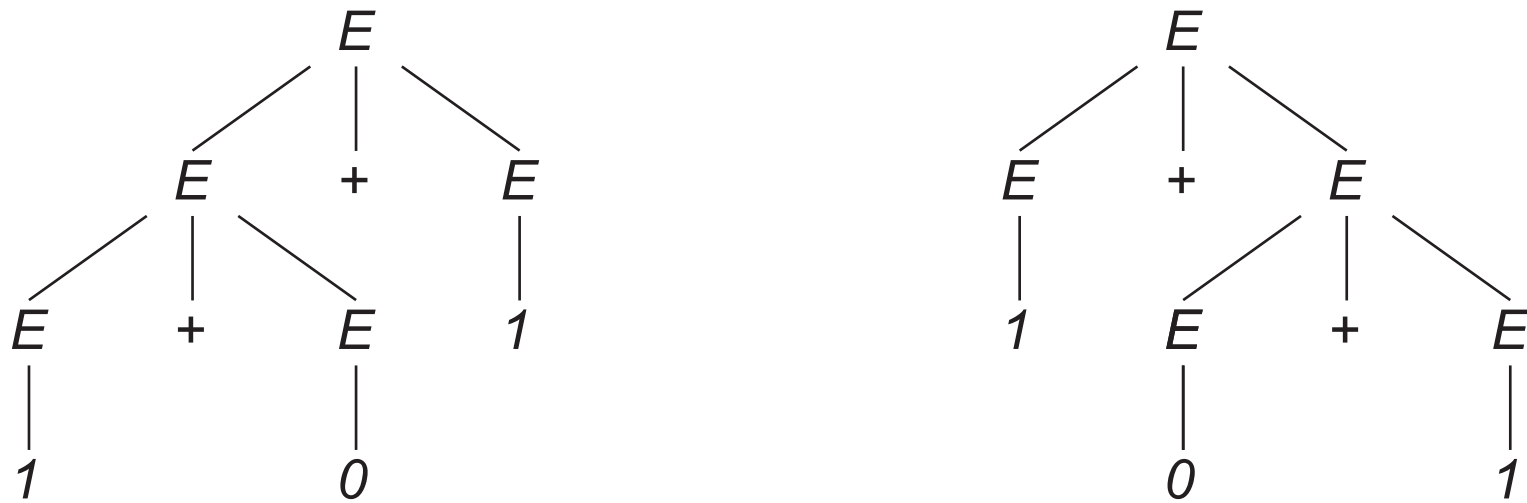


Even though every derivation tree defines a unique leftmost derivation, some words may have several different leftmost (or rightmost) derivations. This happens if a word is the yield of several different derivation trees.

Example. In our sample grammar word $1 + 0 + 1$ is the yield of two different derivation trees



corresponding to two different leftmost derivations

$$\begin{aligned}
 E &\Rightarrow E + E \Rightarrow E + E + E \Rightarrow 1 + E + E \Rightarrow 1 + 0 + E \Rightarrow 1 + 0 + 1, \text{ and} \\
 E &\Rightarrow E + E \Rightarrow 1 + E \Rightarrow 1 + E + E \Rightarrow 1 + 0 + E \Rightarrow 1 + 0 + 1,
 \end{aligned}$$

A context-free grammar G is called **ambiguous** if some word has more than one derivation tree.

Equivalently: G is ambiguous if some word has more than one leftmost derivation (or rightmost derivation).

A context-free grammar G is called **ambiguous** if some word has more than one derivation tree.

Equivalently: G is ambiguous if some word has more than one leftmost derivation (or rightmost derivation).

One can prove ambiguity by finding two leftmost derivations for some word.

Proving unambiguity is harder:

- There does not exist an algorithm that would determine if a given context-free grammar G is ambiguous or unambiguous. It is an **undecidable** property. (We are going to prove this before the end of the semester.)
- As a consequence, there exist unambiguous grammars whose unambiguity cannot be proved.

For individual grammars one can come up with ad hoc proofs.

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSSb \mid ab.$$

Is this grammar ambiguous or unambiguous ?

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSSb \mid ab.$$

Is this grammar ambiguous or unambiguous ?

The grammar is unambiguous. (Proof on the blackboard)

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ where P contains productions

$$S \longrightarrow aSb \mid aaS \mid \varepsilon.$$

Is G ambiguous or unambiguous ?

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ where P contains productions

$$S \longrightarrow aSb \mid aaS \mid \varepsilon.$$

Is G ambiguous or unambiguous ?

Ambiguous: word aab has two different leftmost derivations

$$S \Rightarrow aaS \Rightarrow aaaSb \Rightarrow aaab$$

$$S \Rightarrow aSb \Rightarrow aaaSb \Rightarrow aaab.$$

For some context-free languages there exist only ambiguous grammars. Such languages are called **inherently ambiguous**.

(Note that inherent ambiguity is a property of a language while ambiguity is a property of a grammar.)

Also inherent ambiguity is **undecidable**. In other words, there does not exist an algorithm for determining if the language $L(G)$ generated by given context-free grammar G is inherently ambiguous or not.

An example of an inherently ambiguous context-free language:

$$\{a^n b^n c^m d^m \mid n, m \geq 1\} \cup \{a^n b^m c^m d^n \mid n, m \geq 1\}$$

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ where P contains productions

$$S \longrightarrow aSb \mid aaS \mid \varepsilon.$$

We saw that this grammar is ambiguous because the word $aaab$ has two leftmost derivations.

But is the language $L(G)$ it generates inherently ambiguous ?

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ where P contains productions

$$S \longrightarrow aSb \mid aaS \mid \varepsilon.$$

We saw that this grammar is ambiguous because the word $aaab$ has two leftmost derivations.

But is the language $L(G)$ it generates inherently ambiguous ?

No: the same language is generated by the unambiguous grammar $(\{S, A\}, \{a, b\}, P', S)$ with productions

$$\begin{aligned} S &\longrightarrow aSb \mid A, \\ A &\longrightarrow aaA \mid \varepsilon. \end{aligned}$$

(Using variable A we force the grammar first to use productions $S \longrightarrow aSb$ before productions $S \longrightarrow aaS$.)

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSb \mid bS \mid \varepsilon.$$

Ambiguous or unambiguous ?

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSb \mid bS \mid \varepsilon.$$

Ambiguous or unambiguous ?

Unambiguous.

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSbS \mid bSaS \mid \varepsilon.$$

Ambiguous or unambiguous ?

Ambiguous because *abab* has two different leftmost derivations.

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSbS \mid bSaS \mid \varepsilon.$$

Ambiguous or unambiguous ?

Ambiguous because *abab* has two different leftmost derivations.

What about the corresponding language

$$L(G) = \{w \in \{a, b\}^* \mid w \text{ contains equally many } a\text{'s and } b\text{'s} \}.$$

Is it inherently ambiguous or not ?

Example. Grammar $G = (\{S\}, \{a, b\}, P, S)$ with productions

$$S \longrightarrow aSbS \mid bSaS \mid \varepsilon.$$

Ambiguous or unambiguous ?

Ambiguous because *abab* has two different leftmost derivations.

What about the corresponding language

$$L(G) = \{w \in \{a, b\}^* \mid w \text{ contains equally many } a\text{'s and } b\text{'s} \}.$$

Is it inherently ambiguous or not ?

$L(G)$ is **not inherently ambiguous**. The same language is generated by the unambiguous grammar $(\{S, P, N\}, \{a, b\}, P, S)$ with productions

$$\begin{aligned} S &\longrightarrow aPbS \mid bNaS \mid \varepsilon \\ P &\longrightarrow aPbP \mid \varepsilon \\ N &\longrightarrow bNaN \mid \varepsilon \end{aligned}$$

(Proof in the homework.)

Grammars G and G' are called **equivalent** if they generate the same language $L(G) = L(G')$. Certain types of productions are undesirable in a grammar, and we would like to find an equivalent grammar that does not contain such productions.

We want to remove productions of form

$$\begin{array}{l} A \longrightarrow \varepsilon \quad (\text{\textcolor{blue}{\varepsilon-production}}), \text{ and} \\ A \longrightarrow B \quad (\text{\textcolor{blue}{unit production}}). \end{array}$$

We also want to remove symbols that are unnecessary in the sense that they are not used in any terminating derivations.

The simplification is done in the following order:

1. Remove ε -productions.
2. Remove unit productions.
3. Remove variables that do not derive any terminal strings.
4. Remove symbols that cannot be reached from the start symbol.

STEP 1. Remove ε -productions.

Given a CFG G we (effectively) construct a CFG G' that does not have any ε -productions and

$$L(G') = L(G) - \{\varepsilon\}$$

(So ε is lost from the generated language. This cannot be avoided because a grammar without ε -productions cannot generate the word ε .)

The idea of G' is to remove from derivation trees all subtrees whose yield is ε .

STEP 1. Remove ε -productions.

Given a CFG G we (effectively) construct a CFG G' that does not have any ε -productions and

$$L(G') = L(G) - \{\varepsilon\}$$

(So ε is lost from the generated language. This cannot be avoided because a grammar without ε -productions cannot generate the word ε .)

The idea of G' is to remove from derivation trees all subtrees whose yield is ε .

Let us call a variable A **nullable** if $A \Rightarrow^* \varepsilon$.

We can identify the nullable variable using a simple marking procedure:

- First, mark as nullable those variables A that have a production $A \longrightarrow \varepsilon$.
- Then keep marking variables A that have productions

$$A \longrightarrow \alpha$$

where α is a word of variables that have all been already marked nullable. Repeat this until no new variables can be marked.

Then we construct a new set of productions as follows: For every production of the original grammar

$$A \longrightarrow X_1 X_2 \dots X_n$$

where all $X_i \in V \cup T$, the new production set will have all productions

$$A \longrightarrow \alpha_1 \alpha_2 \dots \alpha_n$$

where

- $\alpha_i = X_i$ if X_i is not nullable, or is a terminal symbol,
- $\alpha_i = \varepsilon$ or $\alpha_i = X_i$ if X_i is nullable,
- $\alpha_1 \alpha_2 \dots \alpha_n \neq \varepsilon$.

Remark: If several X_i are nullable, all combinations of $\alpha_i = \varepsilon$ and X_i are used. One original production can result in up to 2^k new productions if the right-hand-side contains k nullable variables.

Then we construct a new set of productions as follows: For every production of the original grammar

$$A \longrightarrow X_1 X_2 \dots X_n$$

where all $X_i \in V \cup T$, the new production set will have all productions

$$A \longrightarrow \alpha_1 \alpha_2 \dots \alpha_n$$

where

- $\alpha_i = X_i$ if X_i is not nullable, or is a terminal symbol,
- $\alpha_i = \varepsilon$ or $\alpha_i = X_i$ if X_i is nullable,
- $\alpha_1 \alpha_2 \dots \alpha_n \neq \varepsilon$.

$L(G') \subseteq L(G)$: Each production $A \longrightarrow \alpha$ in G' is obtained from a production in G by removing nullable variables. Thus in G there is a derivation $A \Rightarrow^* \alpha$.

Any derivation in G' can hence be turned into a derivation in G by replacing each derivation step $xAy \Rightarrow x\alpha y$ by a derivation $xAy \Rightarrow^* x\alpha y$.

Then we construct a new set of productions as follows: For every production of the original grammar

$$A \longrightarrow X_1 X_2 \dots X_n$$

where all $X_i \in V \cup T$, the new production set will have all productions

$$A \longrightarrow \alpha_1 \alpha_2 \dots \alpha_n$$

where

- $\alpha_i = X_i$ if X_i is not nullable, or is a terminal symbol,
- $\alpha_i = \varepsilon$ or $\alpha_i = X_i$ if X_i is nullable,
- $\alpha_1 \alpha_2 \dots \alpha_n \neq \varepsilon$.

$L(G') \subseteq L(G)$: Each production $A \longrightarrow \alpha$ in G' is obtained from a production in G by removing nullable variables. Thus in G there is a derivation $A \Rightarrow^* \alpha$.

Any derivation in G' can hence be turned into a derivation in G by replacing each derivation step $xAy \Rightarrow x\alpha y$ by a derivation $xAy \Rightarrow^* x\alpha y$.

$L(G) - \{\varepsilon\} \subseteq L(G')$: Let $w \in L(G) - \{\varepsilon\}$. It has a derivation tree T . Removing from T all subtrees whose yield is ε results in a derivation tree of G' with the same yield w .

STEP 2. Remove unit productions.

Assume we have removed all ε -productions from P . The right-hand-side of every production in P is some non-empty word.

Let us find an equivalent grammar that does not contain ε -productions or unit productions (=productions $A \longrightarrow B$ for variables A, B).

The idea is similar to step 1: We anticipate all possible unit derivations, and plug them in directly.

First, for every variable A we find all variables B , such that

$$A \Rightarrow^* B.$$

(In this derivation only unit productions can be used.)

Such variable-to-variable derivations $A \Rightarrow^* B$ for any fixed A can be effectively found using a marking procedure:

- First, mark variable A .
- Then keep marking variables B such that there is a unit production

$$X \longrightarrow B$$

where X is already marked. Repeat this until no new variables can be marked.

A non-mandatory step: At this stage, if one wants, one may simplify the grammar by removing duplicate variables.

If A and B are different variables such that $A \Rightarrow^* B$ and $B \Rightarrow^* A$, it is clear that both variables derive exactly same sentential forms. So we may trim the grammar by replacing B by A everywhere, both left- and righthand sides of all productions.

Also, productions $A \longrightarrow A$ can be directly removed as useless.

Then we construct a new set of productions as follows: For every **non-unit production**

$$B \longrightarrow \alpha$$

of the grammar and for every variable A such that $A \Rightarrow^* B$ we take in the new grammar the production

$$A \longrightarrow \alpha.$$

The new grammar G' then has no unit productions.

Then we construct a new set of productions as follows: For every **non-unit production**

$$B \longrightarrow \alpha$$

of the grammar and for every variable A such that $A \Rightarrow^* B$ we take in the new grammar the production

$$A \longrightarrow \alpha.$$

The new grammar G' then has no unit productions.

$L(G') \subseteq L(G)$: For each production $A \longrightarrow \alpha$ in G' there is a derivation $A \Rightarrow^* B \Rightarrow \alpha$ in G . Any derivation in G' can hence be turned into a derivation in G by replacing each derivation step $xAy \Rightarrow x\alpha y$ by a derivation $xAy \Rightarrow^* x\alpha y$.

Then we construct a new set of productions as follows: For every **non-unit production**

$$B \longrightarrow \alpha$$

of the grammar and for every variable A such that $A \Rightarrow^* B$ we take in the new grammar the production

$$A \longrightarrow \alpha.$$

The new grammar G' then has no unit productions.

$L(G') \subseteq L(G)$: For each production $A \longrightarrow \alpha$ in G' there is a derivation $A \Rightarrow^* B \Rightarrow \alpha$ in G . Any derivation in G' can hence be turned into a derivation in G by replacing each derivation step $xAy \Rightarrow x\alpha y$ by a derivation $xAy \Rightarrow^* x\alpha y$.

$L(G) \subseteq L(G')$: Let $w \in L(G)$. Consider terminal derivations $S \Rightarrow^* w$ that use productions of G and G' . If there is a derivation step $xAy \Rightarrow xBy$ that uses a unit production $A \longrightarrow B$ then the derivation can be shortened by removing xBy . The remaining shorter derivation still uses productions of G and G' . Repeat this until no applications of unit productions remain.

The final derivation then only uses productions of G' because all non-unit productions of G are also in G' .

STEP 3. Remove variables that do not derive any terminal strings.

Such variables can be simply removed together with all productions containing the variable on either side of the production. This does not effect the generated language since the variable is not used in any terminating derivation.

To find variables that **do** generate some terminal string, we apply similar marking procedure as with ε -productions:

- mark all variables A such that there is a production $A \longrightarrow w$ in P , where w contains only terminals.
- Then keep marking variables A that have productions

$$A \longrightarrow \alpha$$

where α is a word that contains only terminals and variables that have already been marked. Repeat this until no new variables can be marked.

The variables that have **not** been marked can be removed. All productions containing removable variables are unnecessary, and can be deleted.

STEP 4. Remove symbols (variables and terminals) that are not reachable from the start symbol S .

Any production that contains such an unreachable symbol can be removed.

To find symbols that **can** be reached from S are found again using a marking procedure:

- Initially, mark the start symbol S .
- Then, for all productions $A \longrightarrow \alpha$ where A is a marked variable, mark all symbols that appear in α . Repeat this until no new variables can be marked.

All symbols that have **not** been marked can not be reached from the initial symbol, and they can be removed.

Remark: After step 4 all remaining variables still derive a terminal word, so we do not need to repeat step 3 again.

(Step 3 may introduce new unreachable symbols, so if we do step 4 before step 3, we may have to do step 4 again.)