Pumping Lemma for Context Free Language

The pumping lemma for CFL gives a necessary and sufficient condition for a language to be context free. The lemma states that there are always 2 short substrings close together, that can be pumped or repeated, both the same no. of times as often as we like. It is used to prove certain languages not context free and also in developing algorithms to determine finiteness and infiniteness of CFLs.

Pumping Lemma for CFLs

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 write $z = uvwxy$ such that ~~[illegible]~~

(1) $|vx| \geq 1$

(2) $|vwx| \leq n$

(3) for all $i \geq 0$, uv^iwx^iy is in L.

Proof

Let G be a CFG in CNF generating $L - \varepsilon$. If we consider a word z in $L(G)$ which is long, then any parse tree for z must contain a long path. It can be shown by induction that if the parse tree of a word generated by a CNF grammar has no path of length greater than i, then the word length is not greater than 2^{i-1}.

i.e. If the length of the longest path in tree is $\leq i$ then word is of length $\leq 2^{i-1}$.

Basis

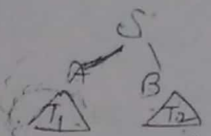
$i=1$ is terminal. When the longest path in tree is of length one, root has only one son whose label is a terminal. So word is of length 1. This tree shown in fig (1)

S
$\downarrow$
a

fig (1)

For induction step let $i \geq 1$

Let the root and sons be as shown in fig (2)



Let T be an S-tree with longest path $\leq i$.
As $i > 1$, root of T has exactly 2 sons A & B
If there are no paths of length $> i-1$ in trees T_1 & T_2,
then trees T_1 & T_2 generate word of 2^{i-2} or fewer symbols. Then the entire tree generates no word longer than 2^{i-1}.

i.e. T_1 & T_2 generates words w_1 & w_2

$|w_1| \leq 2^{i-2}$ $\quad |w_2| \leq 2^{i-2}$

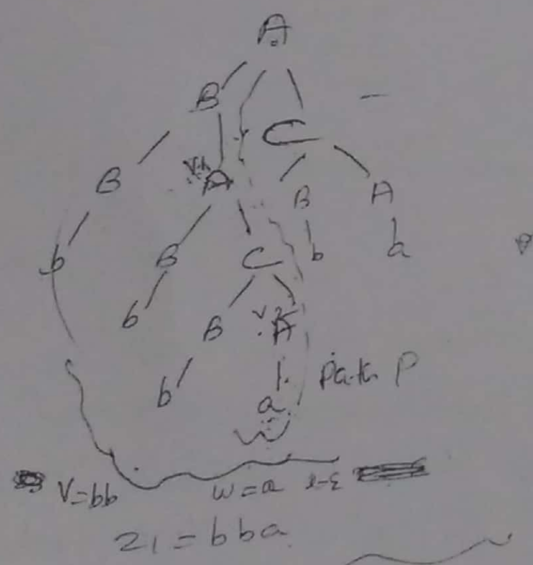
Tree T generates words of length $|w_1 w_2| \leq 2^{i-2} + 2^{i-2}$
$\leq 2^{i-1}$.

Let $G = (V_N, T, P, S)$ be a CNF grammar which has k variables and let $n = 2^k$.
i.e $|V_N| = k$. To prove that n is the required no, start with $z \in L(G)$ i.e. $|z| \geq n \geq 2^k$ and construct a derivation tree for z. We know that if the length of the longest path in T is atmost k, then $|z| \leq 2^{k-1}$. But $|z| \geq 2^k > 2^{k-1}$.
So any parse tree for z has a path of length atleast $k+1$. But such path has $k+2$ vertices and only the last vertex is a leaf. Thus all labels except last are variables. As $|V_N| = k$, some label is repeated twice on the path.

We choose a repeated label as follows:
Let P be a longer path in tree. Then there must be 2 vertices v_1 & v_2 on path satisfying the foll conditions

1) The vertices v_1 & v_2 both have same label
2) vertex v_1 is closer to root than vertex v_2
3) The portion of path from v_1 to leaf is of length atmost $k+1$.

To find v_1 & v_2 proceed up path P from leaf keeping track of labels encountered. Of the first $k+2$ vertices, only leaf has a terminal. The remaining $k+1$ vertices cannot have distinct variables.
For better understanding, we illustrate construction for a grammar with prods $A \to BC$ $B \to BA$ $C \to BA$ $A \to a$ $B \to b$ as in fig (3)

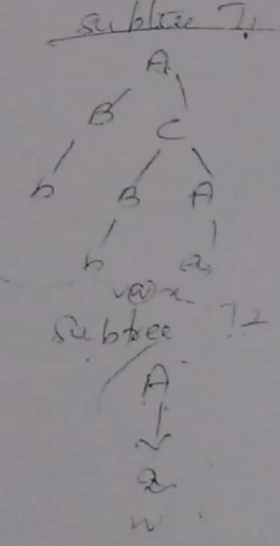


$v = bb$ $w = a$ $x = \varepsilon$

$z_1 = bba$

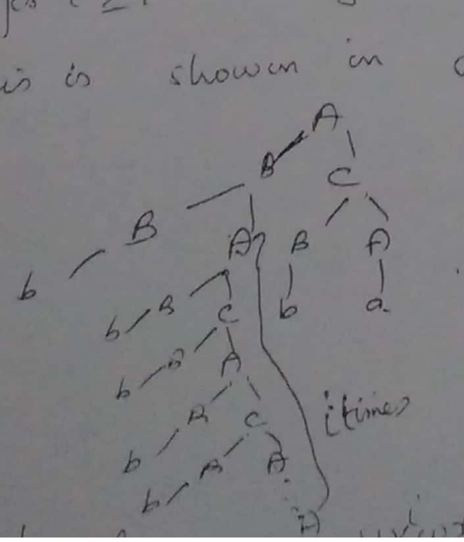
$z = bbbaba = uvwxy$

$u = b$ $y = ba$



u

Let v_1 & v_2 be vertices with label A, v_1 near root
So portion of path from v_1 to leaf has only one label A,
which is repeated so its length is atmost $k+1$
Let T_1 & T_2 be 2 subtrees with v_1 & v_2 as ~~yields~~ roots
and z_1 & z_2 as yields respectively. As P is the
longest path in T, the portion of T from v_1 to
leaf is of longest path in T_1 and is of length
atmost $k+1$ and so $|z_1| \leq 2^k$. z_2 of T_2
is a subtree generated by vertex v_2 and z_2
is the yield of T_2 we write $z_1 = vwx = vz_2x$
where $z_2 = w$ = yield of T_2. Furthermore
v & x cannot be both ε since the first
production used in derivation of z_1 must be
of form $A \to BC$ so $|vx| \geq 1$. As z & z_1 are
yields of T and a proper subtree T_1 of T we
write $z = uz_1y = uvwxy$ with $|vx| \geq 1$ &
$|vwx| \leq n$
As T is a A-tree and T_1 and T_2 are also
A-trees we get $A \overset{*}{\Rightarrow} uAy$
$A \overset{*}{\Rightarrow} vAx$ & $A \overset{*}{\Rightarrow} w$.
As $A \overset{*}{\Rightarrow} uAy \Rightarrow uwy$, $uv^0wx^0y \in L$
$A \overset{*}{\Rightarrow} uAy \Rightarrow uvwxy$). $uvwxy \in L$
Also $A \overset{*}{\Rightarrow} uAy \Rightarrow uvAxy$
$\Rightarrow uvvAxxy \Rightarrow uv^2Ax^2y$
$\Rightarrow uv^2wx^2y$ ie $uv^2wx^2y \in L$
ie for $i \geq 1$ $A \overset{*}{\Rightarrow} uAy \overset{*}{\Rightarrow} uv^iAx^iy \Rightarrow uv^iwx^iy \in L$.
This is shown in fig given below



1. Prove that $L = \{a^i b^i c^i / i \geq 1\}$ is not a CFL

<u>Proof</u>

Let n be a constant of pumping lemma

Select $z = a^n b^n c^n$ This ensures that z is in L and $|z| \geq n$.

If we write $z = uvwxy$, then the possible choices of vx satisfying the conditions $1 \leq |vx| \leq n$ and $|vwx| \leq n$ are

① $vx = a^p$ where $1 \leq p \leq n$ ie vx contains only a's.
For this choice of vx uv^0wx^0y will be $a^{n-p}b^nc^n$ and since $1 \leq p \leq n$ uv^0wx^0y contains less no. of a's than b's and c's. Hence uv^0wx^0y cannot belong to L.
Hence contradiction to pumping lemma.

② $vx = b^p$ $1 \leq p \leq n$ ie vx contain only b's.
For this choice of vx, uv^0wx^0y will be $a^nb^{n-p}c^n$ & since $1 \leq p \leq n$, it contains less no. of b's than a's & c's. Hence uv^0wx^0y cannot belong to L and ∴ contradiction to pumping lemma.

③ $vx = c^p$ $1 \leq p \leq n$
For this choice of vx, uv^0wx^0y will be $a^nb^nc^{n-p}$ ie it contains less no. of c's than a's & b's.
Hence uv^0wx^0y cannot belong to L and so contradiction to pumping lemma.

④ $vx = a^pb^q$ where $1 \leq p, q \leq n$, vx contain both a's and b's.
For this choice of vx, uv^0wx^0y will be $a^{n-p}b^{n-q}c^n$ and since $1 \leq p, q \leq n$, uv^0wx^0y contain less no of a's & b's than c's. Hence $uv^0wx^0y \notin L$ and hence contradiction to pumping lemma.

(5) $vx = b^p c^q$ where $1 \le p, q \le n$

uv^0wx^0y will be $a^n b^{n-p} c^{n-q}$, and since $1 \le p, q \le n$,

uv^0wx^0y contains less no. of b's & c's than a's.

So $uv^0wx^0y \notin L$ and so contradiction to pumping lemma.

We find that vx cannot contain both a's & c's because there are n no. of b's b/w right-most a & leftmost c and hence if we take vx contain both a's & c's, then $|vwx| \le n$ is not satisfied.

Hence we conclude that we cannot have vx such that uv^iwx^iy is in L $\therefore$ It is not a CFL

(2) Prove that $L = \{a^i b^i c^j \mid j \ge i\}$ not a CFL.

Same as Q(1)

but for $vx = a^p b^q$ uv^2wx^2y will be $a^{n+p} b^{n+q} c^n$

and since $1 \le p, q \le n$, uv^2wx^2y contain more no. of a's & b's than c's and hence a contradiction to pumping lemma

(3) Prove that $L = \{a^i \mid i \text{ is prime}\}$ not a CFL.

1. Let n be a constant of pumping lemma,
2. Select $z = a^m$ [$m \ge n$ & m is prime]

This ensures that z is in L and $|z| \ge n$.

3) If we write $z = uvwxy$ satisfying the conditions $1 \le |vx| \le n$ & $|vwx| \le n$, then the possible choices of vx are:

$vx = a^p \quad 1 \le p \le n$

For this choice of vx uv^iwx^iy will be $a^{m-p+ip} = a^{m+(i-1)p}$ and

$|uv^iwx^iy| = m+(i-1)p$ i.e the length of

uv^iwx^iy is $m+(i-1)p$ and for $i=m+1$,
$|uv^iwx^iy| = (m+m+1-1)p = m+mp = m(1+p)$ (2)
which is the product of 2 no:s both greater than 1 - hence not prime ∴ for $i=m+1$,
uv^iwx^iy cannot belong to L, because its length not a prime no. Hence contradiction
to pumping lemma and ∴ L not a CFL.

(4) Prove that $L = \{a^ib^jc^k \mid i<j<k\}$ not a CFL

Let n be a constant of pumping lemma.
Select $z = a^nb^{n+1}c^{n+2}$ This ensures that
z is in L & $|z| \geq n$.
If we write $z = uvwxy$, then the possible
choices of vx satisfying the conditions $1 \leq |vx| \leq n$
and $|vwx| \leq n$ are:

(1) $vx = a^p$ $1 \leq p \leq n$, uv^2wx^2y will be $a^{n+p}b^{n+1}c^{n+2}$
For this choice of vx, uv^2wx^2y contains more or
and since $1 \leq p \leq n$, uv^2wx^2y Hence it cannot
equal no of a's than b's.
belong to L and so contradiction to pumping lemma

(2) $vx = b^p$ $1 \leq p \leq n$ and since $1 \leq p \leq n$,
$uv^0wx^0y = a^nb^{n+1-p}c^{n+2}$
uv^0wx^0y contain less or equal no of b's
than a's. Hence uv^0wx^0y cannot belong to L
and so contradiction to pumping lemma

(3) $vx = c^p$ $1 \leq p \leq n$
uv^0wx^0y will be $a^nb^{n+1}c^{n+2-p}$ i.e. less or equal
no of c's than b's. so contradiction

(4) $vx = a^p b^q \quad 1 \le p, q \le n$

uv^2wx^2y will be $a^{n+p} b^{n+1+q} c^{n+2}$ ie it contains more or equal no of b's than c's. Hence cannot belong to L and so contradiction

(5) $vx = b^p c^q \quad 1 \le p \; q \le n$

uv^0wx^0y will be $a^n b^{n+1-p} c^{n+2-q}$, ~~and~~ Hence ie less or equal no: of b's than a's. So contradiction uv^0wx^0y cannot belong to L and so ~~contradiction~~ L not a CFL

$\therefore uv^iwx^iy \notin L$ and

(5) Prove that $L = \{a^i b^j c^k \mid i \le j \le k\}$ not a CFL.

Same as Q(1) for $vx = b^p$, c^p, $b^p c^q$.

For $vx = a^p \quad 1 \le p \le n$

uv^2wx^2y will be $a^{n+p} b^n c^n$ ie more a's than b's & c's.

so contradiction

For $vx = a^p b^q \quad 1 \le p \; q \le n$

uv^2wx^2y will be $a^{n+p} b^{n+q} c^n$ ie more a's & b's than c's. and hence Hence cannot belong to L

contradiction to pumping lemma

$\therefore L$ not a CFL

(6) Prove that $L = \{a^i b^j \mid j = i^2\}$ not a CFL.

Let n be a constant of pumping lemma

Select $z = a^n b^{n^2}$ This ensures that z is in L and $|z| \ge n$.

If we write $z = uvwxy$ then the possible choices of vx satisfying the conditions $1 \le |vx| \le n$ and $|vwx| \le n$ are :

(1) $vx = a^p$. $1 \le p \le n$

uv^2wx^2y will be $a^{n+p} b^{n^2}$ and since $1 \le p \le n$, uv^2wx^2y contains no. of a's b/w $n+1$ and $n+n = 2n$ whereas no. of b's equal to square of n and hence no. of b's not the square of no. of a's. Hence it cannot belong to L & a contradiction to pumping lemma.

(2) $vx = b^p$ $1 \le p \le n$

uv^2wx^2y will be $a^n b^{n^2+p}$ not square of no. of a's. Hence cannot belong to L and hence contradiction.

(3) $vx = a^p b^q$ $1 \le p, q \le n$

uv^2wx^2y will be $a^{n+p} b^{n^2+q}$ and no. of b's not square of no. of a's even if $p = q$. Hence uv^2wx^2y cannot belong to L and hence contradiction to pumping lemma.

$\therefore$ L not a CFL.

Closure Properties of CFLs

I Context free languages are closed under union, concatenation and kleen closure

i.e. If L_1 & L_2 are the CFLs, $L_1 \cup L_2$ is a CFL, $L_1 L_2$ is a CFL and L^* always a CFL.

Proof

(i) Let L_1 & L_2 be CFLs generated by CFG's $G_1 = (V_1, T_1, P_1, S_1)$ & $G_2 = (V_2, T_2, P_2, S_2)$ respectively. Assume V_1 & V_2 are disjoint.

Let us construct a CFG $G = (V, T, P, S_3)$ using G_1 & G_2 as follows

$$V = V_1 \cup V_2 \cup \{S_3\}$$

$$T = T_1 \cup T_2$$

$$P = P_1 \cup P_2 \cup \{S_3 \to S_1 | S_2\}$$

$L(G)$ will therefore contain those strings that are derivable from S_1 as well as derivable from S_2. Consider a sentence $w \in L_1 \cup L_2$. If $w \in L_1$, $S_3 \Rightarrow S_1 \overset{*}{\Rightarrow} w$. If $w \in L_2$, $S_3 \Rightarrow S_2 \overset{*}{\Rightarrow} w$.

Hence $L(G) = L_1 \cup L_2$.

(ii) Concatention.

Let L_1 & L_2 be 2 CFLs generated by $G_1 = (V_1, T_1, P_1, S_1)$ & $G_2 = (V_2, T_2, P_2, S_2)$ respectively. Construct a CFG $G = (V, T, P, S)$ using G_1 & G_2 as follows.

$$V = V_1 \cup V_2 \cup \{S_4\}$$

$$T = T_1 \cup T_2$$

$$P = P_1 \cup P_2 \cup \{S_4 \to S_1 S_2\}$$

$L(G)$ will therefore contain those strings derivable from S_1, immediately followed by strings derivable from S_2

~~Hence~~ ~~$L(G)$~~ ~~$= L_1 L_2$~~ . i.e. $L = w \in L_1 L_2$.

From G we derive w as $S \Rightarrow S_1 S_2 \overset{*}{\Rightarrow} w_1 S_2 \overset{*}{\Rightarrow} w_1 w_2 = w$

Hence $L(G) = L_1 . L_2$

(3) Kleen closure

Let. L be generated by a CFG $G = (V, T, P, S)$ using G as follow

We construct $G_1 = (V_1, T_1, P_1, S_5)$

$V = V_1 \cup \{S_5\}$

$T_1 = T$

$P_1 = P \cup \{S_5 \rightarrow S_5 S / \varepsilon\}$

Clearly $L(G_1) = L^*$

II Context-free languages are not closed under intersection

~~Consider~~ We know that $L = \{a^i b^i c^i / i \geq 1\}$ is not a CFL (proved using pumping lemma)

Consider the languages $L_1 = \{a^i b^i c^j, i, j \geq 1\}$

$L_2 = \{a^i b^j c^j, i, j \geq 1\}$

Both these languages are CFLs because there exists CFGs generating L_1 & L_2 respectively.

CFG for L_1 : $S \rightarrow AB$

$A \rightarrow aAb / ab$

$B \rightarrow cB / c$

$A \rightarrow aAb$
$\rightarrow aaAbb$

CFG for L_2 is $S \rightarrow CD$

$C \rightarrow aC / a$

$D \rightarrow bDc / bc$

We find that $L_1 \cap L_2 = \{a^i b^i c^i / i \geq 1\}$ which is not a CFL thereby proving that CFLs are not closed under intersection

CFLs are closed under union. If they were closed under complementation, by DeMorgan's law $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$ be closed under intersection, contradicting the stmt that CFLs are not closed under intersection ∴ CFLs not closed under complementation.

IV CFLs are closed under Substitution.

Let Σ & Δ be 2 alphabets.
Substitution from Σ to Δ is a mapping which maps each symbol a to a language L_a over Δ.
If $w = a_1 a_2 \ldots a_n$ is a string in Σ,
$s(w) = s(a_1)\, s(a_2) \ldots s(a_n)$.
If L is a language over Σ, $S(L) = \{ S(w) \mid w \in L \}$.

Let L be a CFL over Σ & s be a substitution from Σ to Δ such that for each a, $s(a) = L_a$ is a CFL over Δ. Then $s(L)$ is a CFL over Δ.

Proof ~~Consider~~ Let L be generated by CFG $G = (V, \Sigma, P, S)$. Consider L_a for each a.
Let L_a be generated by CFG $G_a = (V_a, \Delta, P_a, S_a)$
We will Construct G' for $S(L)$ as follows

The variables of G' are all variables of G & G_a's
The terminals of G' are all terminals of G_a's.
The start symbol of G' is start symbol of G;
The productions of G' are all productions of G_a's together with those productions of G in which a terminal a is replaced by S_a (start symbol of G_a)

It is clear that any string generated by G' is of form $w_{a_1} w_{a_2} \ldots w_{a_n}$ where $a_1 a_2 \ldots a_n$ is in L and $w_{a_i} \in L_{a_i}$

eg Consider the language $L = \{12\}$ over $\{1, 2\}$ and the substitution

$$s(1) = \{a^n b^n \mid n \geq 0\} = L_1$$
$$s(2) = \{b^m \mid m \geq 1\} = L_2$$
$$s(12) = \{a^n b^{m+n}, n \geq 0, m \geq 1\} = \{a^i b^j / j > i\}$$

Let L_1 & L_2 be generated by

$G_1 = (V_1, \{a, b\}, P_1, S_1)$ where $P_1: S_1 \to aS_1b / \varepsilon$

and $G_2 = (V_2, \{a, b\}, P_2, S_2)$ where $P_2: S_2 \to bS_2 | b$

grammar for 12 is: $S \to 12$

The grammar G' for $s(12)$ is:

$$S \to S_1 S_2$$
$$S_1 \to aS_1b / \varepsilon$$
$$S_2 \to bS_2 / b$$

V CFLs are closed under homomorphism

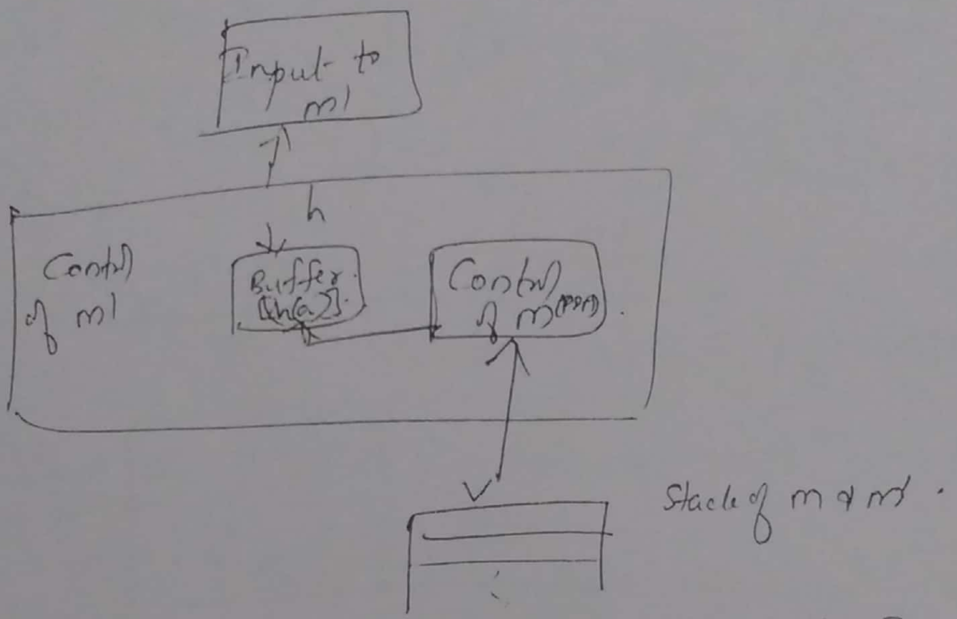
As homomorphism is a special case of substitution, the result is implied by closure under substitution.

VI CFLs are closed under inverse homomorphism

Let h be a homomorphism from Σ to Δ. $h^{-1}(L)$ is a CFL over Σ if L is a CFL over Δ,

Proof Let L be accepted by a PDA $M = (Q, \Delta, \Gamma, \delta, q_0, z_0, F)$. Then PDA M' for $h^{-1}(L)$ constructed as follows:

Given an input $w = a_1 a_2 - a_n$ we will construct $h(w) = h(a_1) h(a_2) - - h(a_n)$ and w accepted by m' iff $h(w)$ accepted by m.
For this purpose m' uses a buffer to hold $h(a)$ for any input symbol a.
The scheme visualized by the diagram



Each state of m' is an ordered pair $[q, x]$ where q is a state from m and x is a suffix of $h(a)$ for an input symbol a. Basically x ~~simulated~~ simulates the buffer. If buffer is non-empty, a state in m' is of form $[q, ay]$ and m' simulates m on state q & i/p a. In this case m' does not consume any external i/p.
If buffer is empty, m' reads the next symbol a and puts $h(a)$ in the buffer without changing the 'state part' and stack.

Formal definition of M' is

$M' = (Q', \Sigma, \Gamma, \delta', q_0', Z, F')$

where

1. $Q' = \{[q, x]$, q in Q and x is a suffix of $h(a)$ for a in Σ

2. δ' is defined using the following

ⓐ $\delta'([q, \epsilon], a, Y) = ([q, h(a)], Y)$

ⓑ $\delta'([q, ax], \epsilon, Y) = ([p, x], \gamma)$
if $\delta(q, a, Y) = (p, \gamma)$ for all all
$q \in Q$, $a \in \Delta$ and $Y \in \Gamma$.

ⓒ $\delta'([q, x], \epsilon, Y) = ([p, x], \gamma)$ if
$\delta(q, \epsilon, Y) = (p, \gamma)$

③ q_0' the start state is $[q_0, \epsilon]$

④ F', final state $= \{(p, \epsilon) / \text{for each } p \text{ in } F\}$

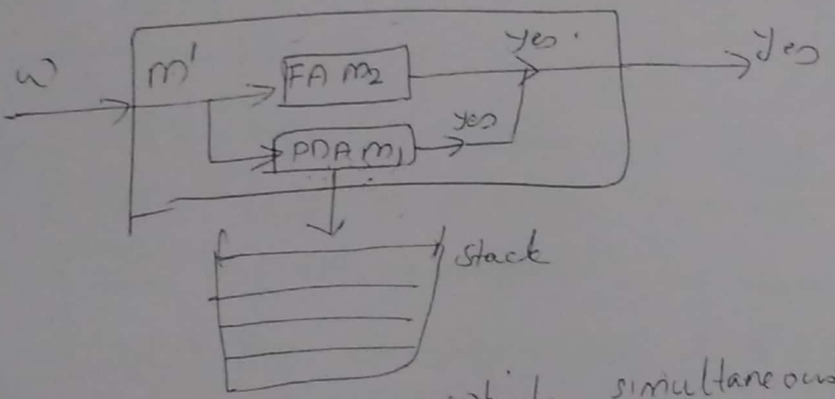
It can be shown by induction on ω, that ω accepted by M' iff $h(\omega)$ accepted by M. Hence M' accepts $h^{-1}(L)$.

VII If L is a CFL and R is a regular set, then $L \cap R$ is a CFL i.e. CFLs are closed under intersection with regular sets.

Proof: Let L be a CFL accepted by PDA $M_1 = (Q_m, \Sigma, \Gamma, \delta_m, q_0, z_0, F_m)$

& R be a regular set accepted by DFA $M_2 = (Q_A, \Sigma, \delta_A, p_0, F_A)$

We construct a PDA M' for $L \cap R$ by running M_1 & M_2 in parallel.



Here we construct a PDA which simultaneously simulates both M_1 & M_2. Formally

$$M' = (Q_A \times Q_m, \Sigma, \Gamma, \delta', [p_0, q_0], z_0, F_A \times F_m)$$

where δ' defined as $\delta'([p,q], a, X)$ contains

$([p', q'], \gamma)$ iff $\delta_A(p, a) = p'$ and

$\delta_m(q, a, X)$ contains (q', γ).

If $a = \varepsilon$, $p' = p$.

It can be shown by induction that for any input w

$([p_0, q_0], w, z_0) \vdash^* ([p, q], \varepsilon, \gamma)$ iff

$(q_0, w, z_0) \vdash^* (q, \varepsilon, \gamma)$ and $\delta_A(p_0, w) = p$. Hence $L(M')$ ...

~~Decision~~

Decision algorithms for CFLs

There are a no: of questions about CFLs we can answer. These include whether a given CFL is empty, finite or infinite and whether a given word is a CFL. There are certain questions which no CFL can answer. These include whether 2 CFGs are equivalent, whether a CFL is cofinite, whether the complement of a CFL is a CFL and whether a given CFG is ambiguous.

(1) There are algorithms to determine if a CFL is
a) empty b) finite or c) infinite

The theorem can be proved by pumping lemma. Let L be a CFL and n be the natural no obtained using pumping lemma.

Then (1) L is non-empty if and only if $z \in L$ and $|z| < n$

(2) L is infinite iff there exists $z \in L$ such that $n \leq |z| < 2n$ (proof similar to regular languages)

But these algorithms are inefficient.

A better algorithm to test whether a CFL is empty or not is given below.

Let $G = (V, T, P, S)$ be a CFG. $L(G)$ is non-empty iff the start symbol S generates some strings of terminals. Otherwise $L(G)$ is empty.

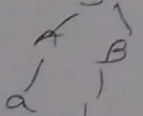
To test whether $L(G)$ is finite or infinite

Consider a CFG in CNF form.
A simple test for finiteness of a CNF grammar
with no useless symbols is to draw a
directed graph with a vertex for each variable
and an edge from A to B if there is
a production of form $A \to BC$ or $A \to CB$ for any
C. Then language generated is finite iff the
graph has no cycle. If there is atleast
one cycle in a directed graph generated
from CFL, then $L(G)$ is infinite. If $L(G)$
is finite there are no cycles. We define the
rank of a variable A to be the length of the
longest path in the graph beginning at A.
If A has rank r, then no terminal string
derived from A has length $> 2^r$

eg: Consider the grammar in CNF form

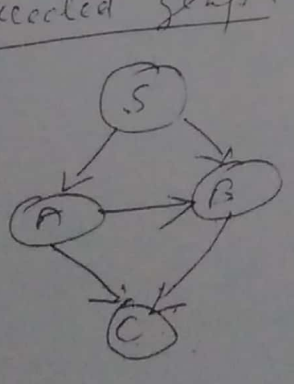
$S \to AB$
$A \to BC \mid a$
$B \to Cc \mid b$
$C \to a$

$S \to AB \to ab$



$L(G)$ is non-empty

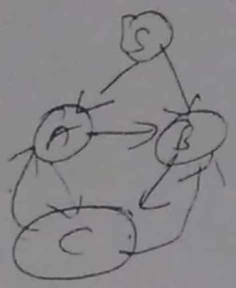
Directed graph



This graph has no cycles.
The ranks of S, A, B, & C are 3, 2, 1 & 0
The longest path from S is S, A, B, C.
This grammar derives no string of
length greater than $2^3 = 8$ and is
finite

Scanned by CamScanner

If we add $C \to BA$ also we get the directed graph as shown below



This graph has cycles
It is infinite

Membership Algorithm

To test whether a given word is in a given CFL, we use CYK algorithm (Cocke - Younger - Kasami) - CYK is of order $O(n^3)$.

Algorithm

begin //n → word length

(1) For $i = 1$ to n do

(2) $V_{i1} = \{A \mid A \to a$ is a production and ith symbol of x is $a\}$

(3) for $j = 2$ to n do

for $i = 1$ to $n - j + 1$ do

begin $V_{ij} = \phi$

for $k = 1$ to $j-1$ do

$V_{ij} = V_{ij} \cup \{A \mid A \to BC$ is a production, B is in V_{ik} & C is in $V_{i+k,\, j-k}\}$

end

end

eg Consider the CFG

$S \to AB \mid BC$

$A \to BA \mid a$

$B \to CC \mid b$

$C \to AB \mid a$

i/p string is baaba find whether the given string is a member or not using CYK

$i \to$

$j \downarrow$

j \ i	1	2	3	4	5
1	B	A, C	A, C	B	A, C
2	S, A	B	S, C	S, A	
3	ϕ	B	ϕ		
4	ϕ	S, A, C			
5	S, A, C				

Top row filled by steps 1 & 2 of algorithm

$V_{11} = B$ as $B \to b$

$V_{21} = A, C$ as $A \to a$ & $C \to a$

||rly V_{31} & V_{51}.

$V_{41} = B$ as $B \to b$

To compute V_{21}

$j = 2$ $\quad i = 1$ to 4.

$V_{12} = \phi$ $\quad V_{11} = B$. $\quad$ there are 2 prods

$V_{21} = A, C$. $\quad S \to BC$ & $A \to BA$

$k = 1$

So $V_{12} = \phi \cup \{S, A\} = \{S, A\}$

$V_{22} = \phi \cup B = B$

$V_{21} = A, C$ $\quad B \to CC$ is only prod in CFG

$V_{21} = A, C$

||rly compute all V_{ij}s.

Since S is a member of V_{15}, the string baaba is the language generated by the grammar.