Context free grammar (CFG):

Defn. A type 2 production is a production of the form $A \to \alpha$, where $A \in V_N$ and $\alpha \in (V_N \cup \Sigma)^*$. In other words, the LHS. has no left context or right context, e.g. $S \to Aa$, $A \to a$, $B \to abC$, $A \to A$ are type 2 produc

Defn. A grammar is called a type 2 grammar if it contains only type 2 productions. It is also called a context-free grammar (as A can be reduced by α in any context).

A language generated by context-free grammar is ca context-free language.

Derivation Trees. The derivations in a CFG can be repre using trees. Such trees are called derivation trees.

Defn. A derivation tree for a CFG $G = (V_N, \Sigma, P, S)$ is tree satisfying the following conditions:

(i) every vertex has a label which is a variable or terminal o

(ii) The root has label S.

(iii) The label of an internal vertex is a variable.

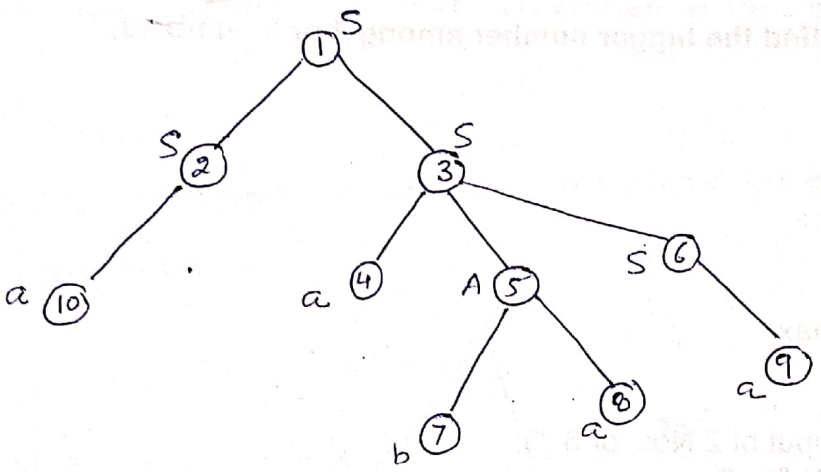
(iv) If the vertices $n_1, n_2, \cdots, n_k$ written with labels $X_1, X_2, \cdots$ are the sons of vertex n with label A, then $A \to X_1 X_2 \cdots$ is a production in P.

(v) A vertex n is a leaf if its label is $a \in \Sigma$ or Λ; n only son of its father if its label is Λ.

~~e.g. let $G = (\{S, A\}, \{a, b\}, P, S)$~~

~~S~~

Ex: Let $G = (\{S, A\}, \{a, b\}, P, S)$



$S \longrightarrow SS/a/aAS$, $A \longrightarrow ba$

Defn. The yield of a derivation tree is the concatanation of the labels of the leaves without repetition in the left-to-right-ordering.

e.g. the yield of the above derivation tree is aabaa.

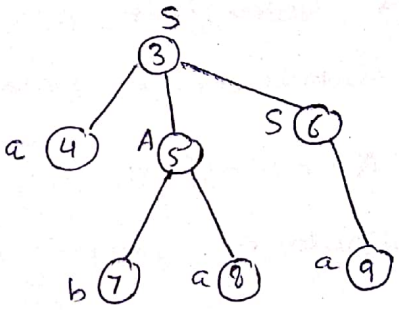
In the above tree,

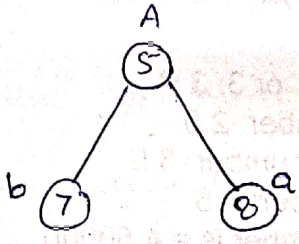
$$S \longrightarrow SS \longrightarrow aS \longrightarrow aaAS \longrightarrow aabaS \longrightarrow aabaa$$

Thus the yield of the derivation tree is a sentential form in G.

Defn. A subtree of a derivation tree T is a tree (i) whose root is some vertex v of T, (ii) whose vertices are the descendants of v together with their labels, and (iii) whose edges are those connecting the descendants of v.

e.g.





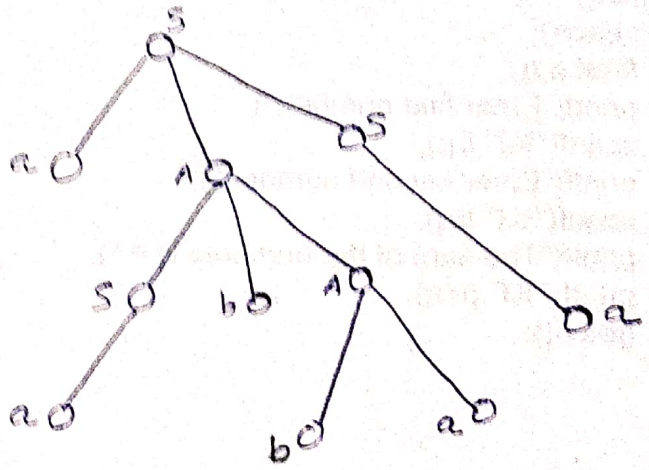
are the subtre
of the above tr

Note: A subtree look like a D-tree except that the label of the root may not be S. It is called an A-tree if the label of its root is A.

Ex: Consider a Grammar productions are $S \to aAS / a$, $A \to SbA / SS / ba$. Show that $S \Rightarrow^* aabbaa$ and construct a D-Tree whose yield is aabbaa.

Soln: $S \to aAS \to aSbAS \to aabAS \to aabbaS \to aabbaa$. (i)

(i) $S \Rightarrow^* aabbaa$.



Also we can write

$$S \to aAS \xrightarrow{S \to a} aAa \xrightarrow{A \to SbA} aSbAa \xrightarrow{A \to ba} aSbbaa \xrightarrow{S \to a} aabbaa \quad \text{(ii)}$$

Also, $S \to aAS \xrightarrow{A \to SbA} aSbAS \xrightarrow{S \to a} aSbAa \xrightarrow{S \to a} aabAa \xrightarrow{A \to ba} aabbaa$ (iii)

In derivation (i), whenever we replace a variable X using a production, there are no variables to the left of X, In (ii), there are no variables to the right of X. But in (iii), no such conditions are satisfied.

Defn. A derivation $A \Rightarrow^* w$ is called a 'leftmost' derivation if we apply a production only to the leftmost variable at every step.

Defn. A derivation $A \Rightarrow^* w$ is called a 'rightmost' derivation if we apply production to the rightmost variable at every step.

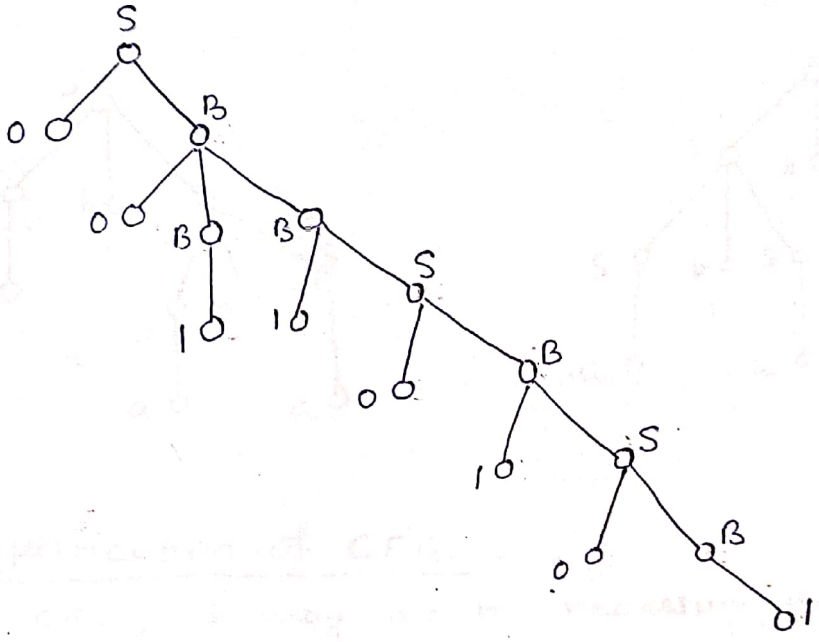
Ex. Let G be the grammar $S \to 0B / 1A$, $A \to 0 / 0S / 1AA$, $B \to 1 / 1S / 0BB$. For the string 00110101, find (a) the leftmost derivation, (b) the rightmost derivation, and (c) the D-tree.

Soln.

(a) $S \xrightarrow{B\to 0BB} 0B \to 00BB \xrightarrow{B\to 1} 001B \xrightarrow{B\to 1S} 0011S \xrightarrow{S\to 0B} 00110B \xrightarrow{B\to 1S} 00110S1 \xrightarrow{S\to 0B} 001101B1 \xrightarrow{B\to 1} 00110101$

(4)

(b) $S \xrightarrow{B\to 0BB} 0B \to 00BB \xrightarrow{B\to 1S} 00B1S \xrightarrow{S\to 0B} 00B10B \xrightarrow{B\to 1S} 00B101S \xrightarrow{S\to 0B} 00B1010B \xrightarrow{B\to 1} 00B10101 \xrightarrow{B\to 1} 00110101$



Ambiguity In CFG :

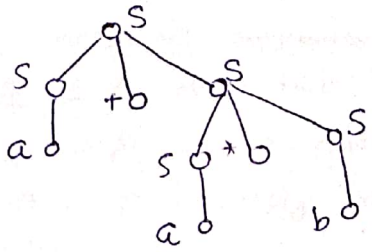
Defn. A terminal string $w \in L(G)$ is ambiguous if there exists two or more D-trees for w [or] two or more left most derivations of w).

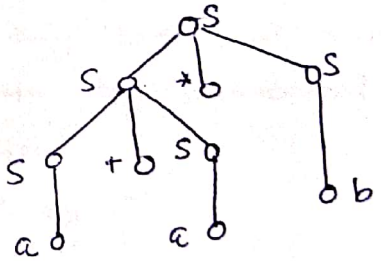
Ex. $G = (\{S\}, \{a, b, +, *\}, P, S)$, where P consists of-

$S \to S+S \mid S*S \mid a \mid b$. We have two D-trees for $a+a*b$

$S \to S+S \to a+S \to a+S*S \to a+a*S \to a+a*b$

$S \to S*S \to S+S*S \to a+S*S \to a+a*S \to a+a*b$.





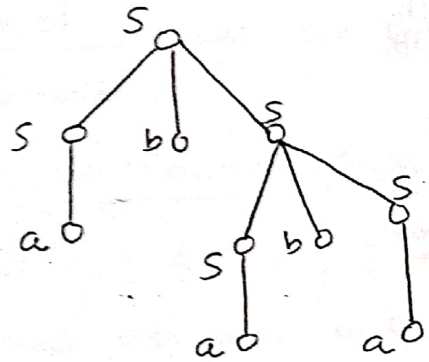
Defn. A CFG is ambiguous if $\exists$ some $w \in L(G)$ which is ambiguous.

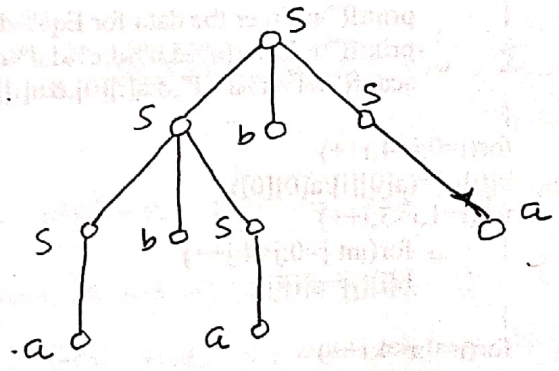
Ex If G is the grammar $S \to SbS / a$, show that G is ambiguous.

Soln: $S \xrightarrow{S\to SbS} SbS \xrightarrow{S\to a} abS \xrightarrow{S\to SbS} abSbS \xrightarrow{S\to a} ababS \xrightarrow{S\to a} ababa$

$S \xrightarrow{S\to SbS} SbSbS \xrightarrow{S\to a} abSbS \xrightarrow{S\to a} ababS \xrightarrow{S\to a} ababa$

$\therefore$ $w = ababa \in L(G)$ which is ambiguous. Thus G is ambiguous.





Simplification of CFG.

In a CFG, it may not be necessary to use all the symbols in $V_N \cup \Sigma$ or all the productions in P for deriving sentences. we may try to eliminate those symbols and productions in G, which are not useful for the derivation of sentences.

e.g. $G = (\{S, A, B, C, E\}, \{a, b, c\}, P, S)$, where

$P = \{S \to AB, A \to a, B \to b, B \to C, E \to c / \Lambda\}$.

It is easy to see that $L(G) = \{ab\}$. Let $G' = (\{S, A, B\}, \{a, b\}, P', S)$ where $P' = \{S \to AB, A \to a, B \to b\}$, $L(G) = L(G')$. we have eliminated the symbols C, E & C and the productions $B \to C$, $E \to c/\Lambda$. we note the following pts.

(i) C does not ~~appear~~ derive any terminal strings
(ii) E & C does not appear in any sentential form
(iii) $E \to \Lambda$ is a null production
(iv) $B \to C$ simply replaces B by C.

We give the construction to eliminate (i) variables not deriving terminal strings, (ii) symbols not appearing in any sentential form, (iii) null productions, and (iv) productions of the form $A \to B$.

Construction of Reduced Grammars

Theorem. If G be a CFG s.t. $L(G) \neq \phi$, we can find an equivalent grammar G' s.t. each variable in G' derives some terminal string.

Proof. Let $G = (V_N, \Sigma, P, S)$. Define $G' = (V_N', \Sigma, P', S)$ as follows:

(a) **construction of V_N'** – we define $W_i \subseteq V_N$ by recursion:

$W_1 = \{ A \in V_N \mid \exists$ a production $A \to w, w \in \Sigma^* \}$. (If $W_1 = \phi$, some variable will remain after the application of any production, and so $L(G) = \phi$).

$W_{i+1} = W_i \cup \{ A \in V_N \mid \exists$ some production $A \to \alpha, \alpha \in (\Sigma \cup W_i)^* \}$

By defn. of W_i, $W_i \subseteq W_{i+1}$ $\forall i$. As V_N is finite, $W_K = W_{K+1}$ for some $K \leq |V_N|$. Therefore, $W_K = W_{K+j}$ for $j \geq 1$. We define $V_N' = W_K$.

(b) **construction of P':** $P' = \{ A \to \alpha \mid A, \alpha \in (V_N' \cup \Sigma)^*$ where $\}$

we can define $G' = (V_N', \Sigma, P', S)$. $S \in V_N'$. (If $S \notin V_N'$, $L(G') = \phi$)

Now we are going to prove that every variable in V_N' derives some terminal string. So, we prove:

(i) If $A \in V_N'$, then $A \overset{*}{\underset{G'}{\Rightarrow}} w$ for some $w \in \Sigma^*$; conversely, if $A \overset{*}{\underset{G'}{\Rightarrow}} w$, then $A \in V_N'$.

(ii) $L(G') = L(G)$.

To prove (i) we note that $W_K = W_1 \cup W_2 \cup \cdots \cup W_K$. We prove by induction on i that for $i = 1, 2, \cdots, k$, $A \in W_i$ implies $A \overset{*}{\Rightarrow} w$ for some $w \in \Sigma^*$

If $A \in W_1$, then $A \underset{G}{\Rightarrow} w$. So $A \to w$ is in P'.
i.e. $A \underset{G'}{\overset{*}{\Rightarrow}} w$. Thus there is a basis for induction. Let us assume the result for i. Let $A \in W_{i+1}$. Then either $A \in W_i$, in which case, $A \underset{G'}{\overset{*}{\Rightarrow}} w$ for some $w \in \Sigma^*$ by induction hypothesis, or $\exists$ a production $A \to \alpha$, $\alpha \in (\Sigma \cup W_i)^*$. By defn. of P', $A \to \alpha$ is in P'. We can write $\alpha = X_1 X_2 \cdots X_m$ where $X_j \in \Sigma \cup W_i$. If $X_j \in W_i$ by induction hypothesis, $X_j \underset{G'}{\overset{*}{\Rightarrow}} w_j$ for some $w_j \in \Sigma^*$. So, $A \underset{G'}{\overset{*}{\Rightarrow}} w_1 w_2 \cdots w_m \in \Sigma^*$ (when X_j is terminal, $w_j = X_j$). By induction result is true for $i = 1, 2, \cdots, k$.

The converse part can be proved in a similar way by induction on the no. of steps in the derivation $A \underset{G}{\overset{*}{\Rightarrow}} w$.

We see immediately that $L(G') \subseteq L(G)$, as $V_N' \subseteq V_N$ and $P' \subseteq P$. To prove $L(G) \subseteq L(G')$, we need an auxillary result $A \underset{G'}{\overset{*}{\Rightarrow}} w$ if $A \underset{G}{\overset{*}{\Rightarrow}} w$ for some $w \in \Sigma^*$.

We prove this result by induction on the no. of steps in the derivation $A \underset{G}{\overset{*}{\Rightarrow}} w$.

If $A \underset{G}{\Rightarrow} w$, then $A \to w$ is in P and $A \in W_1 \subseteq V_N'$. As $A \in V_N'$ & $w \in \Sigma^*$, $A \to w$ is in P'. So $A \underset{G'}{\Rightarrow} w$, and there is a basis for induction. Assume the result for derivations in at most k steps. Let $A \underset{G}{\overset{k+1}{\Rightarrow}} w$. We can split this as $A \underset{G}{\Rightarrow} X_1 X_2 \cdots X_m \underset{G}{\overset{*}{\Rightarrow}} w_1 w_2 \cdots w_m$ s.t. $X_j \underset{G}{\overset{*}{\Rightarrow}} w_j$. If $X_j \in \Sigma$ then $w_j = X_j$.

If $X_j \in V_N$ then by (i), $X_j \in V_N'$. As $X_j \underset{G}{\overset{*}{\Rightarrow}} w_j$ in at most k steps, $X_j \underset{G'}{\overset{*}{\Rightarrow}} w_j$. Also $X_1, X_2, \cdots, X_m \in (\Sigma \cup V_N')^*$ implies that $A \to X_1 X_2 \cdots X_m$ is in P'. Thus $A \underset{G'}{\Rightarrow} X_1 X_2 \cdots X_m \underset{G'}{\overset{*}{\Rightarrow}} w_1 w_2 \cdots w_m$. Hence result is true for all derivations. In particular, $S \underset{G}{\overset{*}{\Rightarrow}} w$ implies $S \underset{G'}{\overset{*}{\Rightarrow}} w$.

$\therefore L(G) \subseteq L(G')$ and (ii) is completely proved.

Theorem For every CFG $G = (V_N, \Sigma, P, S)$, we can construct an equivalent grammar $G' = (V_N', \Sigma', P', S)$ s.t. every symbol in $V_N' \cup \Sigma'$ appears in some sentential form (i.e. $\forall X \in (V_N' \cup \Sigma')$, $\exists \alpha$ s.t. $S \overset{*}{\underset{G'}{\Rightarrow}} \alpha$ and X is a symbol in the string α).

We will not give complete proof but only construction of W_i, V_N', Σ' & P'.

construct $G' = (V_N', \Sigma', P', S)$ as follows:

(a) construction of W_i for $i \geq 1$;

(i) $W_1 = \{S\}$.

(ii) $W_{i+1} = W_i \cup \{ X \in V_N \cup \Sigma \mid \exists$ a production $A \to \alpha$ with $A \in W_i$ and α containing the symbol $X \}$.

$\therefore W_i \subseteq (V_N \cup \Sigma)$ & $W_i \subseteq W_{i+1}$. As $(V_N \cup \Sigma)$ is finite, $W_K = W_{K+1}$ for some K. $\therefore W_K = W_{K+j}$ $\forall j \geq 0$.

Let $X \in (V_N' \cup \Sigma') = W_K$. prove by induction on i. i.e. $X \in W_1 = \{S\}$ then it appears in some sentential form. Let result is true for i. Let $X \in W_{i+1}$ $\Rightarrow$ either $X \in W_i$ for which result is true or $\exists A \to \alpha$ with $A \in W_i$ $\therefore$ A appears in sent. form, hence result.

(b) construction of V_N', Σ' and P';

Define $V_N' = V_N \cap W_K$, $\Sigma' = \Sigma \cap W_K$

$P' = \{ A \to \alpha : A \in W_K \}$.

Defn. A grammar $G = (V_N, \Sigma, P, S)$ is said to be reduced or non-redundant if every symbol in $(V_N \cup \Sigma)$ appears in the course of the derivation of some terminal string, i.e. $\forall X \in (V_N \cup \Sigma)$, $\exists$ a derivation $S \overset{*}{\underset{G}{\Rightarrow}} \alpha X \beta \overset{*}{\Rightarrow} w \in L(G)$.

Theorem For every CFG G, $\exists$ a reduced grammar G' which is equivalent to G.

(For proof, follow the two previous theorems in order i.e. thm 1 and then thm 2).

Ex. Let $G = (V_N, \Sigma, P, S)$ be given by the productions $S \to AB$, $A \to a$, $B \to b$, $B \to C$, $E \to c$. Find G' s.t. every variable in G' derives some terminal string.

Soln. (a) construction of V_N':

$W_1 = \{A, B, E\}$.

$W_2 = W_1 \cup \{A_1 \in V_N : A_1 \to \alpha \text{ for some } \alpha \in (\Sigma \cup \{A, B, E\})^*\}$

$= W_1 \cup \{S\} = \{A, B, E, S\}$.

$W_3 = W_2 \cup \{A_1 \in V_N : A_1 \to \alpha \text{ for some } \alpha \in (\Sigma \cup \{S, A, B, E\})^*\}$

$= W_2 \cup \phi = W_2$.

$\therefore V_N' = \{S, A, B, E\}$.

(b) construction of P': $P' = \{A_1 \to \alpha : \alpha \in (V_N' \cup \Sigma)^*\}$

$= \{S \to AB, A \to a, B \to b, E \to c\}$

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

Ex. consider $G = (\{S, A, B, E\}, \{a, b, c\}, P, S)$, where P consists of $S \to AB$, $A \to a$, $B \to b$, $E \to c$. construct an equivalent grammar $G' = (V_N', \Sigma', P', S)$ s.t. every symbol in $V_N' \cup \Sigma'$ appears in some sentential form.

Soln. $W_1 = \{S\}$, $W_2 = W_1 \cup \{X \in (V_N \cup \Sigma) : \exists\ A_1 \to \alpha$ with $A_1 \in W_1$ & α containing $X\}$

$= \{S\} \cup \{A, B\} = \{S, A, B\}$

$W_3 = \{S, A, B\} \cup \{a, b\}$, $W_4 = W_3$

$\therefore V_N' = \{S, A, B\}$, $\Sigma' = \{a, b\}$, $P' = \{S \to AB, A \to a, B \to b\}$.

Ex. Find a reduced grammar equivalent to the grammar G whose productions are $S \to AB | CA$, $B \to BC | AB$, $A \to a$, $C \to aB | b$.

Soln. **Step1.** $W_1 = \{A, C\}$ as $A \to a$ & $C \to b$ are productions with a terminal string on RHS.

$W_2 = \{A, C\} \cup \{ A_1 : A_1 \to \alpha \text{ with } \alpha \in (\Sigma \cup \{A, C\})^* \}$

$= \{A, C\} \cup \{S\} = \{S, A, C\}$

$W_3 = \{S, A, C\} \cup \{ A_1 : A_1 \to \alpha \text{ with } \alpha \in (\Sigma \cup \{S, A, C\})^* \}$

$= \{S, A, C\} = W_2$

$V_N' = W_2 = \{S, A, C\}$

$P' = \{ A_1 \to \alpha : A_1, \alpha \in (V_N' \cup \Sigma)^* \}$

$= \{ S \to CA, A \to a, C \to b \}$

Thus $G_1 = (\{S, A, C\}, \{a, b\}, \{S \to CA, A \to a, C \to b\}, S)$

Step2- we have to apply 2nd thm. Thus,

$W_1 = \{S\}$

as we have $S \to CA$ & $S \in W_1$, $W_2 = \{S\} \cup \{A, C\} = \{S, A, C\}$

as $A \to a$, $C \to b$ are productions with $A, C \in W_2$, $W_3 = \{S, A, C, a, b\}$

as $W_3 = V_N' \cup \Sigma$, $P'' = \{A_1 \to a : A_1 \in W_3\} = P'$

$\therefore$ $G' = (\{S, A, C\}, \{a, b\}, \{S \to CA, A \to a, C \to b\}, S)$ is the reduced grammar.

Elimination of Null productions.

A CFG may have productions of the form $A \to \Lambda$. This is just used to erase A. So a production of the form $A \to \Lambda$, where A is a variable, is called a null production.

e.g. $S \to aS \mid aA \mid \Lambda$, $A \to \Lambda$ has two null productions $S \to \Lambda$ and $A \to \Lambda$. We can delete $A \to \Lambda$ provided we erase A whenever it occurs in a derivation of a terminal string. So, we can replace $S \to aA$ by $S \to a$. If G_1 denotes the grammar with $S \to aS \mid a \mid \Lambda$ then $L(G_1) = L(G) = \{a^n \mid n \geq 0\}$. Thus it is possible to eliminate the null production $A \to \Lambda$.

Defn. A variable in a CFG is nullable if $A \overset{*}{\Rightarrow} \Lambda$.

Theorem: If $G = (V_N, \Sigma, P, S)$ is a CFG, then we can find a CFG G_1 having no null productions s.t. $L(G_1) = L(G) - \{\Lambda\}$.

Proof: we construct $G_1 = (V_N, \Sigma, P', S)$ as follows:

Step1: construction of the set of nullable variables:

we find these variables recursively

(i) $W_1 = \{A \in V_N : A \to \Lambda \text{ in } P\}$.

(ii) $W_{i+1} = W_i \cup \{A' \in V_N : \exists\ A' \to \alpha \text{ with } \alpha \in W_i^*\}$.

$\therefore W_i \subseteq W_{i+1}\ \forall i$. As V_N is finite, $W_{K+1} = W_K$ for some $K \le |V_N|$. So $W_{K+j} = W_K\ \forall j$. Let $W = W_K$. W is the set of nullable variables.

Step2: (i) construction of P': Any production whose RHS does not have any nullable variable is included in P'.

(ii) If $A \to X_1 X_2 \cdots X_K$ is in P, the productions of the form $A \to \alpha_1 \alpha_2 \cdots \alpha_K$ are included in P', where $\alpha_i = X_i$ if $X_i \notin W$, $\alpha_i = X_i$ or Λ if $X_i \in W$ and $\alpha_1 \alpha_2 \cdots \alpha_K \neq \Lambda$. Actually (ii) gives several productions in P'. The productions are obtained either by not erasing any nullable variable on the RHS of $A \to X_1 X_2 \cdots X_K$ or by erasing some or all nullable variables provided some symbol appears on the RHS after erasing.

Now we have to prove that $L(G_1) = L(G) - \{\Lambda\}$. we prove an auxillary result given by the following relation:

for all $A \in V_N$ and $\omega \in \Sigma^*$,

$$A \overset{*}{\underset{G_1}{\Rightarrow}} \omega \text{ iff } A \overset{*}{\underset{G}{\Rightarrow}} \omega \text{ and } \omega \neq \Lambda. \qquad \text{(iii)}$$

we prove the 'if' part first. Let $A \overset{*}{\underset{G}{\Rightarrow}} \omega$ and $\omega \neq \Lambda$. we prove $A \overset{*}{\underset{G_1}{\Rightarrow}} \omega$ by induction on the no. of steps in the derivation $A \overset{*}{\underset{G}{\Rightarrow}} \omega$. If $A \underset{G}{\Rightarrow} \omega$ & $\omega \neq \Lambda$, $A \to \omega$ is in P', so $A \underset{G_1}{\Rightarrow} \omega$. Assume the result is true for derivations in at most i steps. Let $A \overset{i+1}{\underset{G}{\Rightarrow}} \omega$ & $\omega \neq \Lambda$.

we can split the derivation as

$A \underset{G}{\Rightarrow} X_1 X_2 \cdots X_k \overset{i}{\underset{G}{\Rightarrow}} w_1 w_2 \cdots w_k$, where $w = w_1 w_2 \cdots w_k$.

and $A_j \overset{*}{\underset{G}{\Rightarrow}} w_j$. As $w \neq \Lambda$, not all w_j's are ~~empty~~ Λ.

If $w_j \neq \Lambda$, then by induction hypothesis $X_j \overset{i}{\underset{G}{\Rightarrow}} w_j$. If $w_j = \Lambda$, then $X_j \in w$. So using the production $A \to A_1 A_2 \cdots A_k$ in P, we construct $A \to \alpha_1 \alpha_2 \cdots \alpha_k$ in P', where $\alpha_j = X_j$ if $w_j \neq \Lambda$ and $\alpha_j = \Lambda$ if $w_j = \Lambda$ (i.e. $X_j \in w$). Therefore,

$$A \underset{G_1}{\Rightarrow} \alpha_1 \alpha_2 \cdots \alpha_k \overset{*}{\underset{G_1}{\Rightarrow}} w_1 \alpha_2 \cdots \alpha_k \overset{*}{\underset{G_1}{\Rightarrow}} w_1 w_2 \cdots w_k = w$$

By PMI, 'if' part is proved.

we prove the 'only if' part by PMI on the no. of steps in the derivation of $A \overset{*}{\underset{G_1}{\Rightarrow}} w$. If $A \overset{}{\underset{G_1}{\Rightarrow}} w$ then $A \to w$ is in P'.

$\therefore A \to w$ is obtained from some production $A \underset{G}{\Rightarrow} X_1 X_2 \cdots X_k \overset{*}{\underset{G}{\Rightarrow}} w$.

So, there is a basis for PMI. Assume the result for derivation in at most j steps. Let $A \overset{j+1}{\underset{G}{\Rightarrow}} w$. This can be split as $A \overset{}{\underset{G_1}{\Rightarrow}} X_1 X_2 \cdots X_k \overset{j}{\underset{G_1}{\Rightarrow}} w_1 w_2 \cdots w_k$, where $X_i \overset{*}{\underset{G}{\Rightarrow}} w_i$, $A \Rightarrow X_1 X_2 \cdots X_k$ in P' is obtained from some production $A \to \alpha$ in P by erasing some (or none of the) nullable variables in α. So $A \underset{G}{\Rightarrow} \alpha \overset{*}{\underset{G}{\Rightarrow}} X_1 X_2 \cdots X_k$. If $X_i \in \Sigma$ then $X_i \overset{0}{\underset{G}{\Rightarrow}} X_i = w_i$. If $X_i \in V_N$ then by induction hypothesis, $X_i \overset{*}{\underset{G}{\Rightarrow}} w_i$, So $A \overset{*}{\underset{G}{\Rightarrow}} X_1 X_2 \cdots X_k \overset{*}{\underset{G}{\Rightarrow}} w_1 w_2 \cdots w_k$.

Hence by PMI, result follows.

By applying this result (i.e. (iii)), we have $w \in L(G_1)$ iff $w \in L(G)$ and $w \neq \Lambda$. $\Rightarrow$ Hence, $L(G_1) = L(G) - \{\Lambda\}$.

Ex. Consider the grammar G whose productions are $S \to aS \mid AB$, $A \to \Lambda$, $B \to \Lambda$, $D \to b$. Construct a grammar G_1 without null productions generating $L(G) - \Lambda$.

Soln. Step1 :

$$W_1 = \{ A_i \in V_N : A_i \to \Lambda \text{ is in } G \}$$
$$= \{ A, B \}$$

$$W_2 = W_1 \cup \{ A_i \in V_N : A_i \to \alpha \text{ with } \alpha \in W_1^* \}$$
$$= \{A, B\} \cup \{S\} = \{ S, A, B \}$$

$$W_3 = W_2 \cup \phi = W_2.$$

$$\therefore W = \{ S, A, B \}.$$

Step2 : ~~$D \to b$~~, ~~$S \to aS$~~

(i) $D \to b$ is included in P'.

(ii) $S \to aS$ gives rise to $S \to aS$ and $S \to a$

(iii) $S \to AB$ gives rise to $S \to AB$, $S \to A$ and $S \to B$.

(we can not erase both A, B in $S \to AB$ as we will get $S \to \Lambda$).

Hence $G_1 = (\{S, A, B\}, \{a, b\}, P', S)$

Elimination of Unit Productions.

consider, e.g. G as $S \to A$, $A \to B$, $B \to C$, $C \to a$. Then $L(G) = \{a\}$.

The productions $S \to A$, $A \to B$, $B \to C$ are useful just to replace S by C.

To get terminal string, we need $C \to a$. If G_1 is $S \to a$, then $L(G_1) = L(G)$.

Defn. A unit production in CFG G is a production of the form $A \to B$, where $A, B \in V_N$.

Theorem. If G is a CFG, we can find a CFG G_1 which has no null productions or unit productions s.t. $L(G_1) = L(G)$.

Pf. For null productions, we have already done. i.e. we find G' s.t. $L(G') = L(G)$

Let $A \in V_N$.

Step1. Construction of the set of variables from A :

Define $W_i(A)$ as follows :

$W_0(A) = \{A\}$

$W_{i+1}(A) = W_i(A) \cup \{B \in V_N : C \to B \text{ is in } P \text{ with } C \in W_i(A)\}$

By defn. $W_i(A) \subseteq W_{i+1}(A)$. As V_N is finite, $W_{K+1}(A) = W_K(A)$ for some $K \leq |V_N|$. So $W_{K+j}(A) = W_i(A) \; \forall \; j \geq 0$.

Let $W(A) = W_K(A)$. Then $W(A)$ is the set of variables derivable from A.

Step2. construction of A- productions in G_1:

The A- productions in G_1 are either (i) the nonunit production in G' or (ii) $A \to \alpha$ whenever $B \to \alpha$ is in G with $B \in W(A)$ and $\alpha \notin V_N$.

Define $G_1 = (V_N, \Sigma, P_1, S)$, where P_1 is constructed using step 2 $\forall \; A \in V_N$.

(we have given only construction). Proof of $L(G_1) = L(G)$ can be seen from the book.

Corollary. If G is a CFG, we can construct an equivalent grammar G' which is reduced and has no null productions or unit productions.

pf. eliminate null productions to get G_1. Then eliminate unit productions in G_1 to get G_2. Construct a reduced grammar G' equivalent to G_2. G' is the required grammar.

(do not change the order of construction).

Normal Forms For CFG. In CFG, the RHS of a production can be any string of variables and terminals. When the productions in G satisfy certain restrictions, then G is said to be in a 'normal form'. Among several normal forms, main normal forms are 'chomsky normal form' (CNF) and 'Greibach normal form' (GNF).

Defn. (CNF): A CFG is in CNF if every production is of the form $A \to a$ or $A \to BC$, and $S \to \Lambda$ is in G if $\Lambda \in L(G)$, where $\Lambda \in L(G)$, we assume that S does not appear on the RHS. of any production.

Ex. Let G be $S \to AB$, $A \to a$, $B \to C/b$, $C \to D$, $D \to E$ and $E \to a$. Eliminate unit productions.

Soln. **Step1.** $W_0(S) = \{S\}$, $W_1(S) = W_0(S) \cup \phi$

$\therefore W(S) = (S)$. Similarly $W(A) = A$, $W(E) = E$.

$W_0(B) = \{B\}$, $W_1(B) = \{B\} \cup \{C\} = \{B, C\}$

$W_2(B) = \{B, C\} \cup \{D\}$, $W_3(B) = \{B, C, D, E\} = W_4(B)$

$\therefore W(B) = \{B, C, D, E\}$.

$W_0(C) = \{C\}$, $W_1(C) = \{C, D\}$, $W_2(E) = \{C, D, E\} = W_3(C)$

$\therefore W(C) = \{C, D, E\}$

$W_0(D) = \{D\}$, $W_1(D) = \{D, E\} = W_2(D)$

$\therefore W(D) = \{D, E\}$

Step2. The productions in G_1 are

$S \to AB$, $A \to a$, $E \to a$

$B \to b/a$, $C \to a$, $D \to a$.