DAY:_____ DATE:_____

4th Operation on R.L

intersection.

L1: aa substring

$\{aa, aab, baa, \ldots\}$

R.E = $(a+b)^* aa (a+b)^*$

L2: Even no of a's.

$\{\lambda, b, bb, bbb, \ldots, aa, baa, \ldots\}$

(not sure) R.E = $b^* + (aa + b + a(b)^* a)^*$

intersection ⇒ satisfies both lang.

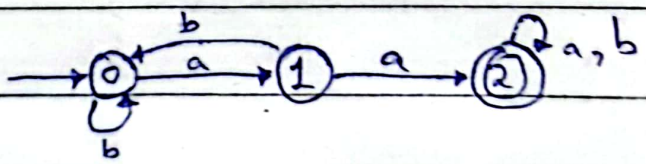
$L1 \cap L2$,

De Morgan's Law:

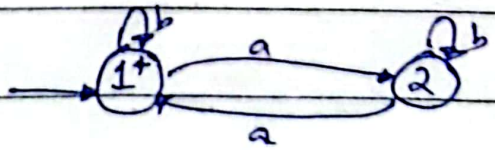
$L1 \cap L2 = (L1' \cup L2')'$

$\{\} =$

L1:



L2:

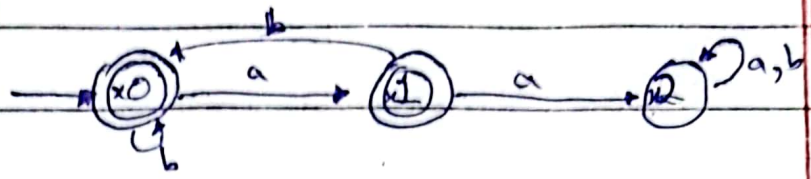


⇒ **Compliment of Language**

⇒ negate ⇒ swap final & non-final

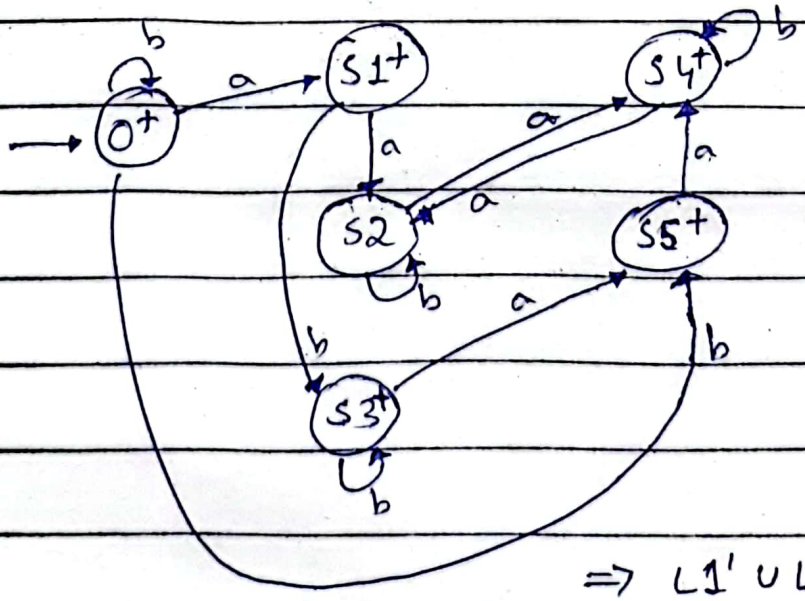
DAY:_____ DATE:_____

L_1':



L_2':

		a	b
f	$S_0(x0,y1)$	$S1(x1,y2)$	$S0(x0,y1)$
f	$S1(x1,y2)$	$S2(x2,y1)$	$S3(x0,y2)$
	$S2(x2,y1)$	$S4(x2,y2)$	$S2(x2,y1)$
f	$S3(x0,y2)$	$S5(x1,y1)$	$S3(x0,y2)$
f	$S4(x2,y2)$	$S2(x2,y1)$	$S4(x2,y2)$
f	$S5(x1,y1)$	$S4(x2,y2)$	$S0(x0,y1)$



$\Rightarrow L_1' \cup L_2'$

(now again negate to get $(L_1' \cup L_2')'$

$\Rightarrow$ Reg is closed over intersection.

DAY:_____ DATE:_____

TOA-16 (19/03/24)

Non Reg Language

R.E ✗ DFA/NFA ✗

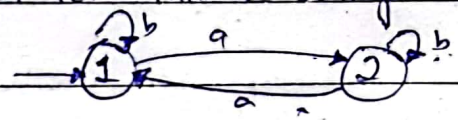
$\Sigma = \{a, b\}$.

L = Language of Palindrome.

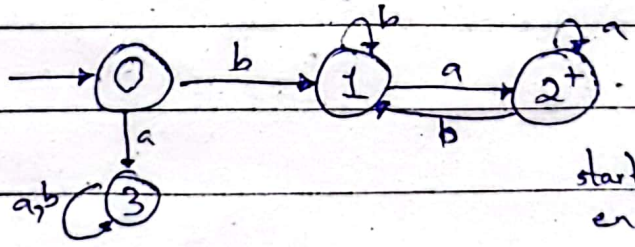
$\{a, b, aa, bb, \ldots\}$.

RL or non R.L ?

DFA for infinite strings ⇒ loop is must.



even a's.



start with b, end on a.

$\{ba, bba, baa, \ldots\}$.

⇒ If states are finite, and infinite strings are produced, then must be a loop (self or multiple states).

⇒ R.L have property, repeating part is also part of R.L.

Pumping Lemma:

↳ to test reg or non reg language.

1) Suppose L is R.L

DAY:_____ DATE:_____

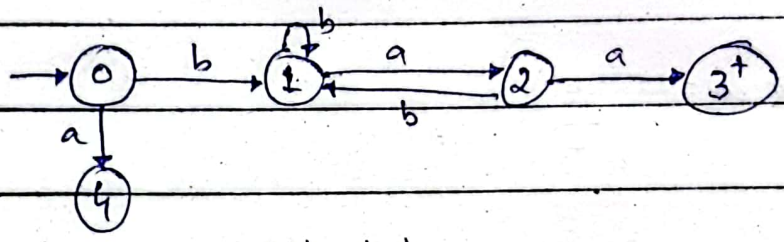
2) The finite automata of language consists of N no of finite states.

3) Suppose string $w \in L$ where $|w| > N$.

4) Divide string w into substring $\Rightarrow w = xyz$.

i) $|y| \neq 0$

ii) $|x| + |y| \leq N$.



$w \rightarrow \underbrace{bbb}_{x}\ \underbrace{ab}_{y}\ \underbrace{ab\ a}_{z}$

iii) $w = xy^i z$. $i > 1$.

if $w \in L \Rightarrow L$ is R.L else not a R.L

xy^1y^2z ↳ will fail on 2nd or 3rd iteration or more.

$\Rightarrow$ Suppose Palindrome language is R.L.

let $N = 5$.

$\{a, b, bb, aa, aba, \ldots\}$.

1) $w = \underbrace{ba}_{x}\ \underbrace{ab}_{y}\ \underbrace{aab}_{z}$

$xy^1z \Rightarrow baababaab \Rightarrow$ Palindrome (reg)

DAY: _____ DATE: _____

2 ⇒ $w = baabaab$ (one more try)

$x \; y \; z$

$xy^2z \Rightarrow baaabaab \Rightarrow$ non-palindrome (not reg).

$L = a^n b^n \quad n \geq 1$ (Reg or non-reg?)

$\{ab, aabb, aaabbb, \ldots\}$

$n = 5$

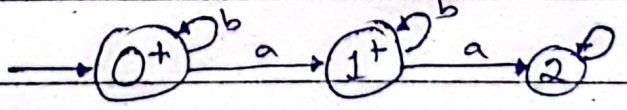
1 ⇒ $w = aabb$

$x \; y \; z$

$xyyb \Rightarrow aaabb \Rightarrow \notin L$ so non-reg.

Q L = only one a

R.E = $b^* a b^*$



$w = bab$ $\quad n = 3$

$x \; y \; z$

$xyyz \Rightarrow \notin L$ so non reg (wrong)

Myhill Nerode Theorem: (2nd Theorem)

Suppose language L is regular over Σ

$L1$ = even no of a's & b's.

1) Identify distinct classes belonging to your language.

Lang $\subseteq$ { $c1 \Rightarrow$ even a even b

non-Lang $\subseteq$ { $c2 \Rightarrow$ even a odd b

DAY:_____ DATE:_____

non lang = { $c_3 \Rightarrow$ odd a even b
$c_4 \Rightarrow$ odd a odd b

2) Pick 2 string x & y from same class.

3) Pick another string z from d/f class.

4) If xz and yz both strings belongs to the language class or non-lang classes then it is regular else non-reg.

x : bab (c_3) y : abb (c_3)

z = ab (c_4)

1) xz = babab yz = abbab

c_2 (non-lang) c_2 (non lang)

Regular lang (wrong)

2) xz = aabb (c_1) y = bbaa (c_1)

z = ab (c_4)

xz = aabbab (c_4) y = bbaaaab (c_4)

(non lang) (non lang)

regular (wrong)

L2 ⇒ end with a

c_1 : end with a.

c_2 : not end with a.

x : ab y : b. z = a.

DAY:_____ DATE:_____

xz : aba yz : aba

↓

lang class ⇒ Reg

L3 ⇒ only 1 a

c_1 : only 1 a.

c_2 : multiple a.

x : a y : ba.

z : aab.

xz : aaab yz : baaab

non-lang ⇒ Reg.

L4 ⇒ Palindrome

c_1 : palindrome

c_2 : non-palindrome

2st try: x : aba y : bab.

z : ab

xz : abaab babab

↓ c_1 (lang) ↓ c_2 (non-lang)

⇒ non reg lang.

2nd try:

x = bab y = baaab

z = bba

DAY:_____ DATE:_____

xz : babbba yz = baabbba

L_2 L_2

Regular lang (wrong).

TOA-17 (25/03/24)

Decidability.

→ Effective Solvable. ⇒ Algo exists to solve problem in finite no of steps

→ Decision Procedure. ⇒ yes/no to declare using algo

⇒ one language can have multiple RE/DFA.

Q: How to prove both languages are same?

⇒ Produce all strings to prove (impossible)

Ex: L1: {1, 2, 3, ---- 10}

L2: {1, 2, 3, ---- 10}

⇒ $(L1 \cap \overline{L2}) \cup (\overline{L1} \cap L2)$

↳ If both lang are same ⇒ empty set will be resulted.

⇒ If 2 DFA's are given for both languages, create DFA using the formula, if no string is accepted by resultant DFA ⇒ same lang.

DAY:_____ DATE:_____

⇒ $[(L1 \cap L2') \cup (L1' \cap L2)]$.

↳ R.E or DFA could be given for this formula. How to prove using that?

R.E:

let $(a+b)^*(abb+\lambda)(a^++aa)^*$

1) Remove all Kleen Star operators:

$(a+b)^*(abb+\lambda)(a^++aa)$

2) Remove all Kleen Plus operators:

$(a+b)(abb+\lambda)(a+aa)$.

3) Removing R.H.S. part of union op

$(a)(abb\overline{\lambda b})(a)$

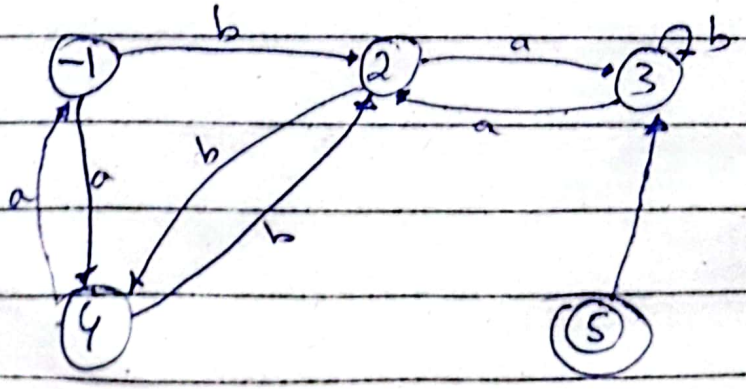
4) Concatenate the word

aabba

↳ If word is produced ⇒ not same lang.

else if empty string ⇒ same lang.

DFA:



DAY:_____ DATE:_____

⇒ No path till final state ⇒ same lang
else: diff lang.

⇒ Multiple final states,

1) Start tracking from initial st.

2) Move to next states, mark them & remove edges (used).

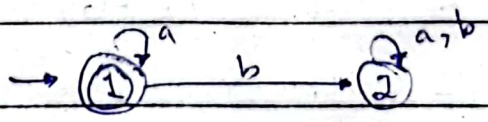
3) If final state highlighted ⇒ diff languages.

Example:

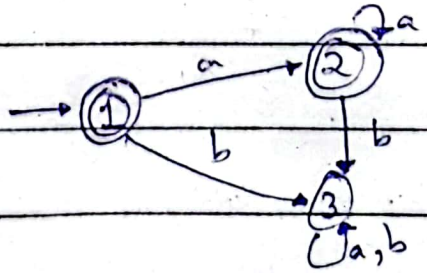
L1: a^*

L2: $\lambda + aa^*$.

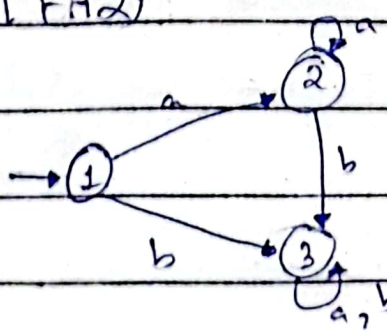
DFA1:



DFA2:



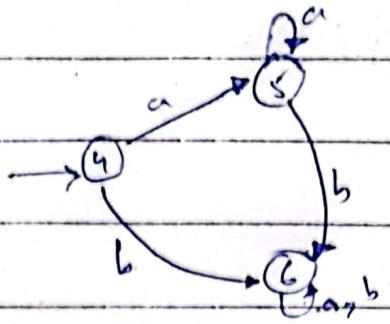
(FA1' ∩ FA2)'



VU handout

DAY:_____ DATE:_____

$(FA1 \cap FA2')'$ → If this compliment removed all are final.



⇒ No final ∪ No final ⇒ No final state.

↳ nothing accept ⇒ same lang.

⇒ Is no final state possible?

TOA-18 (26/03/24)

Non Reg Language

Context Free Grammer (CFG)

CFG = { Terminal , non-Terminal , Production }

- Terminal → Σ
- non-Terminal → variables (can be replaced by val)
- Production → rule against each variable to replace with value

capital letter ⇒ non-Terminal

(already defined) a, b s Terminal ⇐

S, A s non Terminal ⇐

Grammar:

$S \rightarrow a$

$S \rightarrow b$

$S \rightarrow A$

$A \rightarrow aA$

$A \rightarrow \lambda$

⇒ Production for each non-terminal is must.

DAY:_____ DATE:_____

non-terminal ⟶ terminal/non-terminal/mixture.

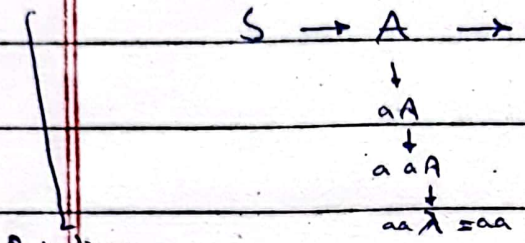
⇒ 1st non-terminal (S) is starting non-terminal by convention.

S or start ⇒ non-terminal

⇒ S can be replaced by a, b, A

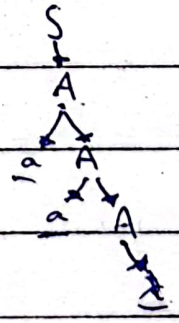
Ex: to produce aa

$S \rightarrow A \rightarrow aA \rightarrow aaA \Rightarrow aa\lambda \Rightarrow aa.$



Derivation: producing string from production.

1) Derivation Tree



leaf ⇒ string.

2) Conventional steps:

$S \Rightarrow A$

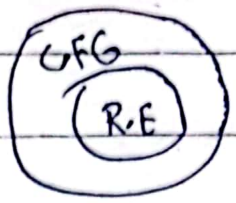
$\Rightarrow aA$

$\Rightarrow aaA$

$\Rightarrow aa\lambda$

$\Rightarrow aa$

DAY:_____ DATE:_____



⇒ R.E generated can be represented by CFG.

Representing language in CFG

1) $\{\lambda, a, aa, aaa, \ldots\}$

R.E = a^*

CFG:

$S \to aS \mid \lambda$

2) R.E = a^+

$S \to aS$ ⇒ infinite.

$S \to aS \mid a$

3) R.E = $(a+b)^*$

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

4) R.E = $a(a+b)^*b$

1) $S \to aA$

$A \to aA \mid bA \mid b$

2) $S \to aXb$

$X \to aX \mid bX \mid \lambda$

5) L = aa appear

R.E = $(a+b)^*aa(a+b)^*$.

$S \to XaaX$

$X \to bX \mid aX \mid \lambda$

$a(ab)^*b$
aabab

DAY:_____ DATE:_____

6) Equal no of a's & b's in which a comes before b.

$a^n b^n : n > 0$

{ ab, aabb, aaabbb, ... }

1) $S \rightarrow aSb \mid ab$.

2) $S \rightarrow aXb \mid \lambda \Rightarrow a(ab)^n b$

$X \rightarrow aXb \mid \lambda$

7) Palindrome Language (Even length).

{ λ, aa, bb, abba, baab, ... }

$S \rightarrow aSa \mid bSb \mid \lambda$.

8) Odd length Palindrome + (even also)

{ λ, a, b, aa, bb, aba, aaa, ... }

$S \rightarrow a \mid b \mid aSa \mid bSb \mid \lambda$.

9) Palindrome without λ

$S \rightarrow a \mid b \mid aSa \mid bSb \mid aa \mid bb$

⇒ context free grammar ⇒ to verify coding syntax.

int a, b, c;

D.T id, id;

$D.S \rightarrow D.T \; v;$

$D.T \rightarrow int \mid float \mid char$

$V \rightarrow id, v \mid id$

DAY:_____ DATE:_____

D.S => Declaration statement.

D.T => Data Type.

=> CFG of C++ if/else.

TOA-19 (01/04/24)

L1 → a's & b's even no.

R.E: $(aa+bb+(ab+ba)(aa+bb)^*(ab+ba))^*$

aaS bbS

CFG: $S \to aSa \mid bSb \mid XSX \mid \lambda \mid SS$

$X \to ab \mid ba$.

=> abbabaabaabs.

$S \to XSX$.

ab | ba

aSa

bSb

aa

=> abbaaa

$S \to XSX$

$ab \Rightarrow X$

$S \to aSa$

bSb

X => SS in CFG

DAY:_____ DATE:_____

L2 : $a^* b^*$

$S \to aS \mid B \mid \lambda$

$B \to bB \mid \lambda$

$S \to AB$

$A \to aA \mid \lambda$

$B \to bB \mid \lambda$

Ambigius Grammar

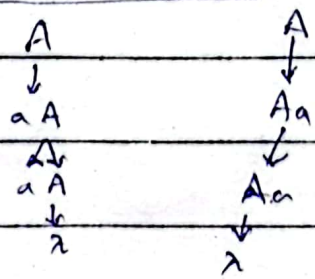
→ one of more left most or right most non-terminal

Derivation
- ↳ left (resolve left most non-terminal first)
- ↳ Right (resolve right most non-terminal first)

⇒ If more than one left most grammar or right most grammar possible ⇒ ambigius grammar

⇒ Compiler parser gets confused.

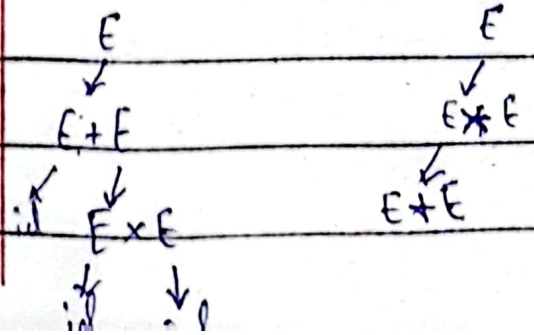
eg $A \to aA \mid Aa \mid \lambda$



⇒ Is grammar ambigius?

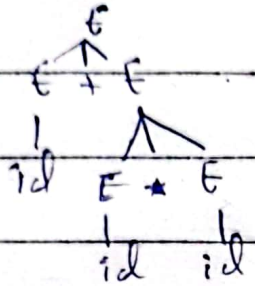
id + id * id;

$E \to E + E \mid E \times E \mid id.$

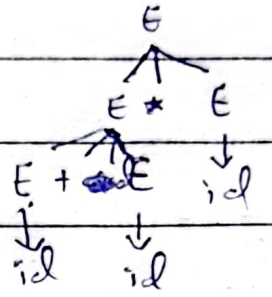


DAY: _____ DATE: _____

$E \to E+E$
$\to id+E$
$\to id+E*E$
$\to id+id*E$
$\to id+id*id$



$E \to E*E$
$\to E+E*E$
$\to id+E*E$
$\to id+id*E$
$\to id+id*id$



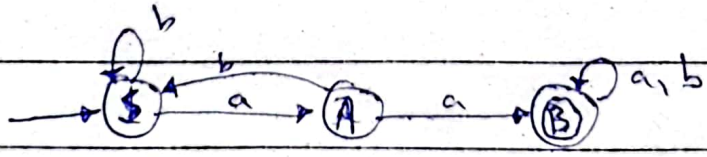
TOA-20 (02/04/24)

Regular Grammar

⇒ Grammar of R.L

⇒ CFG for R.E

L1: $(a+b)^*aa(a+b)^*$



1) States become non-terminal.

2) $S \to aA \mid bS$

$A \to aB \mid bS$

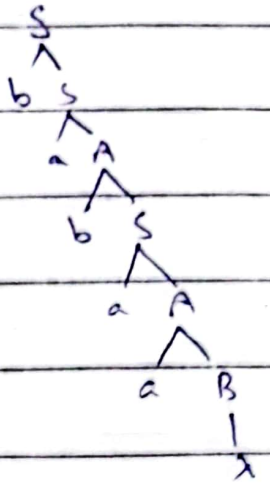
$B \to aB \mid bB \mid \lambda$

2) Define ~~#~~ production = edge + state.

3) λ on final ~~state~~

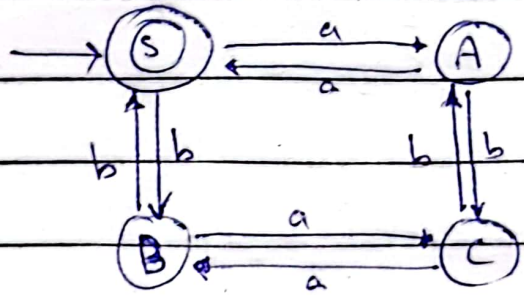
DAY:_____ DATE:_____

verify babaa



→ Easy to generate CFG for R.L by DFA

L2: even a's & even b's.



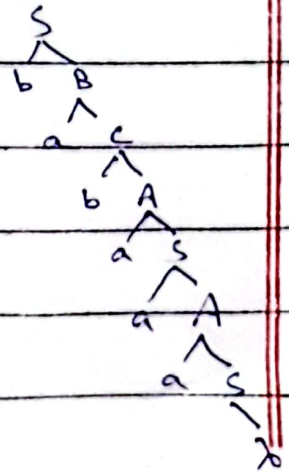
$S \rightarrow \lambda | aA | bB$

$A \rightarrow aS | bC$

$B \rightarrow aC | bS$

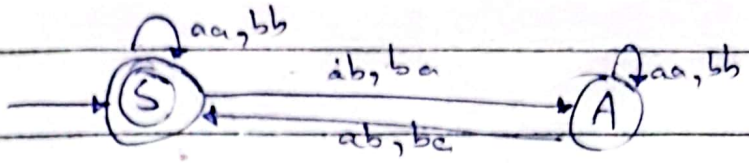
$C \rightarrow bA | aB$

verify: babaaa



DAY:_____ DATE:_____

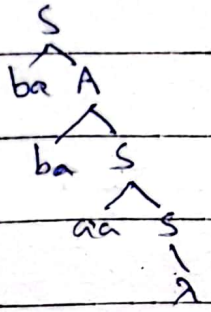
L3: By TG of even a & even b.



$S \rightarrow \lambda \mid aaS \mid bbS \mid abA \mid baA$

$A \rightarrow aaA \mid bbA \mid abS \mid baS.$

verify: babaaa.



⇒ CFG of R.L has certain pattern.

R.G → Semiword | word. ⇒ if this R.L else Rule/non Rule

Semiword ⇒ (terminal)*. non-terminal

word → terminal.

CHOMSKY's Normal form (CNF)

1) Remove Null & Nullable production.
2) Remove Unit production.
3) non-terminal → Two non-terminal.
   // → single terminal

DAY:_____ DATE:_____

Example:

G1: $S \to ab|bA|\lambda$

$A \to aSa|bS|ab.$

1) $S \to ab|bA'$

$A \to aSa|bS|ab|aa|b$

↳ only λ can't be produced by this grammar

G2: $S \to ab|bA|\lambda$

$A \to aSa|bS|ab|S$

↳ indirectly null ⇒ A is nullable

1) $S \to ab|bA|\$$

$A \to aSa|bS|ab|S$

(now A can also produce λ, so put λ against A & S both).

$S \to ab|bA|b$

$A \to aSa|bS|ab|S|aa|b$

↓ if AbS here

$A \to aSa|AbS|ab|S|aa|bS|bAb|b$

DAY:_____ DATE:_____

G3:

$S \to Xa \mid YY \mid aX \mid ZYX$

$X \to Za \mid bZ \mid ZZ \mid Yb$

$Y \to Ya \mid XY \mid \lambda$

$Z \to aX \mid YYY$

Sol:

$S \to Xa \mid YY \mid Y \mid aX \mid ZY \mid X \mid ZX$

$X \to Za \mid bZ \mid ZZ \mid Yb \mid b \mid a \mid Z$

$Y \to Ya \mid XY \mid X \mid a$

$Z \to aX \mid YYY \mid YY \mid Y$

$S \to Xa \mid YY \mid aX \mid ZYX \mid a \mid Y \mid YX \mid ZX \mid ZY \mid X \mid Z$

$X \to Za \mid bZ \mid ZZ \mid Yb \mid a \mid b \mid Z$

$Y \to Ya \mid XY \mid X \mid a \mid \lambda$

$Z \to aX \mid YYY \mid a \mid YY \mid Y$

2) Remove unit Production.

↳ non-terminal → non-terminal.

G1: $S \to ab \mid A \mid bA$

$A \to b \mid aS$

→ $S \to ab \mid b \mid aS \mid bA$

$A \to b \mid aS$.

→ cycles production

G2: $S \to ab \mid aS \mid bA$

DAY:_____ DATE:_____

$A \to b|aS|S$

$\Rightarrow$ $S \to ab|b|aS|\$|bA$

$A \to ab|aS|b|\$|bA$

All values replaced.

G3:

$S \to A|bb \Rightarrow B|b|bb \Rightarrow S|a|b|bb$

$A \to B|b \Rightarrow S|a|b \Rightarrow A|bb|a|b.$

$B \to S|a \Rightarrow A|bb \Rightarrow B|b|bb$

$\Rightarrow a|b|bb.$

$S \to \$|a|b|bb$

$A \to \$|a|b|bb.$

$B \to \$|a|b|bb$

3) Match the pattern

if non-terminal $\to$ 3 non-terminal | X

↳ extra non-terminal.

$X \to YYY$

↳ $Z \to YY$, $X \to YZ$

Convert to CNF

$S \to aA|bB|\lambda$

$A \to aa|SA$

$B \to aS.$

DAY:_____ DATE:_____

$S \rightarrow aA \mid bB$.

$A \rightarrow aa \mid SA \mid$ [illegible] A

$B \rightarrow as \mid a$.

DAY:_____ DATE:_____

TOA-21 (15/04/24)

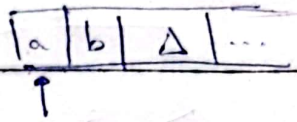
Push Down Automata (PDA)

1- Input Tape (like Array)

2- States ⇒ {Accept, Reject, Start, Read}

3- Stack ⇒ {push, pop}

4. Σ ⇒ Input Alphabet.

Tape: 

(contains input)

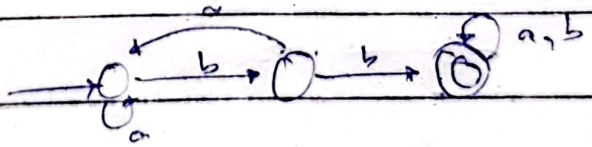
Δ ⇒ empty

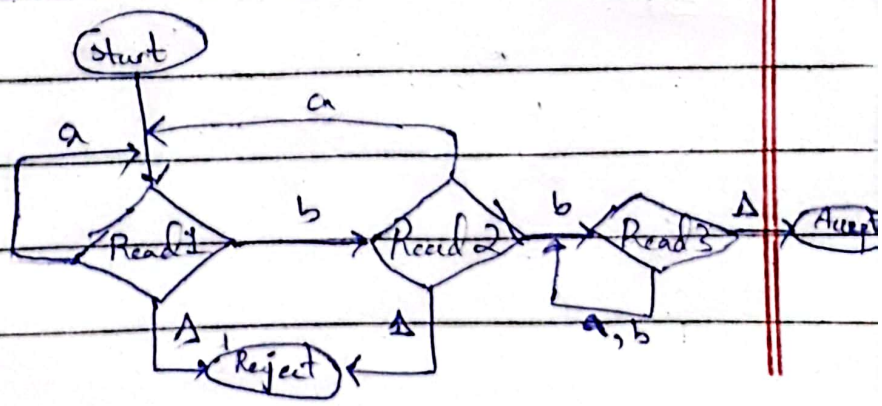
Stack: 

⇒ some PDA can be constructed without stack.

Example:

L1: $(a+b)^* bb (a+b)^*$

DFA: 

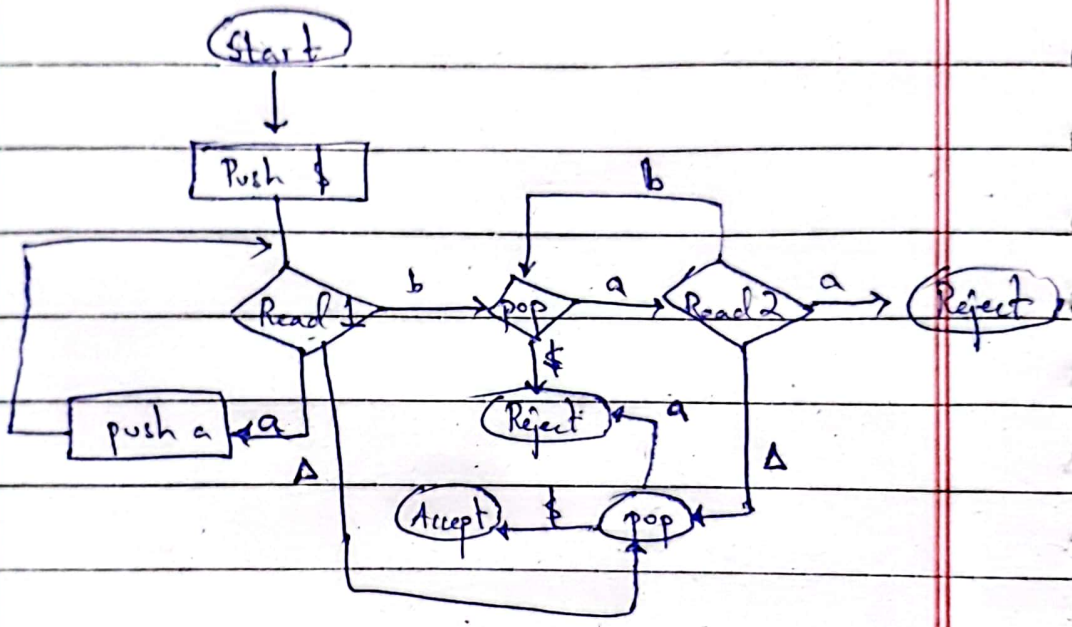
PDA 

DAY:_____ DATE:_____

2) $a^n b^n$

⇒ push any special char other than Δ

PDA:



3) $a^n b^{2n}$ ⇒ pop 1a after 2b.