Chomsky Normal Form

A, B, C are variables, B and C are not the start symbol and a is a terminal

A grammar G is in Chomsky normal form if every rule is in the form $A \to a$ or $A \to BC$

(Exception: If A is the start variable and if ε is in the language then $A \to \varepsilon$ is allowed)

Also, G has no useless symbols.

Any context-free grammar can be put in Chomsky normal form

<u>Uses</u>:

1. any string of length n>0 can be derived in 2n-1 steps
   Provides an (inefficient) parsing algorithm that works for *any* CFL.
   CYK algorithm: determines whether a string can be generated by a particular grammar (membership problem)
2. Parse trees are binary trees

Chomsky Normal Form: algorithm

A, B and R are variables. α, β, χ are strings of variables and terminals

1. Optional: if ε is in the language then:
   Add a new start symbol S_0 and the rule $S \to S_0$ where S was the initial start symbol
2. Eliminate null productions (except from the start symbol if ε is in language)
3. Eliminate unit rules :
   1. Eliminate rules in the form $A \to B$
   2. Then, whenever $B \to \alpha$, add the rule $A \to \alpha$ unless this was a unit rule previously removed
4. Convert the remaining rules in proper form.
   1. Replace each rule $A \to u_1u_{2..}\, u_k$ where $k \geq 3$ and each u_i is a variable or a terminal
      with the rules $A \to u_1A_1$, $A_1 \to u_2A_2$, …. $A_{k-2} \to u_{k-1}\, u_k$ where the A_i's are new variables
   2. If $k \geq 2$ replace any terminal α_i in the preceding rules with the new variable U_i and add the rule $U_i \to u_i$

CNF: Example

$S \to ASA \mid aB$
$A \to B \mid S$
$B \to b \mid \varepsilon$

1) ε not in L, so step can be omitted

2) Eliminate ε productions

$S \to ASA \mid aB \mid \underline{\mathbf{a}}$
$A \to B \mid S \mid \underline{\boldsymbol{\varepsilon}}$
$B \to b \mid \cancel{\varepsilon}$

This rule is useless so we can delete it straight away

$S \to ASA \mid aB \mid a \mid \underline{\mathbf{SA}} \mid \underline{\mathbf{AS}} \mid \cancel{\underline{\mathbf{S}}}$
$A \to B \mid S \mid \cancel{\varepsilon}$
$B \to b$

3) Eliminate unit rules from A

$S \to ASA \mid aB \mid a \mid SA \mid AS$
$A \to \cancel{B \mid S} \mid \underline{\mathbf{b}} \mid \underline{\mathbf{ASA}} \mid \underline{\mathbf{aB}} \mid \underline{\mathbf{a}} \mid \underline{\mathbf{SA}} \mid \underline{\mathbf{AS}}$
$B \to b$

4) Make substitutions and add variables as needed

$S \to \cancel{ASA \mid aB} \mid a \mid SA \mid AS \mid \underline{\mathbf{AC}} \mid \underline{\mathbf{DB}}$
$A \to b \mid \cancel{ASA \mid aB} \mid a \mid SA \mid AS \mid \underline{\mathbf{AC}} \mid \underline{\mathbf{DB}}$
$B \to b$
$\underline{\mathbf{C \to SA}}$
$\underline{\mathbf{D \to a}}$

So final grammar in Chomsky Normal Form is

$S \to a \mid SA \mid AS \mid AC \mid DB$
$A \to b \mid a \mid SA \mid AS \mid AC \mid DB$
$B \to b$
$C \to SA$
$D \to a$

Greibach Normal Form

A, B_i's C are variables, Bi is a string of
A grammar is in Greibach normal form if every rule is in the form
$A \rightarrow a$ or $A \rightarrow aB_1B_2..B_n$
Exception: If A is the start variable and if ε is in the language then $A \rightarrow \varepsilon$ is allowed)

Any context-free grammar can be put in Greibach normal form

Uses:

1. any string of length n>0 can be derived in n steps !
2. Method for transforming any CFG into a PDA with no epsilon transition

Greibach Normal Form: general idea

Full algorithm is a little tedious, but here is the general idea:

- Eliminate all left recursions
- Remove null productions except for the start symbol
- Make substitutions to transform the grammar into the proper form:
  - expand the first variable of each rule until we get a terminal on the left
  - short cut cycles if we cannot reach a terminal

GNF: example

$S \rightarrow AB \mid B$
$A \rightarrow Aa \mid b$
$B \rightarrow Ab \mid c$

1) eliminate left recursions

$A \rightarrow Aa \mid b$ is replaced with $A \rightarrow bR$
$R \rightarrow aR \mid \varepsilon$

2) Remove ε productions

$S \rightarrow AB \mid B$
$A \rightarrow bR \mid$ **b**
$R \rightarrow aR \mid \cancel{\varepsilon} \mid$ **a**
$B \rightarrow Ab \mid c$

3) Make substitutions to obtain the final form of the grammar

First, we substitute variables occurring on the left of the right hand side by its rules

$S \rightarrow AB \mid \cancel{B} \mid$ **Ab** $\mid$ **c**
$A \rightarrow bR \mid$ **b**
$R \rightarrow aR \mid a$
$B \rightarrow Ab \mid c$

$S \rightarrow$ ~~AB | Ab~~ | c | **bRB** | **bB** | **bRb** | **bb**
~~A → bR | b~~ (useless symbol, no longer needed)
$R \rightarrow aR \mid a$
$B \rightarrow Ab \mid c$

Then we do some renaming and create a variable X with the rule b, and here is our final grammar in GNF.

$S \rightarrow c \mid bRB \mid bB \mid bRX \mid bX$
$R \rightarrow aR \mid a$
$B \rightarrow c \mid bRX \mid bX$
$X \rightarrow b$