Example: Context-free Grammar

Rules
- E → a
- E → EAE
- A → +
- A → *

Terminals: + * a
Variables: E A
Start Variable: E

Context-free since can apply rule to variable in any context.
Use rules starting from start variable to derive strings.

Designing CFG's

Ex. \{ 0^n 1^n | n ≥ 0 \}

Ex. \{ 0^n 1^n | n ≥ 0 \} \cup \{ 1^n 0^n | n ≥ 0 \}

Ex. \{ w w^r | w \in \{0,1\}^* \} \text{ where } r \text{ is the reverse op.}
**Parse Trees**

Similar to derivation, ignores order of replacement

Rule: \( A \rightarrow X_1 X_2 X_3 \)

Parse tree:

```
     A
   /   \
X1    X2  X3
```

Ex.  

```
   E
  / \
E   A
 /   \
|    E
|    a
|   + E
|   / \
|  a * a
```

*Unique leftmost (rightmost) derivation

Parse tree*
Ambiguity of a Grammar

Ex. \( a + a^*a \)

\[
\begin{array}{c}
E \\
E \quad A \\
a \\
E \quad A \\
a \quad * \\
a
\end{array}
\quad
\begin{array}{c}
E \\
E \quad A \\
a \\
E \quad A \\
a \quad + \\
a
\end{array}
\]

Leaves the same: derive same string
Correspond to \( a + (a^*a) \) and \( (a + a) \quad * \quad a \)
May yield different answers if evaluated!

Def. \( G \) is ambiguous if there is some string \( w \) in \( \Sigma^* \) with two different parse trees.

Ambiguity of a Grammar?

Ex. \( S \rightarrow (S) \mid SS \mid \varepsilon \) where \( \Sigma \) is \{ \( \varepsilon, \), \( ) \} \)

What is \( L(G) \)?
Is \( G \) ambiguous?

Ex. \( E \rightarrow E + E \mid ExE \mid (E) \mid a \)

Is \( G \) ambiguous?

Ex. \( S \rightarrow \) if \( E \) then \( S \) \mid if \( E \) then \( S \) else \( S \) \mid other \)
where \( \Sigma \) is \{ if, then, else, other \}

Is \( G \) ambiguous?
Regular Languages and CFG's

Given NFA, can construct CFG with same language

\[ \text{State of NFA} \quad \text{Variable} \]

\[ \Sigma \quad \Sigma \]

\[ \text{Edge} \quad q_1 \overset{a}{\rightarrow} q_2 \quad Q_1 \rightarrow a \ Q_2 \quad \text{Rule} \]

\[ q_1 \overset{\epsilon}{\rightarrow} q_2 \quad Q_1 \rightarrow Q_2 \quad \text{Rule} \]

\[ q_0 \text{ (start)} \quad Q_0 \text{ (start)} \]

\[ q_2 \text{ is final} \quad Q_2 \rightarrow \epsilon \quad \text{Rule} \]

Ex.

\[
\begin{array}{c}
q_0 \quad b \\
q_1 \quad b \\
q_2 \quad b \\
q_3 \quad a,b
\end{array}
\]

Chomsky Normal Form

Def. A CFG is in **CNF** if all rules are of the form

- \( A \rightarrow BC \) \( B, C \) variables, not \( S \)
- \( A \rightarrow \alpha \) \( \alpha \) terminal
- \( S \rightarrow \epsilon \)

Th. For any CFG \( G \), there is an equivalent CFG \( G' \) in Chomsky normal form.

Proof Idea: Assume no \( \epsilon \) rules.

1. Add new start symbol \( S_0 \) and new rule \( S_0 \rightarrow S \).
   (So \( S_0 \) not on any RHS)
2. Eliminate unit rules \( A \rightarrow B \).
3. Replace other rules \( A \rightarrow B_1 \ B_2 \ldots \ B_m \)
   First, get all variables on RHS, then get in form
Example, Chomsky Normal Form

V \rightarrow bA
A \rightarrow a
A \rightarrow aV
A \rightarrow bAA

1. Add S \rightarrow V, S new start variable.
2. Eliminate unit rule S \rightarrow V:
   - V appears on LHS of rule V \rightarrow bA
   - Add the rule S \rightarrow bA
     (Only add non-unit rules.)
3. RHS all variables: For any rule with non-terms on the RHS, replace with variables:
   - A \rightarrow bAA replace with A \rightarrow BAA, B \rightarrow b

Example, Chomsky Normal Form

Replace S \rightarrow bA V \rightarrow bA by S \rightarrow BA V \rightarrow BA

Replace A \rightarrow aV by A \rightarrow QV Q \rightarrow a

4. Rewrite long rules:
   - A \rightarrow BAA to A \rightarrow BZ Z \rightarrow AA

In step 2: eliminate all unit rules. Determine for which variables you have A \rightarrow B (how?). For each rule
   - B \rightarrow u |u| > 1
   - add rule A \rightarrow u
<table>
<thead>
<tr>
<th>Why Chomsky Normal Form?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Have simple and easily bounded algorithm to determine if string is generated by CNF form</td>
</tr>
<tr>
<td>How?</td>
</tr>
</tbody>
</table>