TOC ASSIGNMENT - 3

Ans 1) (a)

- $\Sigma = \{a, b\}$
- $V = \{S, A\}$
- Start symbol : S
- Productions (P):

$$S \rightarrow aaS \mid aaA$$

$$A \rightarrow bA \mid \epsilon$$

S generate an even number of a's (at least 2), and A generates any number of b's

(b) string : aaaabb

Leftmost Derivation :

$$S \rightarrow aaS$$

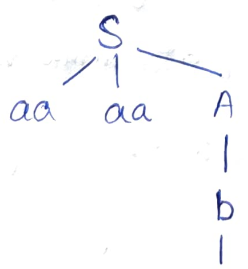
$$\rightarrow aa\ aaA$$

$$\rightarrow aa\ aabA$$

$$\rightarrow aa\ aab\ bA$$

$$\rightarrow aa\ aa\ bb$$

(c) Derivation tree



Leaves : a a a a b b → "aaaabb"

(d) Regular expression for L:

$$(aa)^+ b^*$$

Since this RE exists, L is a regular language

And every regular language is also context-free.

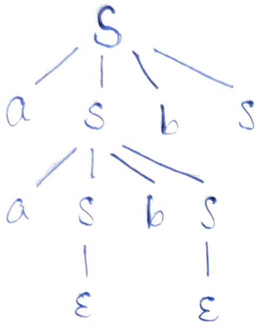
Ans 2) (a) Derive aabb

$S \to aSbS$

$\to a(aSbS)bS$

$\to aa\epsilon b\epsilon b\epsilon = aabb$

$\Rightarrow$ Parse tree



Leaves (left-to-right) : aabb → aabb

b) Prove that the grammar is ambiguous using formal definitions.

A grammar is ambiguous if there exists at least one string in the language that has two distinct parse trees (equivalently two different leftmost or rightmost derivations).

Claim: The grammar $S \to aSbS / \varepsilon$ is unambiguous.

Proof:

- Any nonempty string generated by this grammar must begin with a (because the only production introducing terminals start with a) and therefore must be produced by one application of $S \to aS_1bS_2$.

- In any derivation of a non empty string w, the first a in w must match some b later in w. Let that matching b be at position k. The grammar forces that matching b to be the b produced in the same production as bS_2 that produced the first a. That uniquely splits w into three parts.
- a (the first terminal)
- substring generated by S_1 (between this a and its matching b)

- b (The matching b).
- Substring generated by S_2 (The remainder)
- The Position k (first a's matching b) is uniquely determined by the usual blance argument (count a minus b scanning from left) - it is the smallest index where blance returns to the level before the first a. Hence the split into parts for S_1 and S_2 is unique.
- by induction on string length, the derivations (and Therfore passe tree) of each of The Two Substrings are unique
- Therefore the whole parse is unique.

So no string has Two different parse trees is unambiguous.

(c) Construct a ~~mon~~ non - ambiguous grammar that generates the same language.

The given grammar itself is already non-ambiguous you can present an equivalant (one maybe clearer) unambiguous grammar.

$S \to TS / \epsilon$

$T \to aSb$

T generates one matched pair a.....b where the ..... is a balanced substring (generated by S) and then $S \to TS$ allows concatenation of such matched blocks. This grammar is the same as the original $S \to aSbS / \epsilon$ but written to emphasize the unique decomposition $S = (aSb) S$ - hence unambiguous.

Qus 3 PDA Construction for Balanced Expressions

Construct a PDA that acepets the language $L = \{a^n b^n \mid n \geq 0\}$

a) Define the PDA as a 7 - tuple ($Q \cdot \Sigma \cdot T \cdot S \cdot q^0 \cdot z^0 F$)

let $M = (Q \cdot \Sigma \; T \; S \; q^0 \; z_0 F)$ where:

- $Q = \{q^0 \cdot q_1, q_f\}$
- $\Sigma = \{a \cdot b\}$
- $T = \{z_0, x\} \cdot z_0$

Marker for each a
- q_0 is start state
- Z_0 is initial stack symbol
- $F = \{q_f\}$ (accept by final state)

In q_0 push one x for each a. one seeing first b go to q_1 and pop one x per b. If input finished and stack has only Z_0. go to q_f.

b) Provide the complete transition table.

1) $\delta(q_0, a, Z_0) = \{(q_0, XZ_0)\}$
2) $\delta(q_0, a, X) = \{(q_0, XX)\}$
3) $\delta(q_0, b, X) = \{(q_1, \epsilon)\}$
4) $\delta(q_1, b, X) = \{(q_1, \epsilon)\}$
5) $\delta(q_1, \epsilon, Z_0) = \{(q_f, Z_0)\}$
6) $\delta(q_0, \epsilon, Z_0) = \{(q_f, Z_0)\}$

c) Show step-by step instantaneous descriptions (IDs) for the input "aaabbb".

Start: $(q_0, aaabbb, Z_0)$

1. $(q_0, aaabbb, Z_0)$
2. $\overset{a}{\Rightarrow} (q_0, aabbb, XZ_0)$ " used $\delta(q_0, a, Z_0)$
3. $\overset{a}{\Rightarrow} (q_0, abbb, XXZ_0)$ " $\delta(q_0, a, X)$
4. $\overset{a}{\Rightarrow} (q_0, bbb, XXXZ_0)$ " $\delta(q_0, a, X)$

On reading first b, move to q_1 and pop one X:

5. $\overset{b}{\Rightarrow} (q_1, bb, XXZ_0)$ " $\delta(q_0, b, X) \to (q_1, \epsilon)$

Consume ~~remaining~~ remaining two b's in q_1, popping Xs:

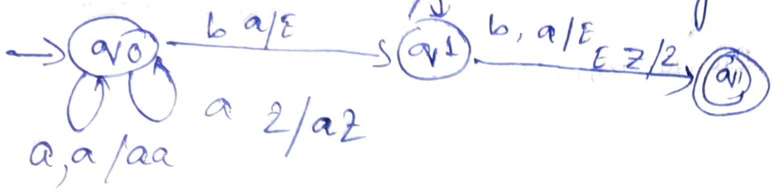
6. $\overset{b}{\Rightarrow} (q_1, b, XZ_0)$ " $\delta(q_1, b, X)$
7. $\overset{b}{\Rightarrow} (q_1, \epsilon, Z_0)$ " $\delta(q_1, b, X)$

Now empty and Stack has Z0 — take E - move to final

8. $\overset{E}{\Rightarrow} (q_f, E, Z_0)$ " $S(q_1, E, Z_0)$

Since in final state q_f with empty input. String accepted

d) Draw the State transition diagram.



Ques 4 Grammar Transformation and Normal forms

Given the CFG : $S \to aA / bB$; $A \to aA / E$: $B \to bB / E$

a) Eliminate - E Production and Unit Productions

Step 1 - find nullable non terminals :

A is nullable (Since $A \to E$)

B is nullable (Since $B \to E$) S is not nullable

Step 2 - add Productions obtained by omitting nullable occurrences.

- From $S \to aA$: because A nullable add $S \to a$.
- From $S \to bB$: because B nullable add $S \to b$.
- From $A \to aA$: because A nullable add $A \to a$.
- From $B \to bB$: because B nullable, add $B - b$.

Step-3 - remove original E - Production : remove $A \to E$ and $B \to E$

No unit Productions (of the from $X \to V$) are Present now

Resulting grammar (no E. no unit):

$S \to aA / a / bB / b$

$A \to aA / a$

$B \to bB / b$

b) Convert the resulting grammar into Chomsky normal form (CNF)

CNF Requires Production of The form $X \to YZ$ (Two nonterminals) or $X \to a$ (Single terminal) Also introduce new nonterminal for terminals when they appear in longer right-hand sides

Introduce terminal nonterminal:

$X\_a \to a$

$X\_b \to b$

Replace Terminals in mixed Production (Length 2 where one Symbol Terminal)

From $S \to aA$ Replace a by X_a: $S \to X\_aA$

From $S \to bB$ replace b by X_b: $S \to X\_bB$

From $A \to aA \to A \to X\_aA$

From $B \to bB \to B \to X\_bB$

Keep terminal only Production:

$S \to a$ and $S \to b$ and $A \to a$ and $B \to b$ are of allowed from

$\Rightarrow$ Final CNF grammar.

$S \to X\_aA$

$S \to X\_bB$

$S \to a$

$S \to b$

$A \to X\_aA$

$A \to a$

$B \to X\_bB$

$B \to b$

$X\_a \to a$

$X\_b \to b$

c) Show ~~deriv~~ derivation of the String "aab" in both The orginal and CNF forms.

Given grammar:

$S \to aA / bB$
$A \to aA / \epsilon$
$B \to bB / \epsilon$

This grammar generates strings of the form a^+ an b^+ only (like 'a', "aa", "bbb Ect) Hence "aab" cannot be drived because it mixes both a's and b's. no rule allows switching from A (a's) to B (b's)

In both original & CNF the same limitition exists, so "aab" is not derivable.

d) Discuss how CNF simplifies PDA simulation & parsing.

- Uniform structure: Every production ~~on~~ is either $A \to BC$ or $A \to a$ making parsing steps systamatic.
- Simple PDA construction: Easier to simulate grammar as stek operations follow patterns.
- Efficient algorithms: CNF allows algorithms like CYK to check string membership efficiently ($O(n^3)$)
- Less ambigaity: Reduces complexity and non determinism during parsing.

Ques 5. CFL membership via pumping lemma prove using the pumping lemma for CFLS that the language $L = \{a^n b^n c^n / n \geq 0\}$ is not context-free

a) State the pumping lemma clearly.
There is a pumping length $P \geq$ such that for every string $S \in L$ with $|S| \geq P$. we can write

$$S = ~~uv~~ uvwxy$$

Satisfying

1) $|vwx| \leq P$.
2) $|vx| > 0$ (i.e. at least one of v, x, is non empty)
3) For all $i \geq 0$, the string $uv^i w x^i y \in L$

we will show this property fails for L. so L is not context-free.

) Choose a string $S \in L$ where $|S| \geq P$ (Pumping length)

Let P be the Pumping length. Choose

$$S = a^P b^P c^P \in L,$$

so $|S| = 3P \geq P$

Assume $S = uvwxy$ satisfies the Pumping-lemma

conditions: $|vwx| \leq p$ and $|vx| > 0$

c) Show some pumped string leaves L

Let $S = a^P b^P c^P$. Since $|vwx| \leq P$. the substring vwx lies fully inside one block (a's / b's / c's) as at most.

- If vwx is within one block pumping adds/removes symbols from only that block → unequal a, b, c, counts.
- If vwx spans two block pumping disturbs this balance while the third block stays unchanged → counts mismatch → not in L.

Hence. for some $i \neq 1$ $uv^i w x^i y \in L$

a) Conclude why L cannot be accepted by any PDA.

Because the Pumping lemma for CFLs is a necessary property of all context free languages and L violates it.

L is not context free by the equivalence of context-free languages and pushdown automata. no PDA can accept L.

Given: $L(G) = \{a^m b^n \mid m > 0, n \geq 0\}$

To Find: Context-free grammar G that generates this language.

Grammar.

Let $G = (V, E, P, S)$ where

- $V = \{S, T\}$
- $E = \{a, b\}$
- S = is the start symbol.

- Productions P:

$S \rightarrow aS / aT$

$T \rightarrow bT / E$

⇒ Explomations

- S ensures at least one (a) is Produced (Since Every derivation Starts with a)
- T generates Zero or more b's

Thus. all Strings have one or more a's folleawed by any number of b.s

⇒ Eg. derivation

$S \Rightarrow aS \Rightarrow aaT \Rightarrow aabT \Rightarrow aabbT \Rightarrow aabbb$

Generated String : $aabbb \in L(G)$

Hence, the grammar Correctly generates

$L(G) = \{a^m b^n \mid m \geq 0,\ n \geq 0\}$

Que 7 is this grammare ambiguous ? If so prove It and Construct a non - ambiguous grammer. That derives That Same language

$S \rightarrow aS / aSbS / \epsilon$

Yes it is ambiguaes . we show two diffrent Prass trees for This String.

Two diffrent Parss trees for ~~aab~~ aaebe

Derivation A

1. $S \Rightarrow aS$
2. $aS \Rightarrow a(aSbS)$
3. $a \Rightarrow (aSbS) \Rightarrow a(aebS)$
4. $a(aebS) \Rightarrow a(aebe) = aaebe$

Parse Tree A (Shape)

```
     S
     |
     a
     |
     S
    /|\
  a S b S
    |   |
    e   e
```

Derivation B

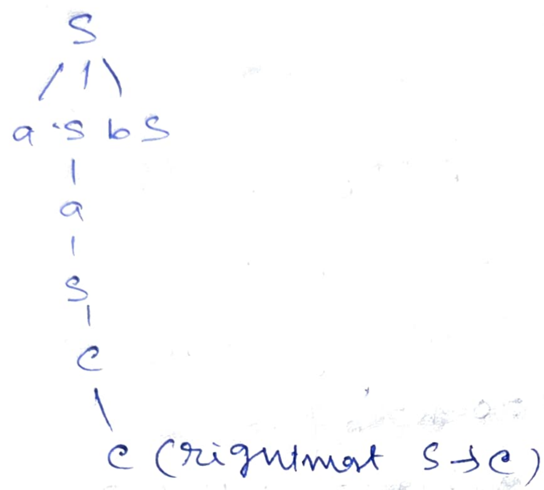
1. $S \Rightarrow aSbS$

2. $aSbS \Rightarrow a(aS)bS$

3. $a(aS)bS \Rightarrow a(ae)bS$

4. $a(ae)bS \Rightarrow a(ae)be = aaebe$

Parse tree B (different Shape):



⇒ The Two Trees are structurally different (in A The left child of the root is 'a' Then a ~~bS~~ subtree aSbS; in B the root expands to aSbS immediately) Therefore the grammar is ambiguous (Exists a string with Two distinct parse trees)

⇒ A grammar is ambiguous if some string has two different Parse trees We exhibited two distinct parse trees for aaebe Hence The grammer is ambiguous.

Ques 8 give the CFG $G = (\{S, A, B\}, \{a\}, \{S \to A,\ A \to B,\ B \to a\}, S)$. Remove unit Production and rewrite the grammar.

Given CFG:

$G := (\{S, A, B\}, \{a\}, P, S)$ with

$P: S \to A,\ A \to B,\ B \to a$

⇒ Remove unit Production:

we have unit chains $S \to A \to B \to a$ and $A \to B \to a$.

Replace Them by direct terminal Production:

- from S follow Chain to terminal a ⇒ add $S \to a$
- from A follow Chain to terminal a ⇒ add $A \to a$
- Keep $B \to a$.

Remove unit rules $S \to A$ and $A \to B$

Resulting grammar (no unit production):

$S \to a$
$A \to a$
$B \to a$

Note: all non terminals still generate a. The languages is {a}

Ques 9 Give the CFG $G = (\{S.A.B\}\{a.b.c\}\{S \to A, A \to aB, B \to c\} S)$ Remove useless productions the updated grammar.

Given:

$G = (\{S, A, B\} \{a, b, c\} P. S)$ with

$P: S \to A \quad A \to ab \cdot \quad B \to c$

Step 1 - find non Terminals that generate Terminals (useful for generating)

- $B \to c$ ⇒ B generates
- $A \to aB$ and B generates ⇒ A generates
- $S \to A$ and A generates ⇒ S generates

So all S. A. B. are generating

Step 2 ⇒ find reachable non terminals from Start. S:

- S is Start (reachable)
- from $S \to A$ ⇒ A reachable
- from $A \to aB$ ⇒ B reachable

So all S. A. B are reachable

⇒ There are no useless Productions (no unreachable or non-generating non terminals)

- Terminal grammar (after removing useless Production)

$S \to A$
$A \to aB$
$B \to c$

Language Produced: Strings of the from ae (Specifically any only "ae")

Ques 10 Convert the given CFG to CNF Consider the given grammar G1:

$S \to a \mid aA \mid B$

$A \to aBB \mid \epsilon$

$B \to Aa \mid b$

Step 1 - Remove ε - Production:

$A \to \epsilon$ is nullable ⇒ update others

$S \to a \mid aA \mid B$

$A \to aBB$

$B \to Aa \mid a \mid b$

Step 2 - Remove unit Production:

$S \to B$ ⇒ Replace with B.S RHS.

$S \to a \mid b \mid aA \mid Aa$

$A \to aBB$

$B \to Aa \mid a \mid b$

Step-3 Convert to CNF from

Introduce $X\_a \to a$ $Y \to BB$

Replace Terminals in long RHS and make binary.

$S \to a \mid b \mid X\_a A \mid A X\_a$

$A \to X\_a Y$

$Y \to BB$

$B \to A X\_a \mid a \mid b$

$X\_a \to a$

Final CNF: all Rules are either $A \to BC$ or $A \to a$