NORMAL FORMS

(PART II)

Recall: Any context-free grammar can be converted into either Chomsky Normal Form and/or Greibach Normal Form.

Each rule is of the form $A \to c$ or $A \to BC$

Each rule is of the form $A \to cB_1 \cdots B_k$.

Neither form allows the generation of the empty string ε.

We showed that to convert a given CFG into either normal form, we start by eliminating ε-productions and unit-productions.

$A \to \varepsilon$ $A \to B$.

Consider a context-free language $\mathcal{L}$.

We showed that there is a CFG G which generates $\mathcal{L}$ s.t. G does not contain any ε- or unit-productions.

So every rule is of the form $A \rightarrow \gamma_1 \gamma_2 \cdots \gamma_n$, where $A \in NT$ and $\gamma_i \in NT \cup T$, for all $i \in \{1, \ldots, n\}$.

For Chomsky Normal Form, we said that every rule must be of the form $A \rightarrow c$, or $A \rightarrow BC$, where $c \in T$, and $A, B, C \in NT$.

Can we obtain a G' s.t. $\mathcal{L}(G) = \mathcal{L}(G') = \mathcal{L}$, where G' is in Chomsky Normal Form?

$G' = (NT', T, R', S)$, where

$NT_1 = NT \cup \{A_c \mid c \in T\}$

add a new non-terminal symbol for each letter in the alphabet

$R_1 = \{A_c \to c \mid c \in T\}$

$$R_2 = R \setminus \left\{ A \to \gamma_1 \cdots \gamma_{i-1}\, c\, \gamma_{i+1} \cdots \gamma_n \;\middle|\; \begin{array}{l} c \in T, \text{ and} \\ \text{for some } j \neq i,\ \gamma_j \neq \varepsilon \end{array} \right\}$$

$$\cup \left\{ A \to \gamma_1 \cdots \gamma_{i-1}\, A_c\, \gamma_{i+1} \cdots \gamma_n \;\middle|\; \begin{array}{l} c \in T, \text{ and} \\ \text{for some } j \neq i,\ \gamma_j \neq \varepsilon \end{array} \right\} \cup R_1$$

All rules in R_2 are of the form $A \to c$ $(A \in NT, c \in T)$, or $A \to B_1 \cdots B_k$ $(A, B_1, \ldots, B_k \in NT, k \geq 2)$.

Consider a rule of the form $A \to B_1 \cdots B_k$, where $A, B_1, \ldots, B_k \in NT$ and $k > 2$.

Need to convert this into a rule with only two non-terminal symbols on the right.

For every rule of the above form in R_2, choose a new non-terminal symbol $C \notin NT_1$, and replace this rule by the following two rules.

$$\{A \to B_1 C, \;\; C \to B_2 \cdots B_k\}.$$

Do this repeatedly till all such rules have exactly two non-terminal symbols on the right.

NT' is the final set of non-terminals, and R' the final set of rules.

Example: $G = (\{S\}, \{a,b\}, R, S)$, where

$R = \{S \to \varepsilon,\ S \to aSb,\ S \to bSa,\ S \to SS\}$.

Convert G into G', where G' does not contain ε- or unit-productions.

① $\{S \to \varepsilon,\ S \to aSb\} \subseteq R \Rightarrow S \to ab \in R'$

② $\{S \to \varepsilon,\ S \to bSa\} \subseteq R \Rightarrow S \to ba \in R'$

So $R' = R \setminus \{S \to \varepsilon\} \cup \{S \to ab,\ S \to ba\}$

$= \{S \to aSb,\ S \to bSa,\ S \to SS,\ S \to ab,\ S \to ba\}$

We now convert G' into Chomsky Normal Form.

$$G_{cnf} = (NT', T, R', S), \text{ where}$$

$$NT' = \{S, A, B, C_1, C_2\}$$

$$T = \{a, b\}$$

$$R' = \left\{ \begin{array}{l} S \rightarrow AC_1, \; S \rightarrow BC_2, \\ S \rightarrow SS, \; S \rightarrow AB, \; S \rightarrow BA, \\ A \rightarrow a, \; B \rightarrow b, \\ C_1 \rightarrow SB, \; C_2 \rightarrow SA, \\ S \rightarrow \varepsilon \end{array} \right\}$$

So now we can convert a given CFG into Chomsky Normal Form.

What about Greibach Normal Form?

Here, every rule needs to be of the form $A \to c B_1 \cdots B_k$,
where $k \geq 0$, $A, B_1, \ldots, B_k \in NT$, and $c \in T$.

So the $A \to c$ kind of rules in the Chomsky Normal Form is fine
But how do we handle the $A \to BC$ kind of rules?

Suppose we have a grammar in Chomsky Normal Form.

Enumerate the non-terminals as $N_1, \ldots, N_k$ s.t. $|NT| = k$.

Modify the rules in R such that if $N_i \to N_j \gamma$, then $i < j$.

What if we end up with something of the form $N_i \to N_i \gamma$?

Suppose we have a bunch of left recursive rules of the form

$$A \to A\gamma_1 \mid A\gamma_2 \mid \dots \mid A\gamma_k$$

and some other rules of the form $A \to \alpha_1 \mid \alpha_2 \mid \dots \mid \alpha_\ell$.

Then, we add new rules of the form

$$A \to \alpha_j \mid \alpha_j B \quad \text{for } 1 \le j \le \ell \text{, and}$$

$$B \to \gamma_i \mid \gamma_i B \quad \text{for } 1 \le i \le k.$$

Then turn each of these into $A \to cB_1 \cdots B_k$ form.

Claim: This transformation preserves the language generated by the grammar. Exercise!

Example:

$S \to rA \mid BB$
$B \to r \mid SB$
$A \to t$

$R \to r \quad T \to t$
$S \to RA \mid BB$
$B \to r \mid SB$
$A \to t$

$A_1 = S, \quad A_2 = A, \quad A_3 = B$

$A_1 \to rA_2 \mid A_3 A_3$
$A_3 \to r \mid A_1 A_3$ → problem: $i > j$
$A_2 \to t$

$A_3 \to r \mid rA_2A_3 \mid A_3A_3A_3$ left recursive

$A_3 \to r \mid rB \mid rA_2A_3 \mid rA_2A_3B$ $\qquad B \to A_3A_3 \mid A_3A_3B$

$A_1 \to rA_2 \mid \underline{A_3A_3}, \quad A_2 \to t,$ not in the form $A \to cB_1 \cdots B_k$.
$A_3 \to r \mid rB \mid rA_2A_3 \mid rA_2A_3B$ $\qquad B \to \underline{A_3A_3} \mid \underline{A_3A_3}B.$

$$A_1 \longrightarrow rA_2 \mid rA_3 \mid rBA_3 \mid rA_2A_3A_3 \mid rA_2A_3BA_3$$

$$A_2 \longrightarrow t$$

$$A_3 \longrightarrow r \mid rB \mid rA_2A_3 \mid rA_2A_3B$$

$$B \longrightarrow rA_3 \mid rBA_3 \mid rA_2A_3A_3 \mid rA_2A_3BA_3 \mid$$
$$rA_3B \mid rBA_3B \mid rA_2A_3A_3B \mid rA_2A_3BA_3B$$