Convert PDA to CFG

- Given a PDA $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, \{\} \rangle$, we construct a CFG $G = \langle V, \Sigma, P, S \rangle$
  - $V = \{[q, A, p] \mid q, p \text{ are in } Q, A \text{ is in } \Gamma + \{S\}\}$
  - $P$ is a set of productions constructed as follows:
    - $S \rightarrow [q_0, Z_0, q]$ for all $q$ in $Q$
    - for each $(p, \varepsilon)$ in $\delta(qa, A)$, where $a$ is in $\Sigma + \{\varepsilon\}$, we define production:
      $[q, A, p] \rightarrow a$
    - for each $(p, B_1B_2\ldots B_m)$ in $\delta(qa, A)$, where $a$ is in $\Sigma + \{\varepsilon\}$ and $B$ are in $\Gamma$
      $[q, A, q_{m+1}] \rightarrow a [q_1, B_1, q_2] [q_2, B_2, q_3] \ldots [q_m, B_m, q_{m+1}]$
      for all $q, q_1, q_2, \ldots, q_{m+1}$ in $Q$
Deterministic PDA’s

• A PDA is deterministic if there is at most one move for any input symbol, any top stack symbol, and at any state.
• A no move or move without advancing input string are possible in a deterministic PDA.
• A move without advancing input string implies that no other move exists.
• The pushdown automaton for $wcw^R$ is deterministic.
Deterministic PDA’s

- The set of languages accepted by deterministic PDD’s is only a subset of the languages accepted by nondeterministic PDA’s.
- A regular language is language accepted by a DPDA.
- Not all languages accepted by DPAD are regular.
- All languages accepted by DPAD have an unambiguous CFG.
- Not all unambiguous languages are accepted by DPAD.
Grammatical Format

• Eliminate useless symbols
  – derivable from S
  – can derive a string of terminals

• Eliminate $\varepsilon$-productions
  – $A \rightarrow \varepsilon$ is an $\varepsilon$-production.

• Eliminate unit productions
  – $A \rightarrow B$ is a unit production if both $A$ and $B$ are variables.

• For any CFG $G$ and $\varepsilon$ is not in $L(G)$, there exists an equivalent context-free grammar with no useless symbol, $\varepsilon$-production, and unit production.
Eliminating Useless Symbols

• Given a CFG $G = (V, T, P, S)$, with $L(G)$ is not empty, we can effectively find an equivalent CFG $G’ = (V’, T, P’, S)$ such that for each $A$ in $V’$ there is some $w$ in $T^*$ for which $A=>*w$.

• Given a CFG $G = (V, T, P, S)$, we can effectively find an equivalent CFG $G’ = (V’, T’, P’, S)$ such that for each $X$ in $(V’+T)$ there exist $\alpha$ and $\beta$ in $(V’+T’)^*$ for which $S=>*\alpha X\beta$. 
Eliminating $\varepsilon$-productions

• If $\varepsilon$ is not in $L(G)$, we can eliminate all $\varepsilon$-productions.

• A variable is nullable if $A \Rightarrow^* \varepsilon$.

• For $A \Rightarrow X_1X_2\ldots X_n$, add the following productions $A \Rightarrow Y_1Y_2\ldots Y_n$
  – If $X_i$ is not nullable, then $Y_i = X_i$.
  – If $X_i$ is nullable, then $Y_i = X_i$ or $Y_i = \varepsilon$.
  – If $X_i$ is nullable for $(i = 1, \ldots, n)$, don’t add $A \Rightarrow \varepsilon$. 
Eliminating Unit Productions

• If $A \rightarrow *B$ and for all nonunit production $B \rightarrow X$, add
  – $A \rightarrow X$

• Eliminate all unit productions.
Eliminating Useless Symbols and $\varepsilon$ and Unit Productions

1. Eliminate $\varepsilon$-productions
2. Eliminate unit productions
3. Eliminate useless symbols.
Chomsky Normal Form (CNF)

- If aCFG has only productions of the form, it is called Chomsky Normal Form (CNF)
  - A -> BC, or
  - A -> a
    where A, B, and C are variables and a is a terminal.

- Any context-free language with e can be generated by a context-free grammar in which all productions are in CNF.