Simplification of CFG

Beulah A.

AP/CSE

SSNCE

Three ways to simplify/clean a CFG

(clean)

1. Eliminate useless symbols

(simplify)

2. Eliminate ε -productions

$$A \gg \epsilon$$

3. Eliminate unit productions

$$A \Rightarrow B$$

Eliminating useless symbols

A symbol X is <u>reachable</u> if there exists:

•
$$S \Rightarrow * \alpha X \beta$$

A symbol X is generating if there exists:

•
$$X \Rightarrow^* w$$
,
for some $w \in T^*$

For a symbol X to be "useful", it has to be both reachable and generating

•
$$S \Rightarrow^* \alpha X \beta \Rightarrow^* w'$$
, for some $w' \in T^*$

Omitting useless symbols obviously will not change the language generated by the grammar.

Algorithm to detect useless symbols

- 1. First, eliminate all symbols that are *not* generating
- 2. Next, eliminate all symbols that are *not* reachable

Is the order of these steps important, or can we switch?

Example: Useless symbols

- S→AB | a
- A → b
- 1. A, S are generating
- 2. *B* is *not generating* (and therefore B is useless)
- 3. Eliminating B... (i.e., remove all productions that involve B)
 - 1. $S \rightarrow a$
 - 2. $A \rightarrow b$
- 4. Now, A is *not reachable* and therefore is useless
- 5. Simplified G:
 - 1. $S \rightarrow a$

What would happen if you reverse the order: i.e., test reachability before generating?

Will fail to remove:

 $A \rightarrow b$

Algorithm to detect useless symbols

$$S \rightarrow aSb \mid A \mid \epsilon$$

 $A \rightarrow aA$

$$S \rightarrow aSb \mid \epsilon$$

Algorithm to find all generating symbols

 $X \rightarrow^* w$

- Given: G=(V,T,P,S)
- Basis:
 - Every symbol in T is obviously generating.
- Induction:
 - Suppose for a production A $\rightarrow \alpha$, where α is generating
 - Then, A is also generating

Algorithm to find all reachable symbols $S \rightarrow^* \alpha X \beta$

- Given: G=(V,T,P,S)
- Basis:
 - S is obviously reachable (from itself)
- Induction:
 - Suppose for a production A $\rightarrow \alpha_1 \alpha_2 ... \alpha_k$, where A is reachable
 - Then, all symbols on the right hand side, $\{\alpha_1, \alpha_2, \dots \alpha_k\}$ are also reachable.

Eliminating &-productions



It is *not* possible to eliminate **\varepsilon**-productions for languages which include **\varepsilon** in their word set

So we will target the grammar for the <u>rest</u> of the language

Theorem: If G=(V,T,P,S) is a CFG for a language L, then L- $\{\varepsilon\}$ has a CFG without ε -productions

<u>Definition:</u> A is "nullable" if $A \rightarrow * \mathcal{E}$

- If A is nullable, then any production of the form "B → CAD" can be simulated by:
 - \bullet B \rightarrow CD | CAD
 - This can allow us to remove ε transitions for A

Algorithm to detect all nullable variables

Basis:

• If $A \rightarrow \epsilon$ is a production in G, then A is nullable (note: A can still have other productions)

• Induction:

• If there is a production $B \rightarrow C_1C_2...C_k$, where *every* C_i is nullable, then B is also nullable

Eliminating \(\epsilon\)-productions

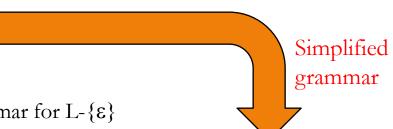
Given: G=(V,T,P,S)

Algorithm:

- 1. Detect all nullable variables in G
- 2. Then construct $G_1 = (V,T,P_1,S)$ as follows:
 - i. For each production of the form: $A \rightarrow X_1 X_2 ... X_k$, where $k \ge 1$, suppose m out of the $k \ge 1$, are nullable symbols
 - ii. Then G_1 will have 2^m versions for this production
 - i.e., all combinations where each X_i is either present or absent
 - iii. Alternatively, if a production is of the form: A $\rightarrow \epsilon$, then remove it

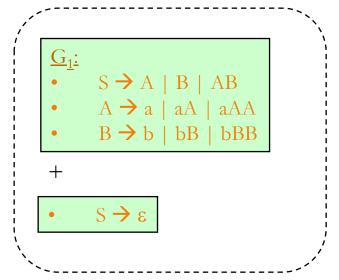
Example: Eliminating ε-productions

- Let L be the language represented by the following CFG G:
 - i. $S \rightarrow AB$
 - ii. $A \rightarrow aAA \mid \varepsilon$
 - iii. B \rightarrow bBB | ϵ



<u>Goal</u>: To construct G1, which is the grammar for L- $\{\epsilon\}$

- Nullable symbols: {A, B}
- G_1 can be constructed from G as follows:
 - \bullet B \rightarrow b | bB | bB | bBB
 - $==> B \rightarrow b \mid bB \mid bBB$
 - Similarly, $A \rightarrow a \mid aA \mid aAA$
 - Similarly, $S \rightarrow A \mid B \mid AB$
- Note: $L(G) = L(G_1) U \{ \epsilon \}$



Summary

- Discussion about context free grammar
- Language of CFG
- Derivations from a grammar for a string/word
- Parse tree for a string/word
- Ambiguous grammar

Test Your Knowledge

- Suppose A \rightarrow xBz and B \rightarrow y, then the simplified grammar would be:
 - a) $A \rightarrow xyz$
 - b) $A \rightarrow xBz | xyz$
 - c) $A \rightarrow xBz|B|y$
 - d) none of the mentioned
- Given Grammar: $S \rightarrow A$, $A \rightarrow aA$, $A \rightarrow e$, $B \rightarrow bA$ Which among the following productions are Useless productions?
 - a) $S \rightarrow A$
 - b) $A \rightarrow aA$
 - c) $A \rightarrow e$
 - d) $B \rightarrow bA$

Reference

 Hopcroft J.E., Motwani R. and Ullman J.D,
 "Introduction to Automata Theory, Languages and Computations", Second Edition, Pearson Education, 2008