Example 6.44 Show that $L = \{a^n b^n c^n$ where $n \geq 1\}$ is not context free.

Solution:

Step I: Assume that the language set L is a CFL. Let n be a natural number obtained by using the pumping lemma.

Step II: Let $z = a^n b^n c^n$. So, $| z | = 3n > n$. According to the pumping lemma for CFL, we can write $z = uvwxy$, where $| vx | \geq 1$.

Step III: $uvwxy = a^n b^n c^n$. As $1 \leq | vx | \leq n$, ($| vwx | \leq n$, so $| vx | \leq n$) v or x cannot contain all the three symbols a, b, and c. So, v or x will be in any of the following forms.

1. Contain only a and b, i.e., in the form $a^i b^i$.
2. Or contain only b and c, i.e., in the form $b^i c^i$.
3. Or contain only the repetition of any of the symbols among a, b, and c.

Let us take the value of k as 2. v^2 or x^2 will be in the form $a^i b^i a^i b^i$ (as v is a string here aba is not equal to $a^2 b$ or ba^2) or $b^i c^i b^i c^i$. So, $uv^2 wx^2 y$ cannot be in the form $a^m b^m c^m$. So, $uv^2 wx^2 y \notin L$.

If v or x contains repetition of any of the symbols among a, b, and c, then v or x will be any of the form of a^i ,b^i, or c^i. Let us take the value of k as 0.$uv^0 wx^0 y = uwy$. In the string, the number of occur-of one of the other two symbols in uvy is less than n. So, $uv^2 wx^2 y \notin L$.

Example 6.45 Prove that the language $L = \{a^{i2}/i \geq 1 \}$ is not context free.

Solution:

Step I: Assume that the language set L is a CFL. Let n be a natural number obtained by using the pumping lemma.

Step II: Let $z = a^{i2}$. So, $| z | = 2^i$. Let $2^i > n$. According to the pumping lemma for CFL, we can write $z = uvwxy$, where $| vx | \geq 1$ and $| vwx | \leq n$.

Step III: The string z contains only 'a', and so v and x will also be a string of only 'a'. Let v = ap and $x = a^q$, where $(p + q) \geq 1$. Since $n \geq 0$ and $uvwxy = 2^i$, $| uv^n wx^n y | = | uvwxy | + | v^{n-1} x^{n-1} | = 2^i + (p + q)(n - 1)$. As $uv^n wx^n y \in L$, $| uv^n wx^n y |$ is also a power of 2, say 2^j.

$$(p + q)(n - 1) = 2^j - 2^i$$
$$\Rightarrow (p + q)(n - 1) + 2^i = 2^j$$
$$\Rightarrow ((p + q)2^{i+1} + 2^i = 2^j$$
$$\Rightarrow (2^i (2(p + q) + 1) = 2^j$$

(p + q) may be even or odd, but 2(p + q) is always even. However, 2(p + q) + 1 is odd, which cannot be
a power of 2. Thus, L is not context free.

Example 6.46 Prove that $L = \{0^p$/where p is prime$\}$ is not a CFL.

Solution:

Step I: Suppose L = L(G) is context free. Let n be a natural number obtained from the pumping lemma
for CFL.

Step II: Let p be a number > n, $z = 0^p \in L$. By using the pumping lemma for CFL, we can write $z = uvwxy$, where $|vx| \geq 1$ and $|vwx| \leq n$.

Step III: Let k = 0. From the pumping lemma for CFL, we can write uv^0wx^0y, i.e., $uwy \in L$. As uvy is in the form 0^p, where p is prime, $|uwy|$ is also a prime number. Let us take it as q. Let $|vx| = r$. Then, $|uvqwxqy| = q + qr = q(1 + r)$. This is not prime as it has factors $q(1 + r)$ including 1 and $q(1 + r)$. So, $uv^qwx^qy \notin L$. This is a contradiction. Therefore, L is not context free.

6.10 Ogden's Lemma for CFL

If L is a context-free language, then there exists some positive integer p such that for any string $w \in L$ with at least p positions marked, then w can be written as

$$w = uvxyz$$

with strings u, v, x, y, and z, such that

1. x has at least one marked position,
2. either u and v both have marked positions or y and z both have marked positions,
3. vxy has at most p marked positions, and
4. uv^ixy^iz is in L for every $i \geq 0$.

Ogden's lemma can be used to show that certain languages are not context free, in cases where the pumping lemma for CFLs is not suffi cient. An example is the language, $\{a^ib^jc^kd^l: i = 0 \text{ or } j = k = l\}$. It is also useful to prove the inherent ambiguity of some languages.

6.11 Decision Problems Related to CFG

Decision problems are the problems which can be answered in a 'yes' or 'no'. There are a number of decision problems related to CFG. The following are some of them.

1. Whether a given CFG is empty or not. (Does it generate any terminal string?)
2. Whether a given CFG is fi nite? (Does it generate up to a certain length word?)
3. Whether a particular string w is generated by a given CFG. (Membership)
4. Whether a given CFG is ambiguous.
5. Whether two given CFG have a common word.
6. Whether two different CFG generate the same language.
7. Whether the complement of a given CFL is also a CFL.

In the following section, we are discussing the fi rst three decision algorithms for CFG.

6.11.1 Emptiness

In a CFG $\{V_N, \Sigma, P, S\}$, if S generates a string of terminal, then it is non-empty; otherwise, the corresponding CFL is empty.

The following algorithm checks whether a CFG is empty or not:

While (there exist a production in the form $NT_i \rightarrow T$/String of T)

{
Replace each NT at the right hand side of each production by corresponding T or string of T.
}
if (S generates a string of terminal)
return non-empty.
else
empty.

Example 6.47 Check whether the following CFG is empty or not.

$$S \rightarrow AB/D$$
$$A \rightarrow aBC$$
$$B \rightarrow bC$$
$$C \rightarrow d$$
$$D \rightarrow CD$$

Solution: C → d is in the form NT → T. Replacing C → d in A → aBC, B → bC, and D → CD, we get

$$S \rightarrow AB/D$$
$$A \rightarrow aBd$$
$$B \rightarrow bd$$
$$D \rightarrow dD$$

B →bd is in the form NT →string of terminals. Replacing B →bd in S →AB and A →aBd, we get

$$S \rightarrow Abd/D$$
$$A \rightarrow abdd$$
$$D \rightarrow dD$$

A →abdd is in the form NT →String of terminals. Replacing A →abdd in S →Abd, we get

$$S \rightarrow abddbd/D$$
$$D \rightarrow dD$$

As S generates a string of terminals, the CFG is non-empty.

Example 6.48 Check whether the following CFG is empty or not.

$$S \rightarrow ABD$$
$$A \rightarrow BC$$
$$B \rightarrow aC$$
$$C \rightarrow b$$
$$D \rightarrow BD$$

Solution: C →b is in the form NT →T. Replacing C →b in A →BC, B →aC, we get

$$S \rightarrow ABD$$
$$A \rightarrow Bb$$
$$B \rightarrow ab$$
$$D \rightarrow BD$$

B →ab is in the form NT →String of T. Replacing B →ab in S →ABD, A →Bb, D →BD, we get

S →AabD
A →abb
D →abD

A →abb is in the form NT →String of T. Replacing A →abb in S →AabD, we get

S →abbabD
D →abD

'While' part is complete, S does not produce any terminal string. Thus, the CFG is empty.

6.11.2 Finiteness

A language L generated from a given CFG is finite if there are no cycles in the directed graph generated from the production rules of the given CFG. The longest string generated by the grammar is determined by the derivation from the start symbol.
(The number of vertices of the directed graph is the same as the number of non-terminals in the grammar. If there is a production rule S →AB, the directed graph is as shown in Fig. 6.15.)

6.11.2.1 Infiniteness

A language L generated from a given CFG is infinite if there is at least one cycle in the directed graph generated from the production rules of the given CFG.
The following examples describe this in detail.

Example 6.49 Verify whether the languages generated by the following grammar are finite or not. If finite, find the longest string generated by the grammar.

a) S →AB	b) S →AB
A →BC	A →B
B →C	B →SC/a
C →a	C →AB/b

Soution:
a) The grammar is not in CNF. Removing the unit productions, the grammar becomes

S →AB
A →BC
B →a
C →a

Now the grammar is in CNF. The directed graph for the grammar is shown in Fig. 6.16.