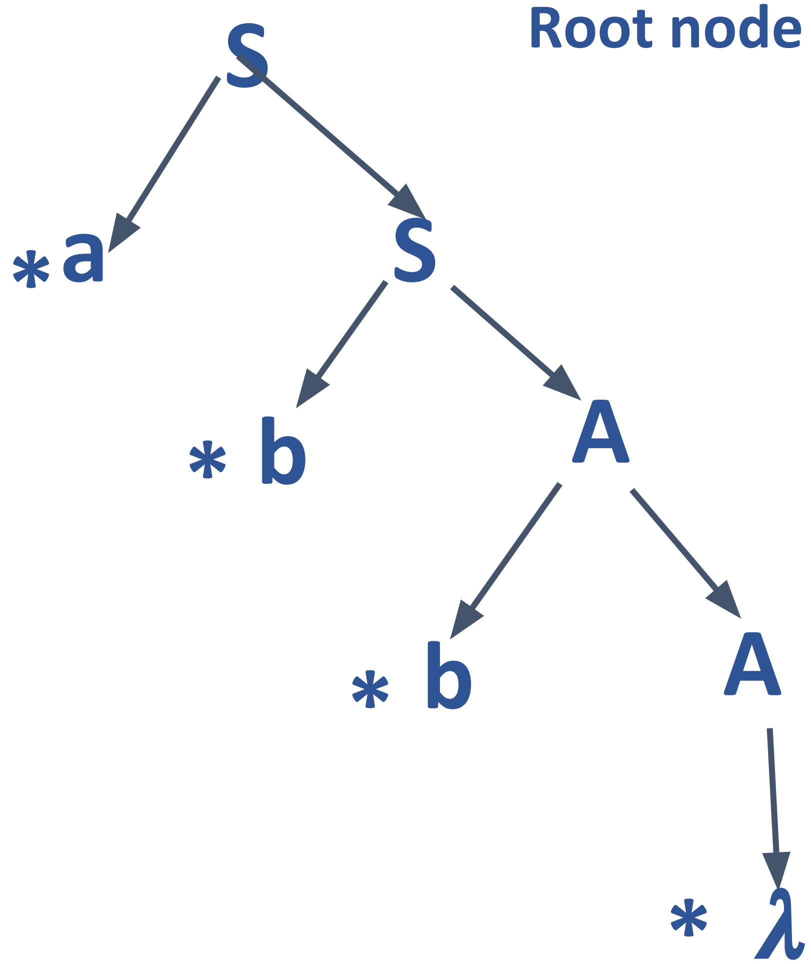
$S \longrightarrow bA$

$A \longrightarrow bA$

$A \longrightarrow \lambda$

Parse Tree

Root node



$S \longrightarrow aS$

$S \longrightarrow bA$

$A \longrightarrow bA$

$A \longrightarrow \lambda$

Leaf Node:*

Yield of the tree:abb

2. Analytical aspect

Parsing

Top down

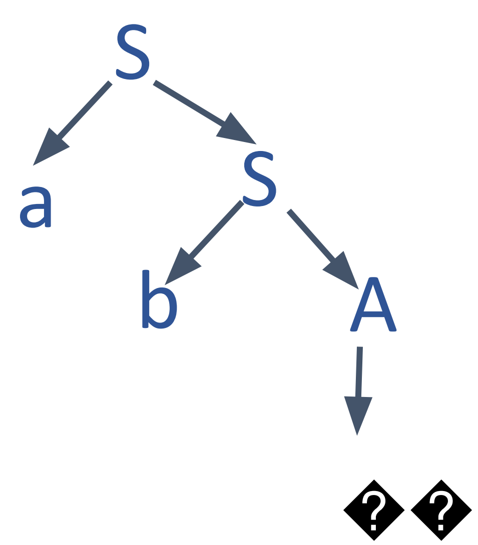
Bottom up

2. Analytical aspect

$S \longrightarrow aS$
$S \longrightarrow bA$
$A \longrightarrow bA$
$A \longrightarrow \lambda$

Parsing

Top down | **Bottom up**

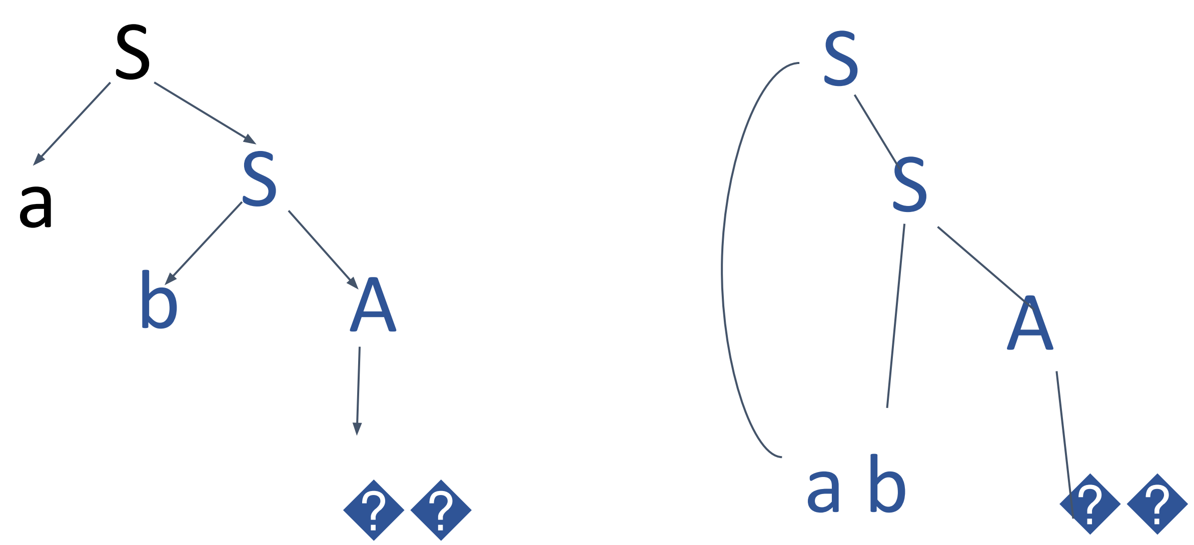


2. Analytical aspect

$S \longrightarrow aS$
$S \longrightarrow bA$
$A \longrightarrow bA$
$A \longrightarrow \lambda$

Parsing

Top down | **Bottom up**



All about Left Linear Grammar

Unit 2 - Constructing a Left Linear Grammar

Right Linear

$A \longrightarrow aB$

$B \longrightarrow aB \mid bB \mid \lambda$

L={starting with 'a'}

→

Left Linear

??

https://www.youtube.com/watch?v=8a2jBm5Syjs

Unit 2 - Constructing a Left Linear Grammar

Right Linear

$A \longrightarrow aB$

$B \longrightarrow aB \mid bB \mid \lambda$

L={starting with ‘a’}

Reverse the Symbols on RHS

Left Linear

$A \longrightarrow Ba$

$B \longrightarrow Ba \mid Bb \mid \lambda$

L=?

https://www.youtube.com/watch?v=8a2jBm5Syjs

Unit 2 - Constructing a Left Linear Grammar

Right Linear	Reverse the Symbols on RHS	Left Linear
$A \longrightarrow aB$	→	$A \longrightarrow Ba$
$B \longrightarrow aB \mid bB \mid \lambda$		$B \longrightarrow Ba \mid Bb \mid \lambda$
L={starting with 'a'}		L^R={ending with 'a'}

When we reverse the RHS in every production in the right linear grammar for " L" we get a left linear grammar which will represent L^R.

https://www.youtube.com/watch?v=8a2jBm5Syjs

Converting Finite Automata to Left linear grammar

Unit 2 - Constructing a Left Linear Grammar

Finite Automata for L

↓ Change initial to final ,final to initial
Reverse the directions of the transition

Finite Automata for L^R

↓

Right Linear grammar for L^R

↓ Reverse all symbols on RHS of production

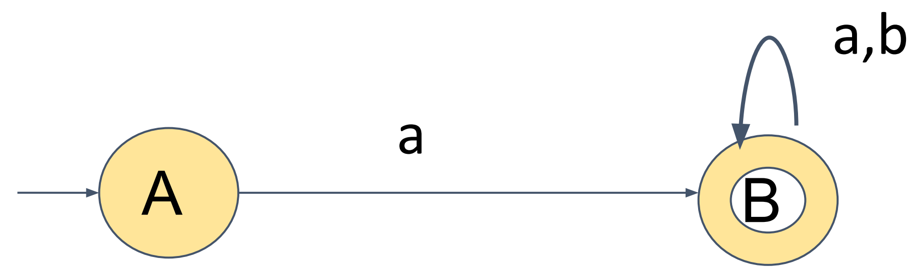
Left Linear grammar $(L^R)^R = L$

https://www.youtube.com/watch?v=8a2jBm5Syjs

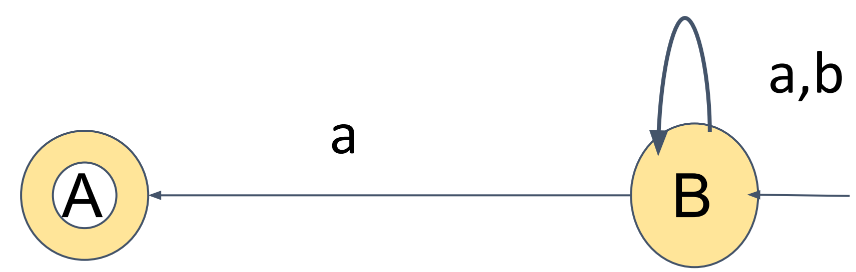
Automata Formal Languages and Logic

Unit 2 - Constructing a Left Linear Grammar

FA for L = {aw, w ϵ {a,b}*}



FA for L^R = {wa, w ϵ {a,b}*}



RLG for L^R

Right Linear Grammar

B⟶aB|bB|aA

A⟶λ

LLG for $(L^R)^R$ =L

Left Linear Grammar

B⟶Ba|Bb|Aa

A⟶λ

Converting Left linear grammar to Finite automata

Unit 2 - Converting left Linear Grammar to finite automata

Left Linear grammar for "L"

Reverse the productions in the grammar for "L" we get L^R

Right linear grammar L^R

Construct Finite Automata L^R

Reverse the finite automata L^R

Finite Automata for$(L^R)^R = L$

Unit 2 - Constructing a Left Linear Grammar

LLG for L

Left Linear Grammar

$B \longrightarrow Ba \mid Bb \mid Aa$

$A \longrightarrow \lambda$

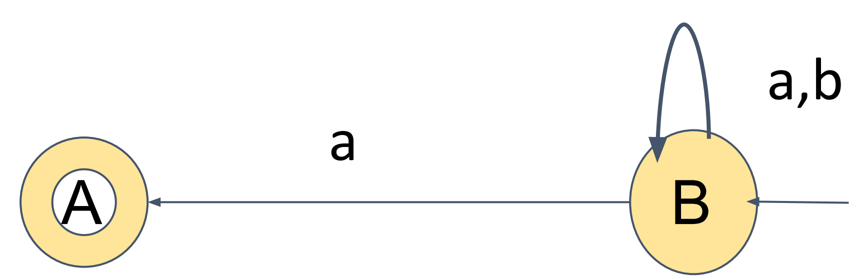
RLG for L^R

Right Linear Grammar

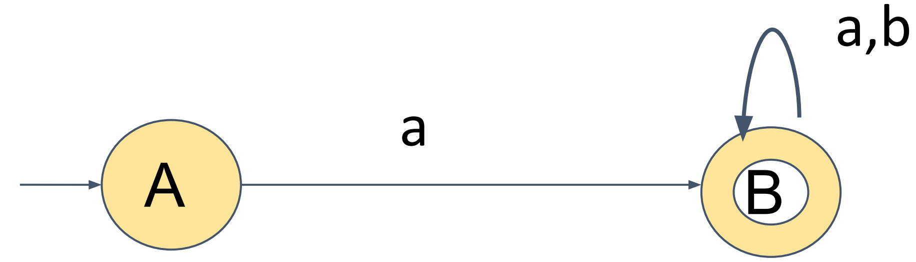
$B \longrightarrow aB \mid bB \mid aA$

$A \longrightarrow \lambda$

FA for L^R = {wa, w ϵ {a,b}*}



FA for $(L^R)^R$ =L = {aw, w ϵ {a,b}*}



Which one is easier LLG or RLG??

Unit 2 - Which one is easier ,Left or Right Linear Grammar?

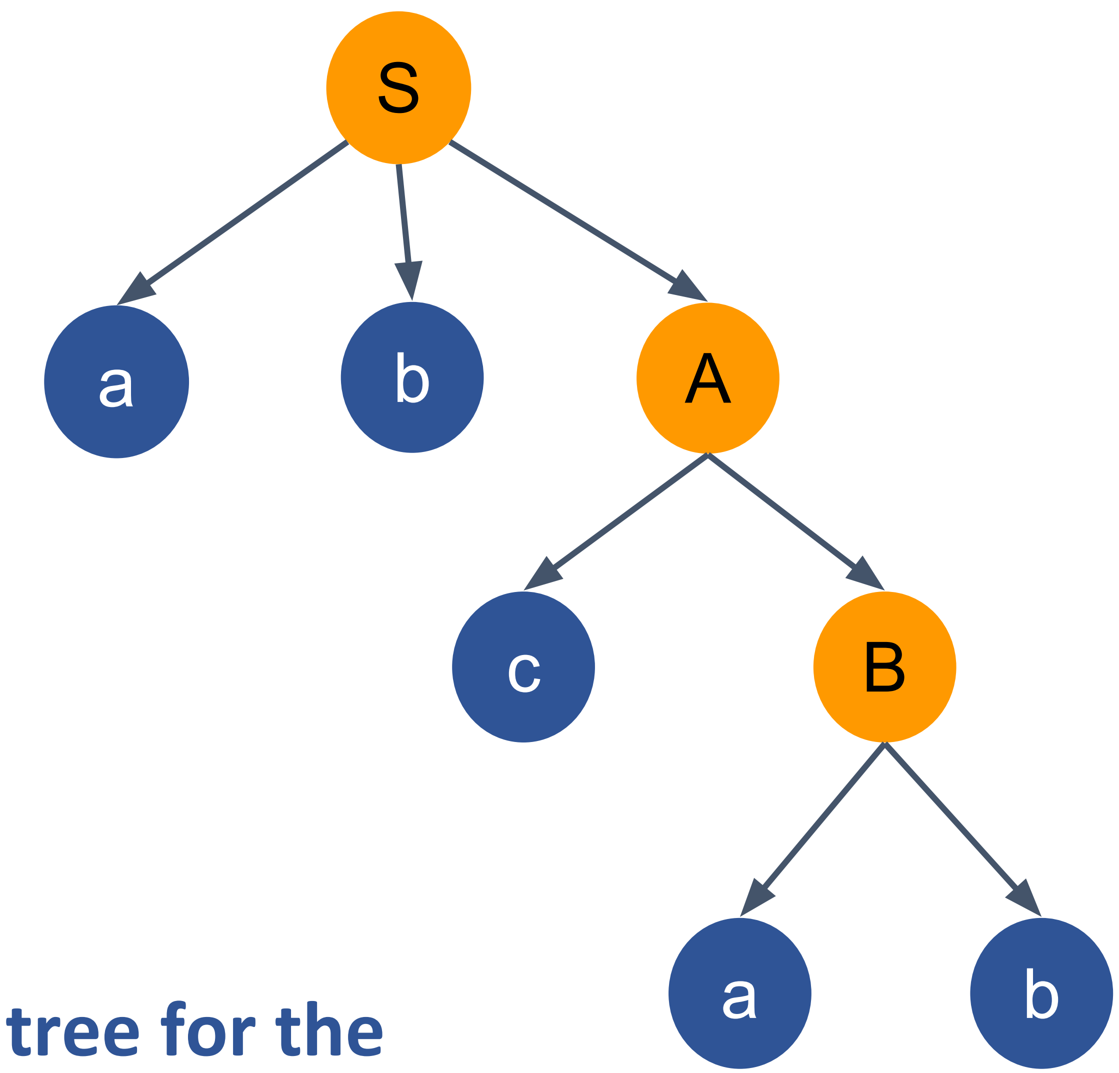
Right Linear

S→abA

A→cB|aC

B→ab

C→b



Construct Parse tree for the String "abcab"

https://www.slideserve.com/kaloni/how-to-convert-a-left-linear-grammar-to-a-right-linear-grammar

Unit 2 - Which one is easier,Left or Right Linear Grammar?

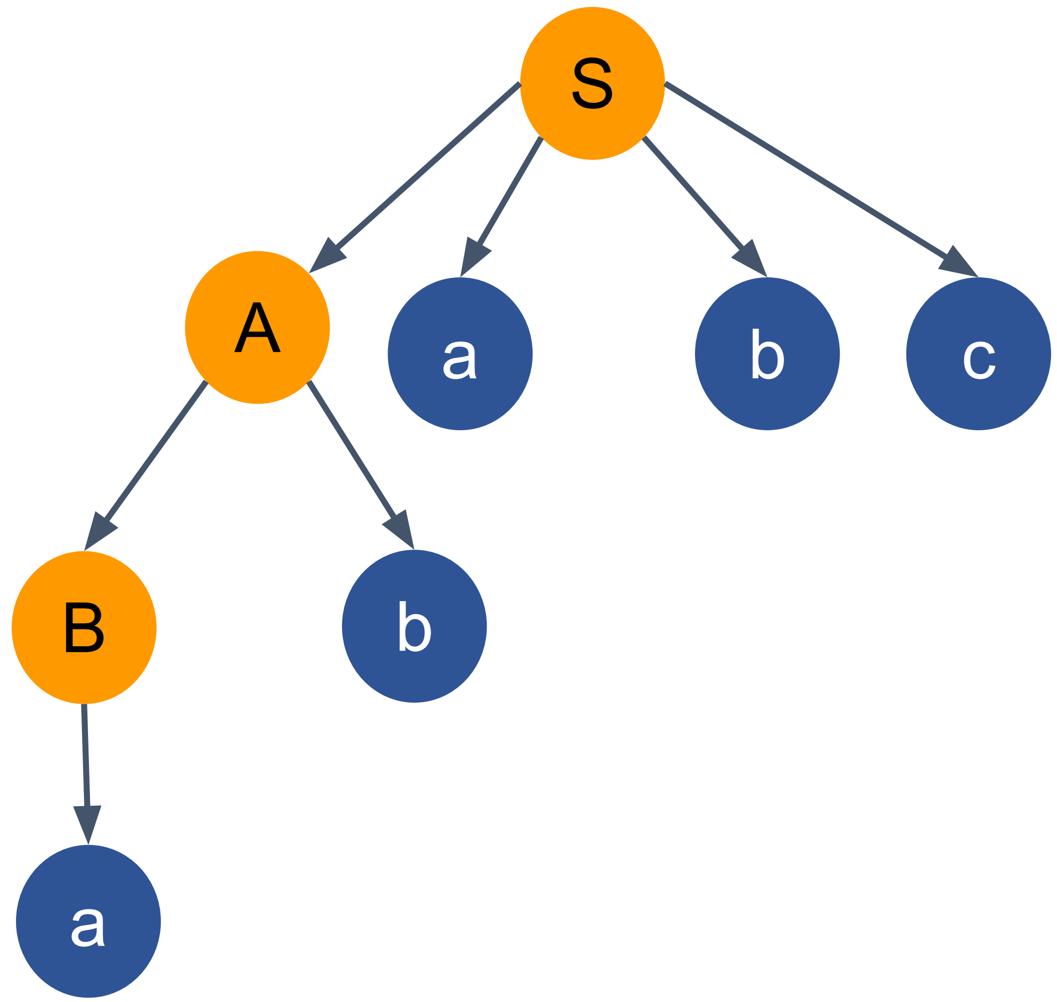
Left Linear

S→Aabc

A→Bb|C

B→a

C→b



Construct Parse tree for the String "ababc"

https://www.slideserve.com/kaloni/how-to-convert-a-left-linear-grammar-to-a-right-linear-grammar

THANK YOU

Preet Kanwal
Department of Computer Science & Engineering

preetkanwal@pes.edu

+91 80 6666 3333 Extn 724