Simplification of CFGs

The goal is to try to get a *Normal Form* of the CFG like *Chomsky Normal Form* or *Greibach Normal Form*. To get there, we need to make a number of preliminary simplifications, which are themselves useful in various ways:

1. We must eliminate *useless symbols*, those variables or terminals that do not appear in any derivation of a terminal string from the start symbol.
2. We must eliminate *ε-productions*, those of the form A → ε for some variable A.
3. We must eliminate *unit productions*, those of the form A → B for variables A and B.

Eliminating Useless Symbols

- A symbol *X* is *useful* for a grammar G = (V, T, P, S) if there is some derivation of the form $S \Rightarrow^* \alpha X \beta \Rightarrow^* w$, where *w* is in T*. Note that X may be in either V or T.
- The approach to eliminating useless symbols begins by identifying the two things a symbol has to be able to do to be useful:
  - A symbol X is *generating* if $X \Rightarrow^* w$ for some terminal string *w*. Note that every terminal is generating, since *w* can be that terminal itself, which is derived by zero steps.
  - A symbol X is *reachable* if there is a derivation $S \Rightarrow^* \alpha X \beta$ for some α and β.
- A symbol that is *useful* will be both *generating* and *reachable*.
- The non-generating symbols will be eliminated first, then the non-reachable symbols will be eliminated from the rest of the symbols. Thus, only the useful symbols will be left in the grammar.

Computing the Generating and Reachable symbols

Let G = (V, T, P, S) be a grammar. To compute the generating symbols of G, the following induction is performed.

> **Basis:** Every symbol of T is obviously generating; it generates itself.
> **Induction:** Suppose there is a production A → α, and every symbol of α is already known to be generating; then A is generating.

To compute the reachable symbols of G, the following induction is performed.

> **Basis:** S is surely reachable.
> **Induction:** Suppose we have discovered that some variable A is reachable. Then for all productions with A in the head, all the symbols of the bodies of those productions are also reachable.

Example:

Consider the grammar:

$$S \rightarrow AB \mid a$$
$$A \rightarrow b$$

Eliminating non-generating symbols: All symbols but B are generating. If we eliminate B, we must eliminate the production $S \rightarrow AB$, leaving the grammar:

$$S \rightarrow a$$
$$A \rightarrow b$$

Eliminating non-reachable symbols: Only S and a are reachable from S. Eliminating A and b leaves only the production:

$$S \rightarrow a$$

Eliminating ε Productions

- The strategy is to begin by discovering which variables are "nullable". A variable A is nullable if $A \Rightarrow^{*} \varepsilon$.
- If A is nullable, then whenever A appears in a production body, say $B \rightarrow CAD$, A might (or might not) derive ε. So, two versions of the production are possible, $B \rightarrow CD$ *or* $B \rightarrow CAD$.

Let G = (V, T, P, S) be a CFG. We can find all the nullable symbols of G by the following iterative algorithm.

Basis: If $A \rightarrow \varepsilon$ is a production of G, then A is nullable.
Induction: If there is a production $B \rightarrow C_1C_2 \ldots C_k$, where each C_i is nullable, then B is nullable. Note that each C_i must be a variable to be nullable, so we only have to consider productions with all-variable bodies.

Example:

Consider the grammar:

$$S \rightarrow AB$$
$$A \rightarrow aAA \mid \varepsilon$$
$$B \rightarrow bBB \mid \varepsilon$$

Finding nullable symbols: A, B are nullable. So, S is also nullable.

Constructing the productions of G_1:

Considering $S \rightarrow AB$:	$S \rightarrow AB \mid A \mid B$
Considering $A \rightarrow aAA$:	$A \rightarrow aAA \mid aA \mid aA \mid a$
Considering $B \rightarrow bBB$:	$B \rightarrow bBB \mid bB \mid bB \mid b$

The two ε productions of G yield nothing for G_1, thus the following productions constitute G_1.

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