Chomsky Normal Form

Chomsky Normal Form

A context-free grammar is in *Chomsky Normal Form* if every rule is of the form

$$A \rightarrow BC$$

$$A \rightarrow a$$

where a is any terminal and A, B, and C are any variables — except that B and C may not be the start variable.

In addition, we permit the rule $S \rightarrow \varepsilon$ where S is the start variable if the language of the grammar contains ε .

Chomsky Normal Form

Theorem: Every (non-empty) context free language can be generated by a context-free grammar in Chomsky normal form.

- In order to obtain **an equivalent grammar in Chomsky normal form** for any given CFG G, we will have the following conversion steps:
    1. **Add a new start variable S_0** and a new production rule $S_0 \rightarrow S$ where S is the original start variable of G.
    2. **Eliminate ε-productions** (productions of the form $A \rightarrow \varepsilon$). After this conversion step, only one ε-production ($S_0 \rightarrow \varepsilon$) if the language of G contains ε.
    3. **Eliminate unit productions** (productions of the form $A \rightarrow B$ where A and B are variables).
    4. **Eliminate useless symbols**. Useless symbols do not appear in any derivation of a terminal string from the start symbol.
    5. **Convert the remaining rules into Chomsky normal form** adding new variables and rules.

Add A New Start Variable

- We **add a new start variable S_0** and the **rule $S_0 \rightarrow S$** , where S was the original start variable.
- This change guarantees that the start variable does not occur on the right-hand side of a rule.
- The new grammar is equivalent to the original grammar (i.e ***they generate same language***).

Example:

$$S \rightarrow \varepsilon \mid 0S1$$

$$\mathbf{S_0 \rightarrow S}$$
$$S \rightarrow \varepsilon \mid 0S1$$

equivalent grammars

Eliminate ε-productions

nullable variables

- In order to remove **ε-productions**, first we will determine **nullable variables.**
- A variable A is said to **nullable** if $A \overset{*}{\Rightarrow} \varepsilon$.
- We can compute nullable(G), the set of all nullable symbols of a CFG G=(V,T,P,S) as follows:

Basis:

$$\text{nullable}(G) = \{A : A \rightarrow \varepsilon \in P\}$$

Induction:

If $\{C_1,\dots,C_k\} \subseteq \text{nullable}(G)$ **and** $A \rightarrow C_1,\dots,C_k \in P$,

then $\text{nullable}(G) = \text{nullable}(G) \cup \{A\}$

Eliminate ε-productions

nullable variables: Example

- A CFG G_1

  $S_0 \rightarrow S$

  $S \rightarrow \varepsilon \mid 0S1$ ➔ nullable(G_1) = $\{S, S_0\}$

- A CFG G_2

  $S_0 \rightarrow S$

  $S \rightarrow AB$

  $A \rightarrow aAA \mid \varepsilon$

  $B \rightarrow bBB \mid \varepsilon$ ➔ nullable(G_2) = $\{A, B, S, S_0\}$

Eliminate ε-productions

Steps for ε-production elimination for CFG G:

1. **Find nullable(G),** the set of all *nullable* symbols of G.
2. **Generate new rules** from a rule R by eliminating *nullable* variables from its right-side, if *nullable* variables appears on its right-side.
   - The number of new rules depends on the number of *nullable* variables on the right-side. If there are k *nullable* variables, we have to generate 2^k-1 new rules.
   - ***Generated new rule is added if it is not already among the rules***.
     - **If $R \rightarrow \alpha A \beta$ is a rule and A is the only *nullable* variable on $\alpha A \beta$, generate and add the new rule $R \rightarrow \alpha\beta$.**
     - **If $R \rightarrow \alpha A \beta B \gamma$ is a rule and A and B are only *nullable* variables on $\alpha A \beta B \gamma$, generate and add the new rules $R \rightarrow \alpha\beta B\gamma$, $R \rightarrow \alpha A \beta\gamma$ and $R \rightarrow \alpha\beta\gamma$.**
     - …
3. **Remove all ε-productions** A→ε (except $S_0 \rightarrow \varepsilon$) from the rules.

The new grammar that is obtained by **eliminating ε-productions** is equivalent to the original grammar (i.e ***they generate same language***).

Eliminate ε-productions: *Example*

$\mathbf{S_0 \rightarrow S} \qquad \mathbf{S \rightarrow \varepsilon \mid 0S1}$

- nullable(G) = $\{S, S_0\}$
- Since S is nullable,
  - generate $S_0 \rightarrow \varepsilon$ from $S_0 \rightarrow S$
- Since S is nullable,
  - generate $S \rightarrow 01$ from $S \rightarrow 0S1$
- After all generations, we have the following rules:

  $S_0 \rightarrow \varepsilon \mid S$

  $S \rightarrow \varepsilon \mid 01 \mid 0S1$
- Remove all ε-productions except $S_0 \rightarrow \varepsilon$, our final grammar is:

$\mathbf{S_0 \rightarrow \varepsilon \mid S}$

$\mathbf{S \rightarrow 01 \mid 0S1}$

equivalent grammars

Eliminate ε-productions: *Example*

$\mathbf{S_0 \to S} \qquad \mathbf{S \to AB} \qquad \mathbf{A \to aAA \mid \cancel{\varepsilon}} \qquad \mathbf{B \to bBB \mid \cancel{\varepsilon}}$

- nullable(G) = $\{A, B, S, S_0\}$
- Generate new rules:

$S_0 \to \varepsilon$			from $S_0 \to S$
$S \to A$	$S \to B$	$S \cancel{\to} \varepsilon$	from $S \to AB$
$A \to aA$	$A \cancel{\to} aA$	$A \to a$	from $A \to aAA$
$B \to bB$	$B \cancel{\to} bB$	$B \to b$	from $B \to bBB$

- Remove ε-productions

$\mathbf{S_0 \to S \mid \varepsilon}$

$\mathbf{S \to AB \mid A \mid B}$

$\mathbf{A \to aAA \mid aA \mid a}$

$\mathbf{B \to bBB \mid bB \mid b}$

equivalent grammars

Eliminate Unit Productions

- **A → B is a unit production**, whenever A and B are variables.
- Unit productions can be eliminated from a grammar to obtain a grammar without unit productions.
  - The resulting grammar that is obtained by eliminating unit productions will be equivalent to the original grammar.

- We will remove unit productions one by one from the grammar.
- **Remove a unit production A → B from the grammar.**
  - **Then, whenever a rule B→u appears, we add the rule A→u unless this was a unit rule previously removed.**
- **We repeat these steps until we eliminate all unit rules.**

Eliminate Unit Productions: *Example*

$S_0 \rightarrow \varepsilon \mid S$

$S \rightarrow 01 \mid 0S1$

- Unit productions: { $S_0 \rightarrow S$ }
- Remove $S_0 \rightarrow S$,
  - Add $S_0 \rightarrow 01$ and $S_0 \rightarrow 0S1$
- The resulting grammar after eliminating unit productions.

$S_0 \rightarrow \varepsilon \mid 01 \mid 0S1$

$S \rightarrow 01 \mid 0S1$

equivalent grammars

Eliminate Unit Productions: *Example*

$S_0 \rightarrow S \mid \varepsilon$
$S \rightarrow AB \mid A \mid B$
$A \rightarrow aAA \mid aA \mid a$
$B \rightarrow bBB \mid bB \mid b$

- Unit productions: { $S_0 \rightarrow S$, $S \rightarrow A$, $S \rightarrow B$ }
- Remove $S \rightarrow B$, add $S \rightarrow bBB \mid bB \mid b$
- Remove $S \rightarrow A$, add $S \rightarrow aAA \mid aA \mid a$
- Remove $S_0 \rightarrow S$, add $S_0 \rightarrow AB \mid aAA \mid aA \mid a \mid bBB \mid bB \mid b$

- The resulting grammar after eliminating unit productions.

$S_0 \rightarrow \varepsilon \mid AB \mid aAA \mid aA \mid a \mid bBB \mid bB \mid b$
$S \rightarrow AB \mid aAA \mid aA \mid a \mid bBB \mid bB \mid b$
$A \rightarrow aAA \mid aA \mid a$
$B \rightarrow bBB \mid bB \mid b$

equivalent grammars

Eliminate Unit Productions: *Example*

E → E+T | T
T → T*F | F
F → G^F | G
G → id | (E)

- Unit productions: { **E→T, T→F, F→G** }
- Remove **F→G,** add **F → id | (E)**
- Remove **T→F,** add **T → G^F | id | (E)**
- Remove **E→T,** add **E→ T*F | G^F | id | (E)**

- The resulting grammar after eliminating unit productions.

E → E+T | T*F | G^F | id | (E)
T → T*F | G^F | id | (E)
F → G^F | id | (E)
G → id | (E)

equivalent grammars

Eliminate Unit Productions: *Example*

Eliminating unit productions in different order do not change the result.

E → E+T | T
T → T*F | F
F → G^F | G
G → id | (E)

- Unit productions: { **E→T, T→F, F→G** }
- Remove **E→T,** add **E → T*F | F**
- Remove **T→F,** add **T → G^F | G**
- Remove **F→G,** add **F → id | (E)**
- Remove newly introduced unit productions
- Remove **E→F,** add **E → G^F | id | (E)**
- Remove **T→G,** add **T → id | (E)**

- The resulting grammar after eliminating unit productions.

E → E+T | T*F | G^F | id | (E)
T → T*F | G^F | id | (E)
F → G^F | id | (E)
G → id | (E)

equivalent grammars

Eliminate Useless Symbols

- A symbol X is useful for a grammar G=(V,T,P,S), if there is a derivation

  $$S \overset{*}{\Rightarrow} \alpha X \beta \overset{*}{\Rightarrow} w$$

  for a terminal string w.
- Symbols that are not useful are called useless.

- A symbol X is generating if $X \overset{*}{\Rightarrow} w$ for some string $w \in T^*$.

- A symbol X is reachable if $S \overset{*}{\Rightarrow} \alpha X \beta$ for some $\{\alpha,\beta\} \subseteq (V \cup T)^*$.

- If we eliminate non-generating symbols first, and then non-reachable symbols, we will be left with only useful symbols.
  - The grammar that is obtained by eliminating useless symbols will be equivalent to the original grammar.

Eliminate Useless Symbols

computing generating symbols

- For a grammar G = (V,T,P,S), the generating symbols **generating(G)** are computed by the following closure algorithm:

Basis: **generating(G) = T**

Induction:

If **$X \rightarrow \varepsilon \in P$ or $X \rightarrow A_1 \ldots A_n \in P$ where $\{A_1, \ldots, A_n\} \subseteq$ generating(G)** then

generating(G) = generating(G) ∪ {X}

Example:

- Let G be $S \rightarrow AB|a$, $A \rightarrow b$
- Initially, generating(G) = {a,b}
- A will be in generating(G) because of $A \rightarrow b$
- S will be in generating(G) because of $S \rightarrow a$

- Thus, generating(G)={a,b,A,S} and non-generating symbols are {B}

Eliminate Useless Symbols

computing reachable symbols

- For a grammar G = (V,T,P,S), the reachable symbols **reachable(G)** are computed by the following closure algorithm:

Basis: **reachable(G) = {S}**

Induction:

If **$X \in$reachable(G) and $X \rightarrow \alpha \in P$** then

add all symbols in α to reachable(G).

Example:

- Let G be $S \rightarrow AB|a$, $A \rightarrow b$, $C \rightarrow a$
- Initially, reachable(G) = {S}
- A and B will be in reachable(G) because of $S \rightarrow AB$
- a will be in reachable(G) because of $S \rightarrow a$
- b will be in reachable(G) because of $A \rightarrow b$

- Thus, reachable(G)={S,A,B,a,b} and non-reachable symbols are {C}

Eliminate Useless Symbols

Steps to eliminate useless symbols from G = (V,T,P,S) :

1. Compute generating(G).
2. Remove all productions containing at least one non-generating symbol in order to create a new grammar G_1 (a grammar without non-generating symbols).
   - Remove a production if a non-generating symbol appears in that production (on its right-side or its left-side)
3. Compute reachable(G_1).
4. Remove all productions containing at least one non-reachable symbol in order to create a new grammar G_2 without useless symbols (a grammar without non-reachable symbols and non-generating symbols).

The new grammar G_2 (a grammar without useless symbols) will be equivalent to the original grammar G.

Eliminate Useless Symbols: *Example*

G: $S \rightarrow AB \mid a$, $A \rightarrow b$

- **Compute generating(G):**
  - generating(G)={a,b,A,S} and **non-generating symbols are {B}.**
- **Remove productions containing non-generating symbols:**
  - Remove **S→AB** because it contains B.
  - Thus, following G_1 is a grammar without non-generating symbols.
  - G_1 is **S→a, A→b**
- **Compute reachable(G_1):**
  - reachable(G_1)={S,a} and **non-reachable symbols are {A,b}.**
- **Remove productions containing non-reachable symbols:**
  - Remove **A→b** because it contains A (and/or b).
- **Grammar G_2 without useless symbols** (non-generating and non-reachable symbols):

G_2: S→a

Chomsky Normal Form (CNF)

Convert the remaining rules into CNF

- **Steps to obtain an equivalent grammar in Chomsky Normal Form:**

1. Add a new start variable S_0.
2. Eliminate ε-productions.
3. Eliminate unit productions.
4. Eliminate useless symbols.
   i. Eliminate non-generating symbols.
   ii. Eliminate non-reachable symbols.

cleanup steps

5. **Convert the remaining rules into CNF:**

   Now, to obtain a grammar in CNF, we want every rule to be the form

   $$A \to BC \qquad A \to a$$

   i. Arrange that all bodies of length 2 or more consists of only variables.

   ii. Break bodies of length 3 or more into a cascade of two-variable-bodied productions.

Chomsky Normal Form (CNF)

Convert the remaining rules into CNF

i. **Arrange that all bodies of length 2 or more consists of only variables.**
   - For every terminal a that appears in a body of length≥2, create a new variable, say X_a, and replace a by X_a in all bodies.
   - Then add a new rule $X_a \rightarrow a$.

ii. **Break bodies of length 3 or more into a cascade of two-variable-bodied productions.**
   - For each rule of the form

$$A \rightarrow B_1, \ldots, B_k$$

k≥3, introduce new variables $Y_1, \ldots, Y_{k-2}$ and replace the rule with

$$A \rightarrow B_1 Y_1$$
$$Y_1 \rightarrow B_2 Y_2$$
$$\ldots$$
$$Y_{k-3} \rightarrow B_{k-2} Y_{k-2}$$
$$Y_{k-2} \rightarrow B_{k-1} B_k$$

Chomsky Normal Form (CNF)

Convert the remaining rules into CNF: Example

$S_0 \rightarrow \varepsilon \mid 01 \mid 0S1$

$S \rightarrow 01 \mid 0S1$

already cleaned grammar

Arrange that all bodies of length 2 or more consists of only variables.

$S_0 \rightarrow \varepsilon \mid X_0 X_1 \mid X_0 S X_1$

$S \rightarrow X_0 X_1 \mid X_0 S X_1$

$X_0 \rightarrow 0$

$X_1 \rightarrow 1$

Break bodies of length 3 or more into two-variable-bodied productions.

$S_0 \rightarrow X_0 S X_1$ ➔ $S_0 \rightarrow X_0 Y_1$ $Y_1 \rightarrow S X_1$

$S \rightarrow X_0 S X_1$ ➔ $S \rightarrow X_0 Y_2$ $Y_2 \rightarrow S X_1$

Grammar in CNF:

$S_0 \rightarrow \varepsilon \mid X_0 X_1 \mid X_0 Y_1$

$S \rightarrow X_0 X_1 \mid X_0 Y_2$

$X_0 \rightarrow 0$

$X_1 \rightarrow 1$

$Y_1 \rightarrow S X_1$

$Y_2 \rightarrow S X_1$

Chomsky Normal Form (CNF)

Converting into CNF: A Full Example

$S \to ABA$
$A \to aA \mid \varepsilon$
$B \to bBc \mid \varepsilon$

Step 1. Add a new start variable S_0

$S \to ABA$
$A \to aA \mid \varepsilon$
$B \to bBc \mid \varepsilon$

$S_0 \to S$
$S \to ABA$
$A \to aA \mid \varepsilon$
$B \to bBc \mid \varepsilon$

Step 2. Eliminate ε-productions.

nullable(G) = $\{A, B, S, S_0\}$

$S_0 \to S$
$S \to ABA$
$A \to aA \mid \varepsilon$
$B \to bBc \mid \varepsilon$

$S_0 \to S \mid \varepsilon$
$S \to ABA \mid BA \mid AA \mid AB \mid A \mid B \mid A \mid \varepsilon$
$A \to aA \mid a \mid \varepsilon$
$B \to bBc \mid \varepsilon \mid bc$

$S_0 \to S \mid \varepsilon$
$S \to ABA \mid BA \mid AA \mid AB \mid A \mid B$
$A \to aA \mid a$
$B \to bBc \mid bc$

Chomsky Normal Form (CNF)

Converting into CNF: A Full Example

Step 3. Eliminate unit productions.

S_0 → S | ε
S → ABA | BA | AA | AB | A | B
A → aA | a
B → bBc | bc

S_0 → ε | ABA | BA | AA | AB | aA | a | bBc | bc
S → ABA | BA | AA | AB | aA | a | bBc | bc
A → aA | a
B → bBc | bc

Step 4. Eliminate useless symbols.

i. **Eliminate non-generating symbols. none**

ii. **Eliminate non-reachable symbols. S**

S_0 → ε | ABA | BA | AA | AB | aA | a | bBc | bc
S → ABA | BA | AA | AB | aA | a | bBc | bc
A → aA | a
B → bBc | bc

S_0 → ε | ABA | BA | AA | AB | aA | a | bBc | bc
A → aA | a
B → bBc | bc

Chomsky Normal Form (CNF)

Converting into CNF: A Full Example

Step 5. Convert the remaining rules into CNF:

Arrange that all bodies of length 2 or more consists of only variables.

$S_0 \rightarrow \varepsilon \mid ABA \mid BA \mid AA \mid AB \mid aA \mid a \mid bBc \mid bc$
$A \rightarrow aA \mid a$
$B \rightarrow bBc \mid bc$

$S_0 \rightarrow \varepsilon \mid ABA \mid BA \mid AA \mid AB \mid XA \mid a \mid YBZ \mid YZ$
$A \rightarrow XA \mid a$
$B \rightarrow YBZ \mid YZ$
$X \rightarrow a$
$Y \rightarrow b$
$Z \rightarrow c$

Break bodies of length 3 or more into two-variable-bodied productions.

$S_0 \rightarrow \varepsilon \mid ABA \mid BA \mid AA \mid AB \mid XA \mid a \mid YBZ \mid YZ$
$A \rightarrow XA \mid a$
$B \rightarrow YBZ \mid YZ$
$X \rightarrow a$
$Y \rightarrow b$
$Z \rightarrow c$

$S_0 \rightarrow \varepsilon \mid AC \mid BA \mid AA \mid AB \mid XA \mid a \mid YD \mid YZ$
$C \rightarrow BA$ $\qquad D \rightarrow BZ$
$A \rightarrow XA \mid a$
$B \rightarrow YE \mid YZ$
$E \rightarrow BZ$
$X \rightarrow a$
$Y \rightarrow b$
$Z \rightarrow c$

Chomsky Normal Form (CNF)

Converting into CNF: A Full Example

Grammar in CNF:

$S_0 \rightarrow \varepsilon \mid AC \mid BA \mid AA \mid AB \mid XA \mid a \mid YD \mid YZ$

$C \rightarrow BA$

$D \rightarrow BZ$

$A \rightarrow XA \mid a$

$B \rightarrow YE \mid YZ$

$E \rightarrow BZ$

$X \rightarrow a$

$Y \rightarrow b$

$Z \rightarrow c$