UNIT-IV: Pushdown Automata

Objective:

To understand and design push down automata's for a given Context free language.

Syllabus:

Chomsky normal form, Greibach normal form, pumping Lemma for context free languages, closure properties of CFL (proofs not required), applications of CFLs. Push Down Automata: Definition, model of PDA, Instantaneous Description, Language Acceptance of Pushdown Automata, Design of Pushdown Automata

Learning Outcomes:

Students will be able to:

- understand ambiguity in context free grammars.
- minimize the given context free grammar.
- apply Chomsky and Greibach Normal Forms on context free grammars.
- understand and design PDA for given context free languages.

Learning Material

4.1 Chomsky Normal Form :(CNF)

Any context-free language without ε is generated by a grammar in which all productions are of the form A→BC or A→ a. Here, A, B, and C, are variables and a is a terminal.

Step 1: Simplify the grammar.

a) Eliminate ε –productions
b) Eliminate unit productions
c) Eliminate Useless symbols.

The given grammar does not contain ε –productions, unit productions and useless symbols.
It is in optimized form.

Step 2: Consider a production in P,of the form $A\text{->}X_1X_2X_3\ldots..X_m$ where m>=2.If X_i is a terminal a, introduce a new variable C_a and a production C_a->a.Then replace X_i by Ca.

Step 3: Consider a production $A\text{->}B_1B_2B_3\ldots..B_m$ where m>=3,create new variables $D_1,D_2,\ldots.D_{m-2}$ and replace $A\text{->}B_1B_2B_3\ldots B_m$ by the set of productions $\{A\text{->}B_1D_1, D_1\text{->}B_2D_2,\ldots\ldots\ldots Dm\text{-}3\text{->}B_{m-2}\,D_{m-2}, Dm\text{-}2\text{->}B_{m-1}B_m\}$

Example:
Consider the grammar ({S, A, B}, {a, b}, P, S) that has the productions:

$S \rightarrow bA \mid aB$
$A \rightarrow bAA \mid aS \mid a$
$B \rightarrow aBB \mid bS \mid b$

Find an equivalent grammar in CNF.

Step 1: Simplify the grammar.

a) Eliminate ε –productions
b) Eliminate unit productions
c) Eliminate Useless symbols.

The given grammar does not contain ε –productions, unit productions and useless symbols.
It is in optimized form.
Step 2: The only productions already in proper form are $A \rightarrow a$ and $B \rightarrow b$.
So we may begin by replacing terminals on the right by variables, except in the case of the productions $A \rightarrow a$ and $B \rightarrow b$.
$S \rightarrow bA$ is replaced by $S \rightarrow C_bA$ and $C_b \rightarrow b$.
Similarly, $A \rightarrow aS$ is replaced by $A \rightarrow C_aS$ and $C_a \rightarrow a$; $A \rightarrow bAA$ is replaced by $A \rightarrow C_bAA$; $S \rightarrow aB$ is replaced by $S \rightarrow C_aB$;
$B \rightarrow bS$ is replaced by $B \rightarrow C_bS$, and $B \rightarrow aBB$ is replaced by $B \rightarrow C_aBB$.
In the next stage, the production $A \rightarrow C_bAA$ is replaced by $A \rightarrow C_bD_1$ and $D_1 \rightarrow AA$, and the production $B \rightarrow C_aBB$ is replaced by $B \rightarrow C_aD_2$ and $D_2 \rightarrow BB$.
Step 3: The productions for the grammar in CNF are :

$S \rightarrow C_bA \mid C_aB$ $\quad D_1 \rightarrow AA$
$A \rightarrow C_aS \mid C_bD_1 \mid a$ $\quad D_2 \rightarrow BB$
$B \rightarrow C_bS \mid C_aD_2 \mid b$ $\quad C_a \rightarrow a$
$C_b \rightarrow b$

4.2 Greibach Normal Form:

Every context-free language L without e can be generated by a grammar for which every production is of the form $A \rightarrow a\alpha$, where A is a variable, a is a terminal, and α is a (possibly empty) string of variables.
Lemma 1: Define an A-production to be a production with variable A on the left. Let G = (V, T, P, S) be a CFG. Let $A \rightarrow \alpha 1 B \alpha 2$ be a production in P and $B \rightarrow \beta 1 \mid \beta 2 \mid \ldots\ldots \mid \beta r$ be the set of all B-productions. Let G1 = (V, T, P1, S) be obtained from G by deleting the production $A \rightarrow \alpha 1 B \alpha 2$ from P and adding the productions $A \rightarrow \alpha 1 \beta 1 \alpha 2 \mid \alpha 1 \beta 2 \alpha 2 \mid \ldots\ldots \mid \alpha 1 \beta r \alpha 2$. Then L(G) = L(G1).
Lemma 2: Let G = (V, T, P, S) be a CFG. Let $A \rightarrow A\alpha 1 \mid A\alpha 2 \mid \ldots\ldots \mid A\alpha r$ be the set of A-productions for which A is the leftmost symbol of the right-hand side. Let $A \rightarrow \beta 1 \mid \beta 2 \mid \ldots\ldots \mid \beta s$ be the remaining A-productions. Let G1 = (V U {B}, T, P1, S) be the CFG formed by adding the variable B to V and replacing all the A-productions by the productions:

$$1) \left.\begin{array}{l} A \rightarrow \beta_i \\ A \rightarrow \beta_i B \end{array}\right\} 1 \leq i \leq s, \qquad 2) \left.\begin{array}{l} B \rightarrow \alpha_i \\ B \rightarrow \alpha_i B \end{array}\right\} 1 \leq i \leq r.$$

Then L(G1) = L(G).

Example:

Convert to Greibach normal form the grammar

G=i{A1,A2,A3}, {a, b}, P A1),

where P consists of the following:

$$A_1 \to A_2 A_3$$
$$A_2 \to A_3 A_1 | b$$
$$A_3 \to A_1 A_2 | a$$

Step 1 Since the right-hand side of the productions for A_1 and A_2 start with terminals or higher-numbered variables, we begin with the production $A_3 \to A_1 A_2$ and substitute the string $A_2 A_3$ for A_1. Note that $A_1 \to A_2 A_3$ is the only production with A_1 on the left.

The resulting set of productions is:

$$A_1 \to A_2 A_3$$
$$A_2 \to A_3 A_1 | b$$
$$A_3 \to A_2 A_3 A_2 | a$$

Since the right side of the production $A_3 \to A_2 A_3 A_2$ begins with a lower-numbered variable, we substitute for the first occurrence of A_2 both $A_3 A_1$ and b. Thus $A_3 \to A_2 A_3 A_2$ is replaced by $A_3 \to A_3 A_1 A_3 A_2$ and $A_3 \to bA_3 A_2$. The new set is

$$A_1 \to A_2 A_3$$
$$A_2 \to A_3 A_1 | b$$
$$A_3 \to A_3 A_1 A_3 A_2 | bA_3 A_2 | a$$

We now apply Lemma 2 to the productions

$$A_3 \to A_3 A_1 A_3 A_2 | bA_3 A_2 | a.$$

Symbol B_3 is introduced, and the production $A_3 \to A_3 A_1 A_3 A_2$ is replaced by $A_3 \to bA_3 A_2 B_3$, $A_3 \to aB_3$, $B_3 \to A_1 A_3 A_2$, and $B_3 \to A_1 A_3 A_2 B_3$. The resulting set is

$$A_1 \to A_2 A_3$$
$$A_2 \to A_3 A_1 | b$$
$$A_3 \to bA_3 A_2 B_3 | aB_3 | bA_3 A_2 | a$$
$$B_3 \to A_1 A_3 A_2 | A_1 A_3 A_2 B_3$$

Step 2 Now all the productions with A_3 on the left have right-hand sides that start with terminals. These are used to replace A_3 in the production $A_2 \to A_3 A_1$ and then the productions with A_2 on the left are used to replace A_2 in the production $A_1 \to A_2 A_3$. The result is the following.

$$
\begin{array}{ll}
A_3 \to bA_3 A_2 B_3 & A_3 \to bA_3 A_2 \\
A_3 \to aB_3 & A_3 \to a \\
A_2 \to bA_3 A_2 B_3 A_1 & A_2 \to bA_3 A_2 A_1 \\
A_2 \to aB_3 A_1 & A_2 \to aA_1 \\
A_2 \to b & \\
A_1 \to bA_3 A_2 B_3 A_1 A_3 & A_1 \to bA_3 A_2 A_1 A_3 \\
A_1 \to aB_3 A_1 A_3 & A_1 \to aA_1 A_3 \\
A_1 \to bA_3 & \\
B_3 \to A_1 A_3 A_2 & B_3 \to A_1 A_3 A_2 B_3
\end{array}
$$

Step 3 The two B_3-productions are converted to proper form, resulting in 10 more productions. That is, the productions

$$B_3 \to A_1 A_3 A_2 \quad \text{and} \quad B_3 \to A_1 A_3 A_2 B_3$$

are altered by substituting the right side of each of the five productions with A_1 on the left for the first occurrences of A_1. Thus $B_3 \to A_1 A_3 A_2$ becomes

$$B_3 \to bA_3 A_2 B_3 A_1 A_3 A_3 A_2, \qquad B_3 \to aB_3 A_1 A_3 A_3 A_2.$$

$$B_3 \to bA_3 A_3 A_2, \qquad B_3 \to bA_3 A_2 A_1 A_3 A_3 A_2, \qquad B_3 \to aA_1 A_3 A_3 A_2.$$

The other production for B_3 is replaced similarly. The final set of productions is

$$
\begin{array}{ll}
A_3 \to bA_3 A_2 B_3 & A_3 \to bA_3 A_2 \\
A_3 \to aB_3 & A_3 \to a \\
A_2 \to bA_3 A_2 B_3 A_1 & A_2 \to bA_3 A_2 A_1 \\
A_2 \to aB_3 A_1 & A_2 \to aA_1 \\
A_2 \to b & \\
A_1 \to bA_3 A_2 B_3 A_1 A_3 & A_1 \to bA_3 A_2 A_1 A_3 \\
A_1 \to aB_3 A_1 A_3 & A_1 \to aA_1 A_3 \\
A_1 \to bA_3 & \\
B_3 \to bA_3 A_2 B_3 A_1 A_3 A_3 A_2 B_3 & B_3 \to bA_3 A_2 B_3 A_1 A_3 A_3 A_2 \\
B_3 \to aB_3 A_1 A_3 A_3 A_2 B_3 & B_3 \to aB_3 A_1 A_3 A_3 A_2 \\
B_3 \to bA_3 A_3 A_2 B_3 & B_3 \to bA_3 A_3 A_2 \\
B_3 \to bA_3 A_2 A_1 A_3 A_3 A_2 B_3 & B_3 \to bA_3 A_2 A_1 A_3 A_3 A_2 \\
B_3 \to aA_1 A_3 A_3 A_2 B_3 & B_3 \to aA_1 A_3 A_3 A_2
\end{array}
$$

4.3 Pumping Lemma for CFL's:

Let L be any CFL. Then there is a constant n, depending only on L, such that if z is in L and $|z| \geq n$, then we may write z = uvwxy such that

1) $|vx| \geq 1$,
2) $|vwx| \leq n$, and
3) for all $i \geq 0$ uv^iwx^iy is in L.

Example:

Consider the language $L = \{a^ib^ic^i \mid \geq 1\}$. Suppose L were context free and let n be the constant.

Consider $z = a^nb^nc^n$. Write z = uvwxy so as to satisfy the conditions of the pumping lemma.

Since $|vwx| \leq n$, it is not possible for vx to contain instances of a's and c's, because the rightmost a is n + 1 positions away from the leftmost c.

If v and x consist of a's only, then uwy (the string uv^iwx^iy with i = 0) has n b's and n c's but fewer than n a's since $|vx| \geq 1$.

Thus, uwy is not of the form $a^ib^ic^i$. But by the pumping lemma vwy is in L, a contradiction.

The cases where v and x consist only of b's or only of c's are disposed of similarly.

If vx has a's and b's, then uwy has more c's than a's or b's, and again it is not in L.

If vx contains b's and c's, a similar contradiction results.

We conclude that L is not a context-free language.

4.4 Closure Properties of CFL's:

- Context-free languages are closed under union, concatenation and Kleene closure.
- The context-free languages are closed under substitution.
- The CFL's are closed under homomorphism.
- The CFL's are not closed under intersection.
- The CFL's are not closed under complementation.

Applications of the pumping lemma:

The pumping lemma can be used to prove a variety of languages not to be context free, using the same "adversary" argument as for the regular set pumping lemma.

4.5 Push down automata:

A *pushdown automaton* M is a system $(Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where

1) Q is a finite set of *states*;
2) Σ is an alphabet called the *input alphabet*;
3) Γ is an alphabet, called the *stack alphabet*;
4) q_0 in Q is the *initial state*;
5) Z_0 in Γ is a particular stack symbol called the *start symbol*;
6) $F \subseteq Q$ is the set of *final states*;
7) δ is a mapping from $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma$ to finite subsets of $Q \times \Gamma^*$.

Moves:

The interpretation of

$$\delta(q, a, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots, (p_m, \gamma_m)\}$$

where q and p_i, $1 \leq i \leq m$, are states, a is in Σ, Z is a stack symbol, and γ_i is in Γ^*, $1 \leq i \leq m$, is that the PDA in state q, with input symbol a and Z the top symbol on the stack can, for any i, enter state p_i, replace symbol Z by string γ_i, and advance the input head one symbol. We adopt the convention that the leftmost symbol of γ_i will be placed highest on the stack and the rightmost symbol lowest on the stack. Note that it is not permissible to choose state p_i and string γ_j for some $j \neq i$ in one move.

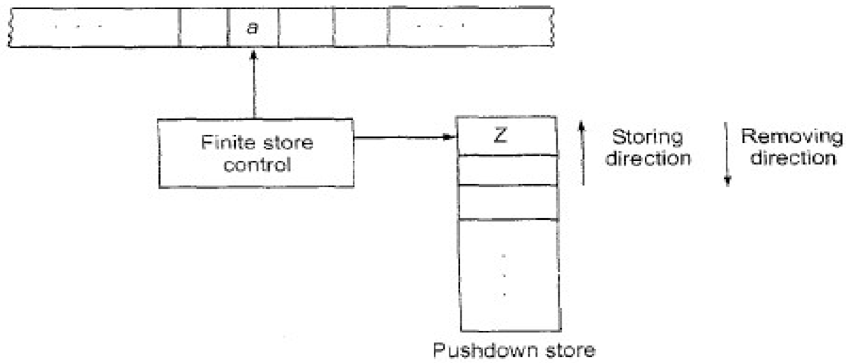
The interpretation of

$$\delta(q, \epsilon, Z) = \{(p_1, \gamma_1), (p_2, \gamma_2), \ldots, (p_m, \gamma_m)\}$$

is that the PDA in state q, independent of the input symbol being scanned and with Z the top symbol on the stack, can enter state p_i and replace Z by γ_i for any i, $1 \leq i \leq m$. In this case, the input head is not advanced.

Model of PDA:

- Pushdown automaton has a read-only input tape, an input alphabet a finite state control, a set of final states, and an initial state as in the case of an FA.
- In addition to these, it has a stack called the pushdown store. It is a read-write pushdown store as we can add elements to PDS or remove elements from PDS.
- A finite automaton is in some state and on reading, an input symbol moves to a new state.
- The pushdown automaton is also in some state and on reading an input symbol and the topmost symbol in PDS, it moves to a new state and writes (adds) a string of symbols in PDS.



4.6 Instantaneous description:

Instantaneous description (ID) is the configuration of a PDA at a given instant. We define an ID to be a triple (q, w, γ), where q is a state, w a string of input symbols, and γ a string of stack symbols.

If $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ is a PDA, we say $(q, aw, Z\alpha) \vdash_M (p, w, \beta\alpha)$ if $\delta(q, a, Z)$ contains (p, β). Note that a may be ϵ or an input symbol.

Accepted Languages:

For PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ we define $L(M)$, the *language accepted by final state*, to be

$$\{w \mid (q_0, w, Z_0) \overset{*}{\vdash} (p, \epsilon, \gamma) \text{ for some } p \text{ in } F \text{ and } \gamma \text{ in } \Gamma^*\}.$$

We define $N(M)$, the *language accepted by empty stack* (or *null stack*) to be

$$\{w \mid (q_0, w, Z_0) \overset{*}{\vdash} (p, \epsilon, \epsilon) \text{ for some } p \text{ in } Q\}.$$

When acceptance is by empty stack, the set of final states is irrelevant, and, in this case, we usually let the set of final states be the empty set.

***Example*:**

Design a PDA that accepts $\{ww^R \mid w \text{ in } (0+1)^*\}$

L = { ε, 0, 1, 00, 11, 0110, 1001, ...}

Let M= $(Q, \sum, \Gamma, \delta, q_0, Z_0, F)$ be the PDA

Consider M = $(\{q1, q2\}, \{0, 1\}, \{0, 1, Z_0\}, \delta, q_1, Z_0, \emptyset)$

$\delta(q1,0,Z_0)=\{(q1,0Z_0)\}$
$\delta(q1,1,Z_0)=\{(q1,1Z_0)\}$
$\delta(q1,0,0)=\{(q1,00),(q2, \varepsilon)\}$
$\delta(q1,1,0)=\{(q1,10)\}$
$\delta(q1,0,1)=\{(q1,01)\}$
$\delta(q1,1,1)=\{(q1,11),(q2, \varepsilon)\}$
$\delta(q2,0,0)=\{(q2, \varepsilon)\}$
$\delta(q2,1,1)=\{(q2, \varepsilon)\}$
$\delta(q1,\varepsilon,Z_0)=\{(q2, \varepsilon \}$
$\delta(q2,\varepsilon,Z_0)=\{(q2, \varepsilon)\}$

4.7 Deterministic PDA:

The PDA is deterministic in the sense that at most one move is possible from any ID.

Formally we say a PDA M is deterministic if:

1) for each q in Q and Z in Γ, whenever $\delta(q, \epsilon, Z)$ is nonempty, then $\delta(q, a, Z)$ is empty for all a in Σ;
2) for no q in Q, Z in Γ, and a in $\Sigma \cup \{\epsilon\}$ does $\delta(q, a, Z)$ contain more than one element.