Context-free Languages.

Closure Properties

Th-1. If L_1 and L_2 are context free languages, then their union, L_1+L_2 is also a CFL. That is, the CFL's are closed under union.

Proof. By Example:-

Let L_1 be Palindrome & CFG is;

$$S \to aSa \mid bSb \mid a \mid b \mid \Lambda$$

Let L_2 be $\{a^n b^n\}$ & CFG is

$$S \to aSb \mid \Lambda$$

Then, CFG for L_1+L_2 is,
$$S \to S_1 \mid S_2$$
$$S_1 \to aS_1a \mid bS_1b \mid a \mid b \mid \Lambda$$
$$S_2 \to aS_2b \mid \Lambda$$

Eg 2. $L_1 = S \to aSa \mid bSb \mid \Lambda$ [even Palindrome]

$L_2 = S \to aSa \mid bSb \mid a \mid b$ [odd Palindrome]

$\therefore$ L_1+L_2 (Palindrome) =
$$S \to S_1 \mid S_2$$
$$S_1 \to aS_1a \mid bS_1b \mid \Lambda$$
$$S_2 \to aS_2a \mid bS_2b \mid a \mid b.$$

Eg 3. Let L_1 be Palindrome over the alphabet $\{a, b\}$

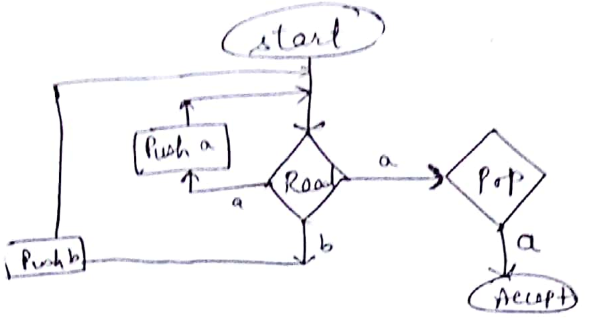
& L_2 be $\{c^n d^n\}$ over the alphabet $\{c, d\}$.

The CFG generated is (L_1+L_2)
$$S \to S_1 \mid S_2$$
$$S_1 \to aS_1a \mid bS_1b \mid a \mid b \mid \Lambda$$
$$S_2 \to cS_2d \mid \Lambda$$

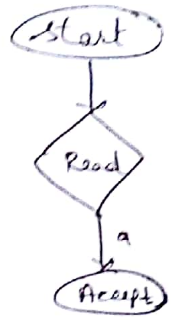
This is a language over the alphabet $\{a\ b\ c\ d\}$

Proof by machines.

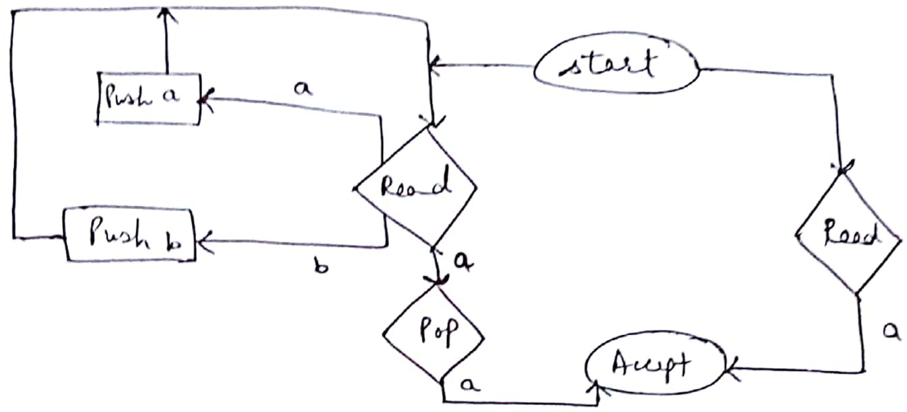
L_1



L_2



$L_1 + L_2$



Th. If L_1 and L_2 are C.F.L's, then so is L_1L_2. That is, the context-free languages are closed under product.

Proof.- Proof by example,

L_1 be Palindrome and CFG_1 be,

$$S \rightarrow aSa \mid bSb \mid a \mid b \mid \Lambda$$

L_2 be $\{a^n b^n\}$ and CFG_2 be,

$$S \rightarrow aSb \mid \Lambda$$

The recommended CFG for the language L_1L_2 is,

$$S \rightarrow S_1 S_2$$
$$S_1 \rightarrow aS_1a \mid bS_1b \mid a \mid b \mid \Lambda$$
$$S_2 \rightarrow aS_2b \mid \Lambda$$

Th: If L is a CFL, then L^* is one too. In other words, the CFL's are closed under the Kleene star.

Proof Example:

$$S \to aSa \mid bSb \mid a \mid b \mid \Lambda$$

then Palindrome* is,

$$S \to XS \mid \Lambda$$
$$X \to aXa \mid bXb \mid a \mid b \mid \Lambda.$$

Intersection

The intersection of two CFL's may or maynot be context-free.

Example (MAY).

If L_1 & L_2 are two CFLs and if L_1 is contained in L_2, then the intersection is L_1 again, which is still context-free, for example,

$$L_1 = \{a^n \text{ for } n = 1\ 2\ 3 \ldots\}$$
$$L_2 = \{ \text{PALINDROME}.$$

L_1 is contained in L_2 $\therefore$ $L_1 \cap L_2 = L_1$ which is context-free.

Example (MAYNOT).

$L_1 = \{a^n b^n a^m$, where $n, m = 1, 2, 3, \ldots$ $n = m$ (not necessarily be the same).

$$CFG_1 = \quad S \to XA$$
$$X \to aXb \mid ab$$
$$A \to aA \mid a.$$

$L_2 = \{a^n b^m a^m,\ n, m = 1, 2, 3, \ldots\}$ $CFG_2 =$ $S \to AX$
$X \to bXa \mid ba$
$A \to aA \mid a$

$L_1 \cap L_2 = L_3 = \{a^n b^n a^n$ for $n = 1, 2, 3, \ldots\}$

because any word in both languages has as many starting a's as middle b's (to be in L_1) & as many middle b's as final a's (to be in L_2).

And L_3 is not Context free language.

Th. The Complement of a CFL may or maynot be context free.

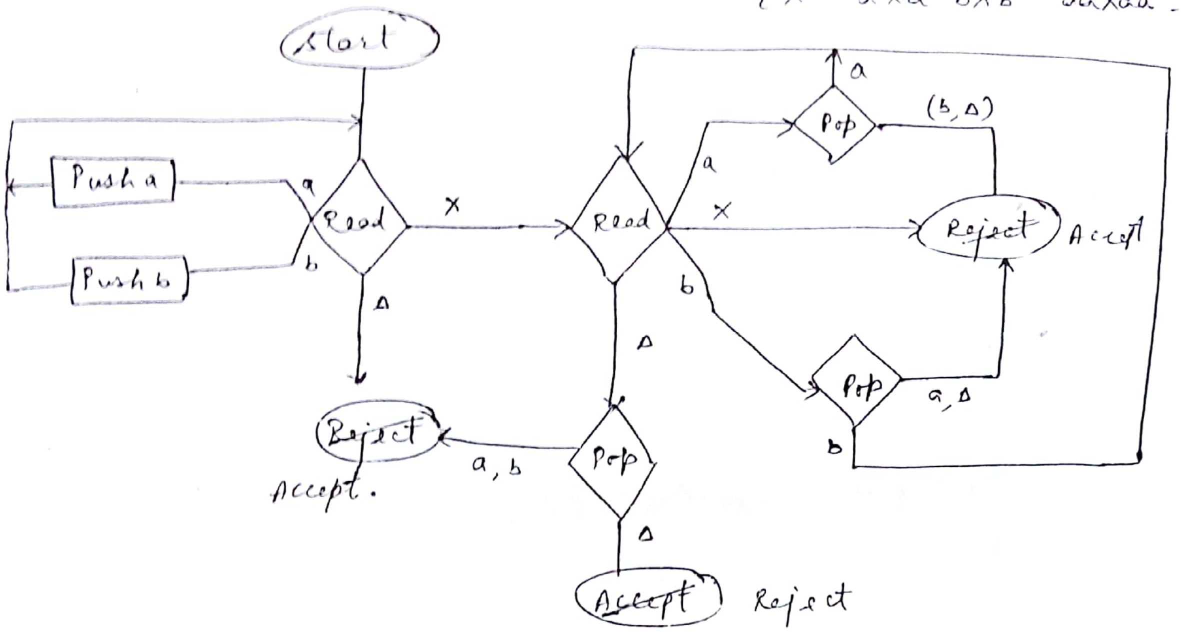
Example (MAY). applies only for deterministic PDA's.

Simply change ACCEPT state to REJECT & vice-versa.

eg: PalindromeX (X in center), Σ = {a b X}

= {w X reverse(w), where w is any string in (a+b)*}

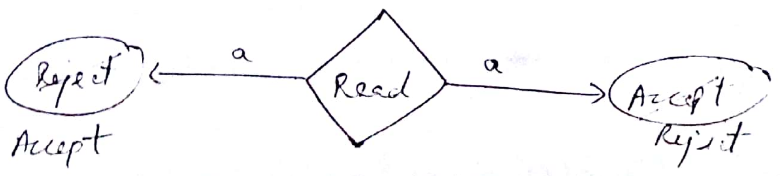
= {X aXa bXb aaXaa ....}



convert accept to reject & vice-versa.
given is the PDA that accepts all strings except PalindromeX.

MAY NOT. (Non-determinism). :-

In " PDA, a word may have two possible paths, the first of which leads to 'Accept' & second which leads to 'Reject'. We accept this word becoz there is atleast one way we can reach to accept state. Now, if we reverse it, there is still a way we can reach accept state. The same word cannot be there in both the languages ie the language itself & its complement, so the halt-status-reversed PDA does not define the complement language.



[same as original]

Pumping Lemma for CFL's.

CNF is :-

Nonterminal → Nonterminal Nonterminal (Live Production)

Nonterminal → terminal (dead Production)

If we are restricted to using the live productions (P) atmost once each then we have (P+1) dead productions.

eg:- $S \Rightarrow XY$

$\Rightarrow aY$

$\Rightarrow aa.$

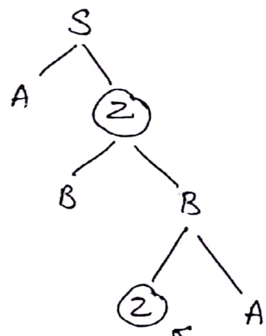
[1 - live
2 - dead].

Tree descendent

1) Suppose, the grammar,

$S \rightarrow AZ$
$Z \rightarrow BB$
$B \rightarrow ZA$
$A \rightarrow a$
$B \rightarrow b$

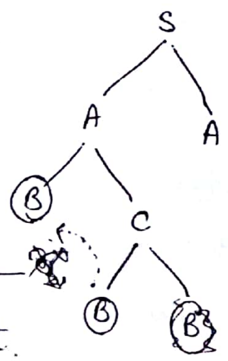
as we proceed with the derivation of some word, we find,



descendent.

2).

$S \rightarrow AA$
$A \rightarrow BC$
$C \rightarrow BB$
$A \rightarrow a$
$B \rightarrow b$

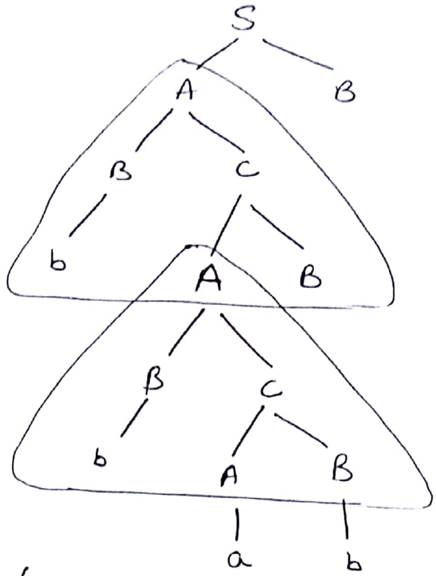


Not a dicendant.

eg:

$S \to AB$
$A \to BC$
$C \to AB$
$A \to a$
$B \to b.$

one Possible derivation is,



Pumping Lemma for CFL's (Theorem 34)

Th. If G is any CFG in CNF with p live productions and w is any word generated by G with length greater than 2^p, then we can break up w into five substrings:
length $(w) > 2^p$

$$w = uvxyz$$ such that x is not Λ & v and y are not both Λ and such that all the words

$uvxyz$
$uvvxyyz$
$uvvvxyyyz$
$uvvvvxyyyyz$

$= uv^n x y^n z$ for $n = 1, 2, 3, \ldots$

can also be generated by G.

Proof:- u = the substring of all the letters of w generated to the left of the Triangle above (this may be Λ)

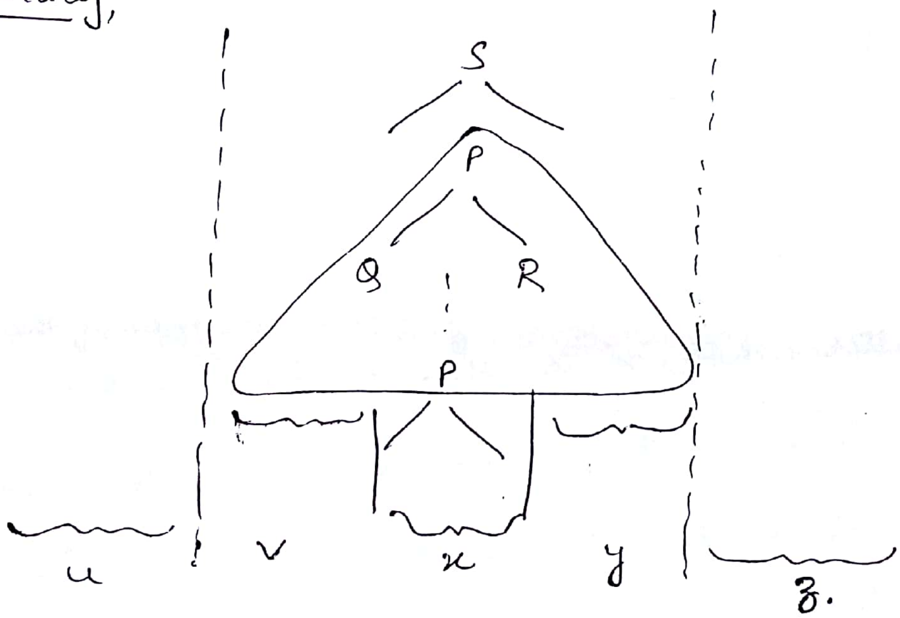
v = the substring of all the letters of w descended from the first P but to the left of the letters generated by the second P (this may be Λ)

the substring of w descended from the lower P (this may not be Λ becoz this nonterminal must turn into some terminals)

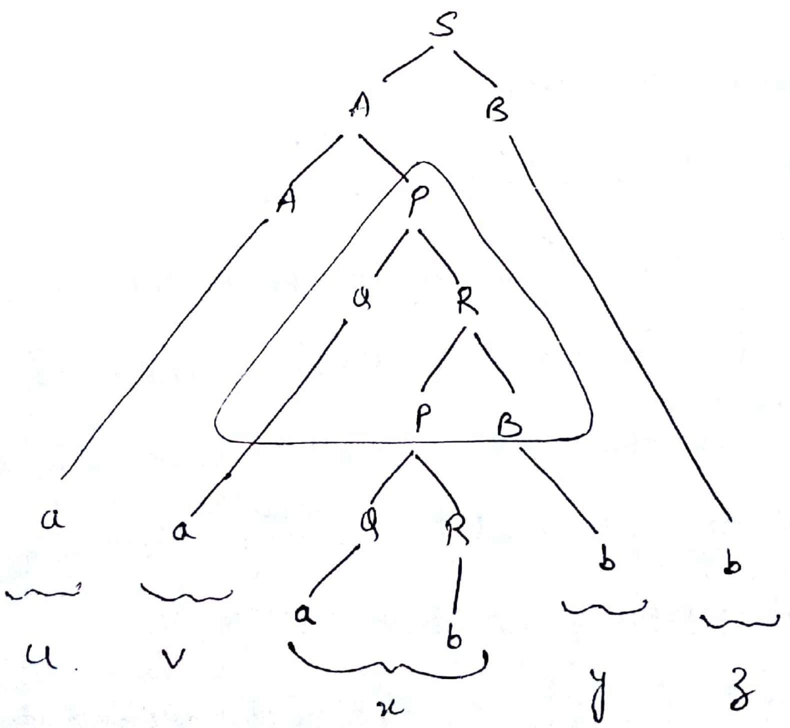
y = the substring of w of all letters generated by the first P but to the right of the letters descending from the second P (this may be Λ, but not if $v = \Lambda$).

z = the substring of all the letters of w generated to the right of the triangle (this may be Λ).

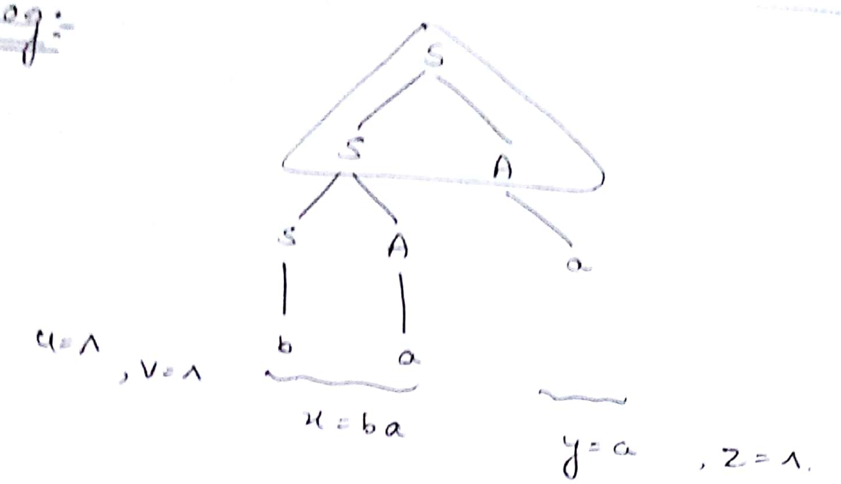
Pictorially,



eg:-



eg:-



Pumping Lemma applies to all CFL's.

eg:- $a^n b^n a^n$ (is not a CFL).

<u>Proof using Pumping Lemma:-</u>

observations:-

1). All words in $\{a^n b^n a^n\}$ have exactly one occurrence of the substring ab no matter what 'n' is, Now if either v-part or y-part has the substring ab in it, then, uv^2xy^2z will have more that one substring of ab & so it cannot be in $\{a^n b^n a^n\}$ ∴ neither v nor y contains ab.

2). similarly for ba substring (as above).

3). The only possibility left is that v & y must be all a's, all b's or Λ, then uv^2xy^2z has increased one or two clumps of solid letters (more a's if v is a's etc). However there are three clumps of solid letters in the word $a^n b^n a^n$ & not all three of those clumps have been increased equally. This would destroy the form of the word.

Therefore, Pumping lemma cannot successfully be applied to the language $\{a^n b^n a^n\}$ at all. But Pumping lemma does apply to all CFL's.

∴ $\{a^n b^n a^n\}$ is not a C.F.language. (Now on Pg-)