Simplification of CFG

Removal of Useless Symbols

-- Sakshi Surve

Simplification of CFG :

- Various languages can efficiently be represented by a context-free grammar.

- All the grammar are not always optimized that means the grammar may consist of some extra symbols(non-terminal).

- Having extra symbols, unnecessary increase the length of grammar.

- Simplification of grammar means reduction of grammar by removing these extra symbols.

- In a CFG, may it happen that all the production rules are not needed for derivation of strings

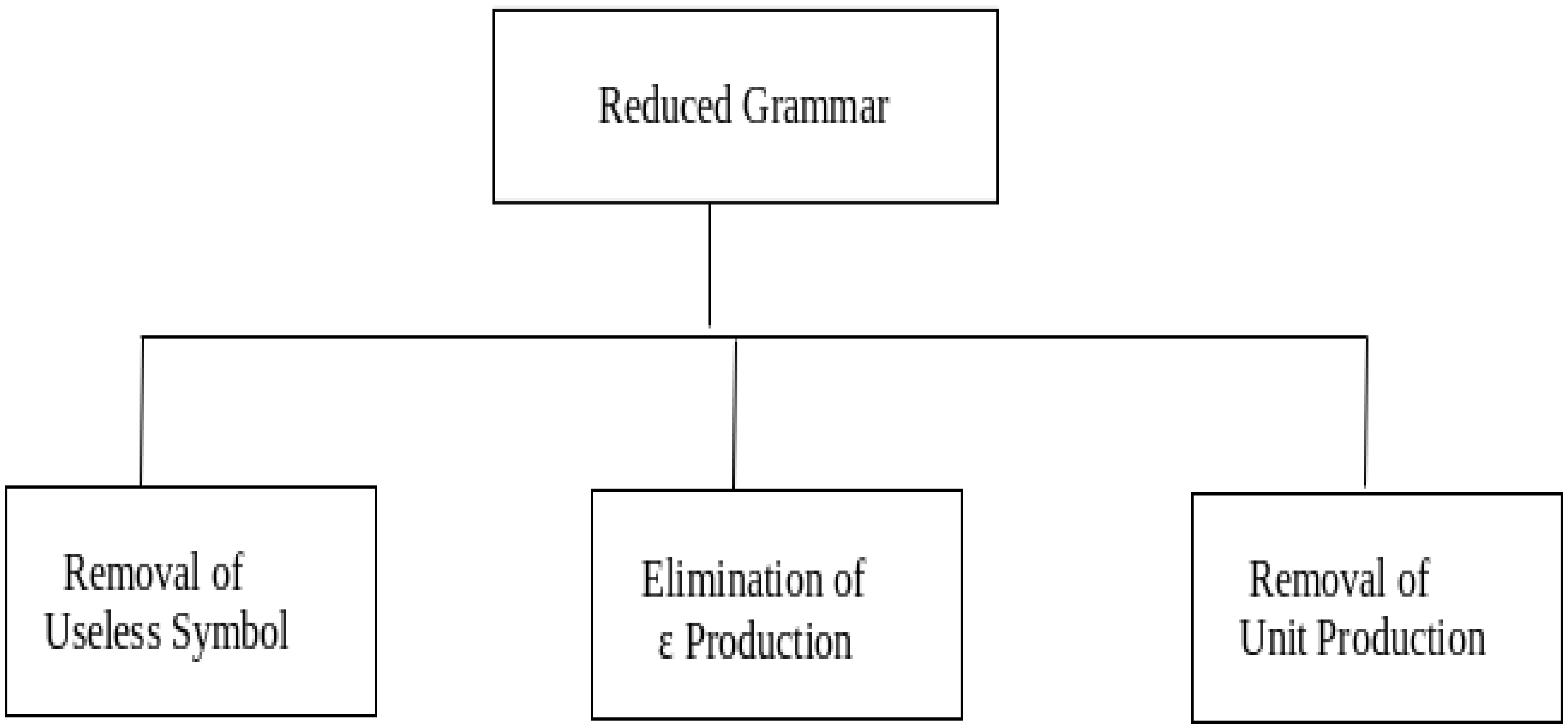
- Elimination of such strings is called "**Simplification of CFG**"

- **By Simplifying, we remove all the unnecessary, redundant productions while keeping the transformed grammar equivalent to the original one.**

- Simplified form can remove ambiguity and improve G

- **The properties of reduced grammar are given below:**
  - Each variable (i.e. non-terminal) and each terminal of G appears in the derivation of some word in L.
    - **Useless Symbols**
  - There should not be any production as $X \rightarrow Y$ where X and Y are non-terminal.
    - **Unit Productions**
  - If ε is not in the language L then the production $X \rightarrow \varepsilon$ need not be there .
    - **Null Productions**

Three ways to Simplify the Grammar :



1. Removal of Useless Symbols :

- **Useful Symbol (NT) :**
  - Appears on the LHS of the production rule
  - It generates some terminal symbol
    - **P =>* a**
- **Useless Symbol :**
  - It does not appear on the right-hand side of the production rule (It is not reachable from the start symbol)
  - It is not live (It doesn't derive a string)

- **A production rule is Useless if it involves any Useless Symbols**

Steps For reduction of a given grammar G:

1. Identify **Non-generating symbols** in the given CFG and eliminate those productions which contains non-generating symbols….Symbols not deriving any string

2. Identify **Non-reachable symbols** and eliminate those productions which contain the non-reachable symbols

Useless Symbols

- Let a CFG G. A symbol X ε (V U ∑) is useful if there is a derivation

$$S \overset{*}{\underset{G}{\Rightarrow}} UxV \overset{*}{\underset{G}{\Rightarrow}} w$$

Where U and V ε (V U ∑) and w ε ∑*. A symbol that is not useful is useless

- A terminal is useful if it occurs in a string of the language of G.
- A variable is useful if it occurs in a derivation that begins from S and generates a terminal string

For a variable to be useful two conditions must be satisfied.

1. The variable must occur in a sentential form of the grammar
2. There must be a derivation of a terminal string from the variable.

- A variable that occurs in a sentential form is said to be reachable from S.
- A two part procedure is presented to eliminate useless symbols.

- **Example 1: Remove the useless symbol from the given context free grammar:**

S -> aB / bX
A -> Bad / bSX / a
B -> aSB / bBX
X -> SBD / aBx / ad

Solution:

- A and X directly derive string of terminals a and ad, hence they are useful.

Since X is a useful symbol so S is also a useful symbol as S -> bX.

But B does not derive any string w … so clearly B is a non- generating symbol.

So eliminating those productions with B in them we get

S -> bX
A -> bSX / a
X -> ad

In the reduced grammar A is a non-reachable symbol so we remove it and the final grammar after elimination of the useless symbols is

S -> bX
X -> ad

- The new grammar generates all and only strings generated by the original grammar. Hence it is equivalent to the original grammar.

- The equivalent grammar G’ can be represented as :
- G’ = (S, V’, P’, T)
- Where
  - S = S
  - V ’ = { S, X }
  - T = { a, b }
  - P ’ =

  {   S -> bX
         X -> ad

  }

- **Example 2 : Find the equivalent useful grammar from the given grammar**

  A -> xyz / Xyzz
  X -> Xz / xYz
  Y -> yYy / Xz
  Z -> Zy / z

  Solution :

- A and Z is a useful symbol as it can be derived to a string of terminal symbol (Z -> z and A -> xyz).
- X and Y are not useful.
- So all the production with X and Y in them should be removed to eliminate non-generating symbols.
- The grammar then becomes

  A -> xyz
  Z -> Zy / z

- Since A is the starting symbol this implies Z is the non-reachable symbol.
- So we remove it to get a grammar free of useless symbols:

  A -> xyz

Example 3 : S -> AB/a

A -> BC/b

B -> aB/C

C -> aC/B

Solution:

Symbol B and C are useless symbol, remove them (whole production in which they are present)

So, Useful Symbols: {a, b, S, A} And any combination of useful symbols will also make LHS a useful symbol.

S -> a

A -> b

But cause A is not reachable so we will remove A -> b as well, the final production is :

S -> a

Example 4: S -> AB/AC

A -> aAb/bAa/a

B -> bbA/aaB/AB

C -> abCA/aDb

D -> bD/aC

Solution:

First find out useful Symbols: {a, b, A, B, S}

And useless symbols are: {C, D}

So remove them and write the whole grammar again:

S -> AB

A -> aAb/bAa/a

B -> bbA/aaB/AB

Example 5:

S -> AB | B |a
A -> aA
B -> b

Solution :

S -> B | a
B -> b

Example 6:

S-> abS | abA | abB

A-> cd

B-> aB

C-> dc

Solution :

S-> abS | abA

A-> cd

Example 7:

S-> aAa | aBC

A-> aS | bD

B-> aBa | b

C-> abb | DD

D-> aDa

Solution :

S-> aBC | aAa

A-> aS

B-> aBa | b

C-> abb

Example 8:

T-> aaB | abA | aaT

A-> aA

B-> ab | b

C-> ad

Solution :

T-> aaB | aaT

B-> ab | b

Example 9:

S-> AC | BS | B

A-> aA | aF

B-> CF | b

C-> cC | D

D-> aD | BD | C

E-> aA | BSA

F-> bB | b

S-> BS | B

A-> aF

B-> b

F-> bB | b

Solution :

S-> BS | B

B-> b

Example 10 :

S-> EA

A-> abA | ab

C-> EC | Ab

E -> bC

G-> EbE | CE | ba

Solution :

S-> EA

A-> abA | ab

C-> EC | Ab

E -> bC

Example 11 :

S-> aS | A | C

A-> a

B-> aa

C-> aCb

After removing the non generating non terminals, the grammar becomes :

S-> aS | A

A-> a

B-> aa

Solution :

After removing the non reachable non terminals, the grammar becomes :

S-> aS |A

A-> a

Example 12 :

S-> aA | bB

A-> aA | a

B-> bB

D -> ab | Ea

E-> aE | d

Solution :

S-> aA

A-> aA | a

Example 13 :

S-> AB | CA

A-> a

B-> BC | AB

C -> aB | b

Solution :

S-> CA

A-> a

C->a

Example 14 :

S-> aA | a | Bb | CC

A-> aB

B-> a | Aa

C -> cCD

D-> ddd

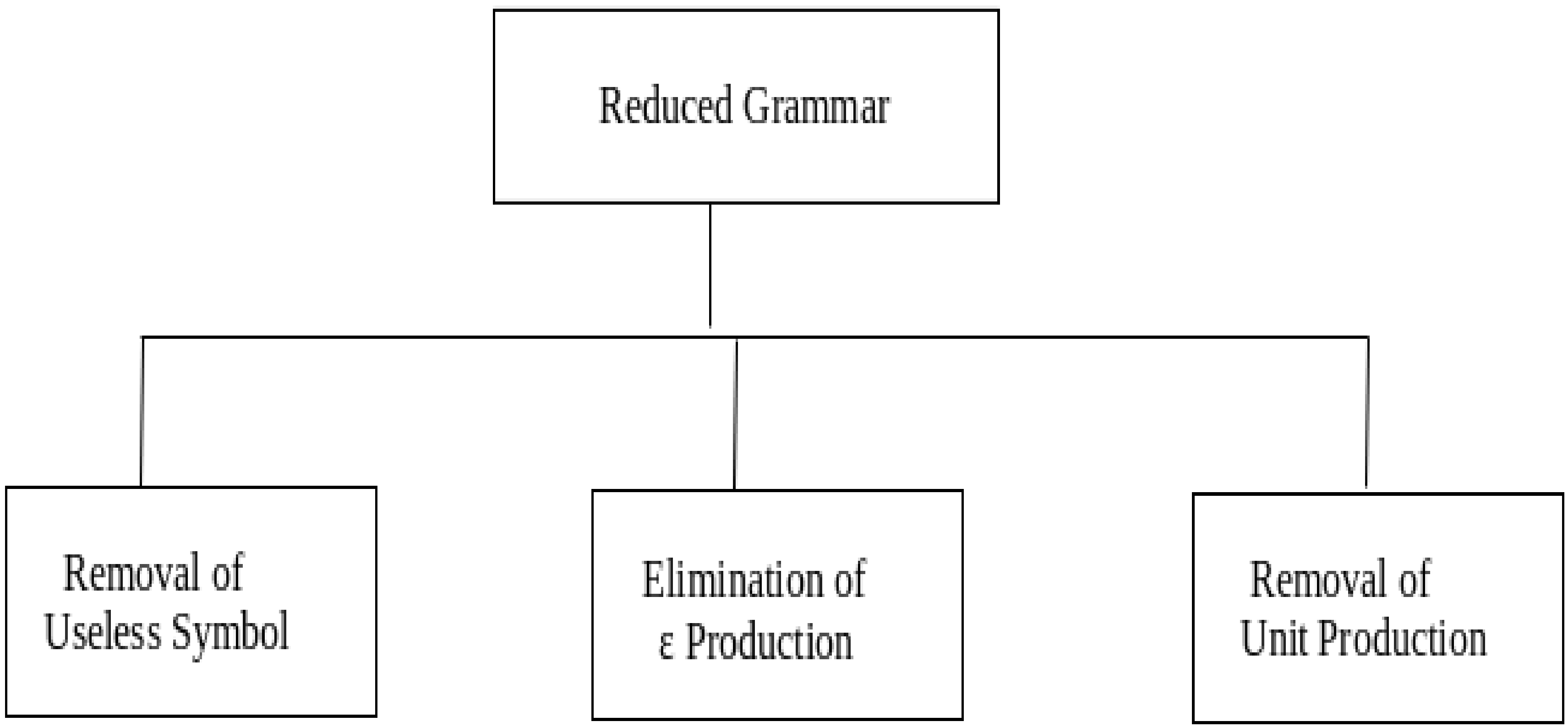
Solution :

S-> aA | a | Bb

A-> aB

B-> a | Aa

Three ways to Simplify the Grammar :



Removal of Unit Productions :

- The unit productions are the productions in which one non-terminal gives another non-terminal. A->B
- Use the following steps to remove unit production:
- **Step 1:** To remove **X → Y**, add production **X → a** to the grammar rule whenever **Y → a** occurs in the grammar. [x ∈ Terminal, x can be Null]
- **Step 2:** Now delete X → Y from the grammar.
- **Step 3:** Repeat step 1 and step 2 until all unit productions are removed.

Example 1 :Remove Unit Productions

$S \rightarrow 0A \mid 1B \mid C$

$A \rightarrow 0S \mid 00$

$B \rightarrow 1 \mid A$

$C \rightarrow 01$

Solution:

$S \rightarrow C$ is a unit production.

But while removing $S \rightarrow C$ we have to consider what C gives. So, we can add a rule to S.

$S \rightarrow 0A \mid 1B \mid 01$

Similarly, $B \rightarrow A$ is also a unit production so we can modify it as

$B \rightarrow 1 \mid 0S \mid 00$

Thus finally we can write CFG without unit production as

$S \rightarrow 0A \mid 1B \mid 01$

$A \rightarrow 0S \mid 00$

$B \rightarrow 1 \mid 0S \mid 00$

$C \rightarrow 01$

EXAMPLE 3. “Remove” unit productions from:

$S \rightarrow Aa \mid B$

$B \rightarrow A \mid bb$

$A \rightarrow a \mid bc \mid B$

ANSWER

$S \rightarrow Aa \mid bb \mid a \mid bc$ since S => B and S => A

$B \rightarrow bb \mid a \mid bc$ since B => A

$A \rightarrow a \mid bc \mid bb$ since A => B

But B is a useless symbol, so discard the production involving B

Example 4 :
S -> Aa | B
A -> b | B
B -> A | a Remove Unit Productions

- S -> Aa A -> b B -> a Now we find all the variables that satisfy ‘X *=> Z’. These are ‘S *=> A’ , ‘S*=>B’, ‘A *=> B’ and ‘B *=> A’. For ‘A *=> B’ , we add ‘A -> a’ because ‘B ->a’ exists in ‘G’.

- Finally we get the following grammar –

  S -> Aa | b | a
  A -> b | a
  B -> a | b

- Now remove B -> a|b , since it doesnt occur in the production ‘S’, then the following grammar becomes,

  S->Aa|b|a
  A->b|a

Example 5 :

Consider the following grammar

$S \rightarrow M \mid S + M$

$M \rightarrow F \mid M \times F$

$F \rightarrow I \mid (S)$

$I \rightarrow a \mid b \mid Ia \mid Ib$

Remove the Unit Productions

We can remove $F \rightarrow I$ and add all the productions of I and obtain

$I \rightarrow a \mid b \mid Ia \mid Ib$

$F \rightarrow a \mid b \mid Ia \mid Ib \mid (S)$

$M \rightarrow F \mid M \times F$

$S \rightarrow M \mid S + M$

$$S \rightarrow a \mid b \mid Ia \mid Ib \mid (S) \mid M \times F \mid S + M$$

$$M \rightarrow a \mid b \mid Ia \mid Ib \mid (S) \mid M \times F$$

$$F \rightarrow a \mid b \mid Ia \mid Ib \mid (S)$$

$$I \rightarrow a \mid b \mid Ia \mid Ib$$

Example 6 :
Simplify the grammar by removing the unit productions from the following grammar

S -> AB

A -> a

B -> C / b

C -> D

D -> E

E -> a

Solution:

There are 3 unit production in the grammar

B -> C

C -> D

D -> E

For production D -> E there is E -> a so we add D -> a to the grammar and add D -> E from the grammar.

Now we have C -> D so we add a production C -> a to the grammar and delete C -> D from the grammar.

Similarly we have B -> C by adding B -> a and removing B -> C we get the final

grammar free of unit production as:

S -> AB

A -> a

B -> a / b

C -> a

D -> a

E -> a

We can see that C, D and E are unreachable symbols so to get a completely reduced grammar we remove them from the CFG.

The final CFG is :

S -> AB

A -> a

B -> a / b

Example 7 :

S-> S + T | T

T-> T * F | F

F-> (S) |a

S-> S+T | T*F | (S) | a

T-> T * F | (S) | a

F-> (S) | a

Circular unit production rules

Consider the following grammar

$A \to B \mid a$

$B \to C$

$C \to A$

Substitutions will run into circular substitutions and will never finish

Simplification (Reduction) of Grammar :

- Order of eliminations We must follow the following order of eliminations.
- Eliminate epsilon productions
- Eliminate unit productions
- Eliminate useless symbols

Steps to remove null productions :

- **Step 1:** Look for the Null producions whose right side contains A
- **Step 2:** Replace each occurrence of A in each of these productions with ϵ
- **Step 3:** Now combine the result of step 2 with the original production and remove ε productions.

Removal of Unit Productions :

- The unit productions are the productions in which one non-terminal gives another non-terminal. A->B

- Use the following steps to remove unit production:

- **Step 1:** To remove **X → Y**, add production **X → a** to the grammar rule whenever **Y → a** occurs in the grammar. [x ∈ Terminal, x can be Null]

- **Step 2:** Now delete X → Y from the grammar.

- **Step 3:** Repeat step 1 and step 2 until all unit productions are removed.

Steps For reduction of a given grammar G:

1. Identify **Non-generating symbols** in the given CFG and eliminate those productions which contains non-generating symbols….Symbols not deriving any string

2. Identify **Non-reachable symbols** and eliminate those productions which contain the non-reachable symbols

Consider the following grammar

$S \rightarrow Aa \mid B$

$B \rightarrow a \mid BC$

$C \rightarrow a \mid \epsilon$

Simplify the grammar

- 1.Removal of null production

  S->Aa|B
  B->a|B|BC
  C->a

- 2.Removal of Unit production
  S->Aa|a|BC
  B->a|BC
  C->a

- 3.Removal of useless production

  S->a|BC
  B->a|BC
  C->a

G' = { S, {B , C}, {a} , P' }

Consider the following grammar

S→AC | B

A→a

C→c | BC

E-> aA | e

Simplify the grammar

- 1.Removal of null production

  There are no Null productions in the given Grammar

- 2.Removal of Unit production

  S-> AC

  A-> a

  C->c

  E-> aA | e

- 3.Removal of useless production

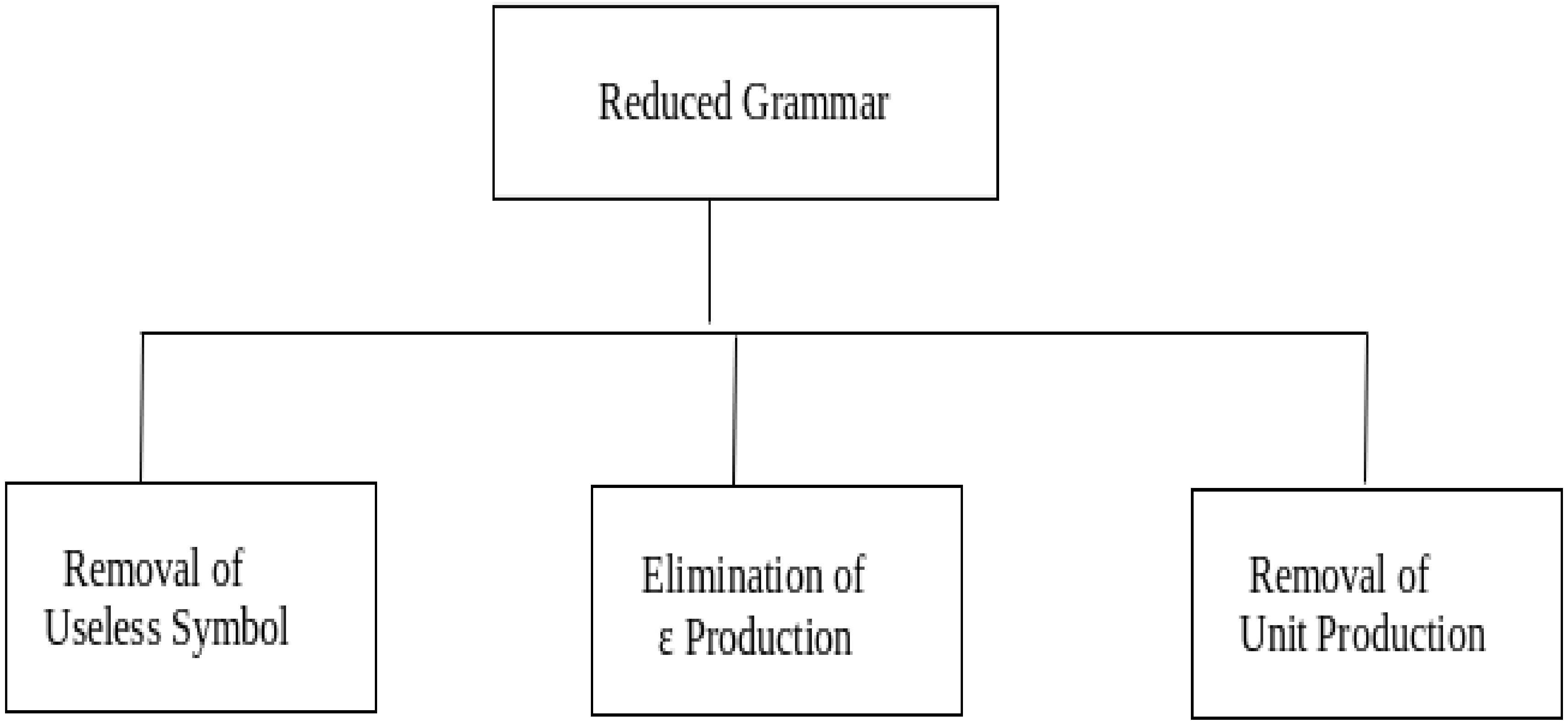
  S-> AC

  A-> a

  C-> c

G' = { S, {A , C}, {a , c} , P' }

Three ways to Simplify the Grammar :



Removal of Null Symbols :

- Null productions are of the form A -> ϵ.

- In a given CFG, we call a non-terminal N nullable if there is a production N -> ϵ or there is a derivation that starts at N and leads to ϵ:

$$N \Rightarrow \ldots \Rightarrow \epsilon.$$

- We cannot remove all ϵ-productions from a grammar if the language contains ϵ as a word, but if it doesn't we can remove all.

- The removal of ϵ-productions increases the number of rules but reduces the length of derivations.

Steps to remove null productions :

- **Step 1:** Look for the Null producions whose right side contains A

- **Step 2:** Replace each occurrence of A in each of these productions with ϵ

- **Step 3:** Now combine the result of step 2 with the original production and remove ε productions.

- If A-> ϵ is a production to be eliminated, then we look for all productions , whose RHS contains A and replace every occurrence of A in each of these productions to obtain Non-ϵ-productions .
- **The resultant Non- ϵ-productions are added to the original grammar**

Example 1 :

S-> aA

A-> b | ϵ

Eliminate ϵ-productions from this grammar

- Here, **A-> ϵ** is a Null Production.
- By putting ϵ at the place of A, we get

S-> a

Now, adding this new production to the original grammar

S-> aA

S-> a

A-> b

OR

S-> aA | a

A-> b

This grammar doesn't contain any ϵ-production

Example 2 :

S-> aSa

S-> bSb | ϵ

Eliminate ϵ-productions from this grammar

- Here, S-> ϵ is the epsilon production
- So, replacing occurrence of S by epsilon , we get

  S-> aa

  S-> bb
- Adding the new productions to the original grammar , we get

  S-> aSa | aa | bSb | bb

Example 3 :

S -> aSb/aAb/ab/a

A -> ε

- Replace NULL producing symbol with and without in R.H.S. of remaining states And drop the productions which has ε directly.

  eg. A -> ε

  S -> aSb/aAb/ab/ab/a

- But we no need to write "ab" twice So, **S -> aSb/aAb/ab/a**

Example 4:

S -> AB

A -> aAA/ε

B -> bBB/ε

- Nullale Variables are {A, B, S}
- Because start state also a Nullable symbol so ε belongs to given CFG
- We will proceed with the method:

S -> AB/A/B

A -> aAA/aA/a

B -> bAA/bB/b

Example 5 : Remove the null productions from the following grammar

S -> ABAC

A -> aA / ϵ

B -> bB / ϵ

C -> c

We have two null productions in the grammar A -> ϵ and B -> ϵ.

To eliminate A -> ϵ we have to change the productions containing A in the right side.

Those productions are S -> ABAC and A -> aA.

So replacing each occurrence of A by ϵ, we get four new productions

S -> ABC / BAC / BC

A -> a

Add these productions to the grammar and eliminate A -> ϵ.

S -> ABAC / ABC / BAC / BC

A -> aA / a

B -> bB / ϵ

C -> c

To eliminate B -> ϵ, we have to change the productions containing B on the right side of the **newly generated grammar**

Doing that we generate these new productions:

S -> AAC / AC / C
B -> b

Add these productions to the grammar and remove the production B -> ϵ from the grammar.

The new grammar after removal of ϵ – productions is:

S -> ABAC / ABC / BAC / BC / AAC / AC / C
A -> aA / a
B -> bB / b
C -> c

Example 6 : Remove the production from the following CFG by preserving the meaning of it.

$S \rightarrow XYX$

$X \rightarrow 0X \mid \varepsilon$

$Y \rightarrow 1Y \mid \varepsilon$

- Now, while removing ε production, we are deleting the rule $X \rightarrow \varepsilon$ and $Y \rightarrow \varepsilon$. To preserve the meaning of CFG we are actually placing ε at the right-hand side whenever X and Y have appeared.
- Let us take

$$S \rightarrow XYX$$

- If the first X at right-hand side is ε. Then

$$S \rightarrow YX$$

- Similarly if the last X in R.H.S. = ε. Then

$$S \rightarrow XY$$

- If $Y = \varepsilon$ then

$$S \rightarrow XX$$

- If Y and X are ε then,

$$S \rightarrow X$$

- If both X are replaced by ε

$$S \rightarrow Y$$

- Now,

$$S \rightarrow XY \mid YX \mid XX \mid X \mid Y \mid \varepsilon$$

- Now let us consider

$$X \rightarrow 0X$$

- If we place ε at right-hand side for X then,

$$X \rightarrow 0$$

$$X \rightarrow 0X \mid 0$$

- Similarly $Y \rightarrow 1Y \mid 1$

- Collectively we can rewrite the CFG with removed ε production as

$$S \rightarrow XYX \mid XY \mid YX \mid XX \mid X \mid Y \mid \varepsilon$$
$$X \rightarrow 0X \mid 0$$
$$Y \rightarrow 1Y \mid 1$$