CS:4330 Theory of Computation

Spring 2018

Context-Free Languages

Context-Free Grammars

Haniel Barbosa

The University of Iowa

Readings for this lecture

Chapter 2 of [Sipser 1996], 3rd edition. Section 2.1.

Context-Free Grammars (CFG)

- ▷ There are languages, such as $\{0^n 1^n \mid n \geq 0\}$ that cannot be described by finite automata (or regexps)

- ▷ Context-free grammars provide a more powerful mechanism for language specification.

- ▷ Context-free grammars can describe features that have a recursive structure, making them useful beyond finite automata.

Formal definition of a CFG

A context-free grammar is a 4-tuple (V, Σ, R, S) in which:

- V is a finite set of symbols called the *variables* or *nonterminals*
- Σ is a finite set of symbols, disjoint from V, called *terminals*
- R is a finite set of *rules* of the form $lhs \to rhs$, in which $lhs \in V$ and $rhs \in (V \cup \Sigma)^*$
- $S \in V$ is the start nonterminal

Example

▷ CFG G_1 has the following rules:

$$\begin{aligned} A &\rightarrow 0A1 \\ A &\rightarrow B \\ B &\rightarrow \# \end{aligned}$$

▷ Nonterminals of G_1 are $\{A, B\}$ and A is the start symbol

▷ Terminals of G_1 are $\{0, 1, \#\}$

Language specification

A grammar is used for a language specification by generating each string of the language in the following manner:

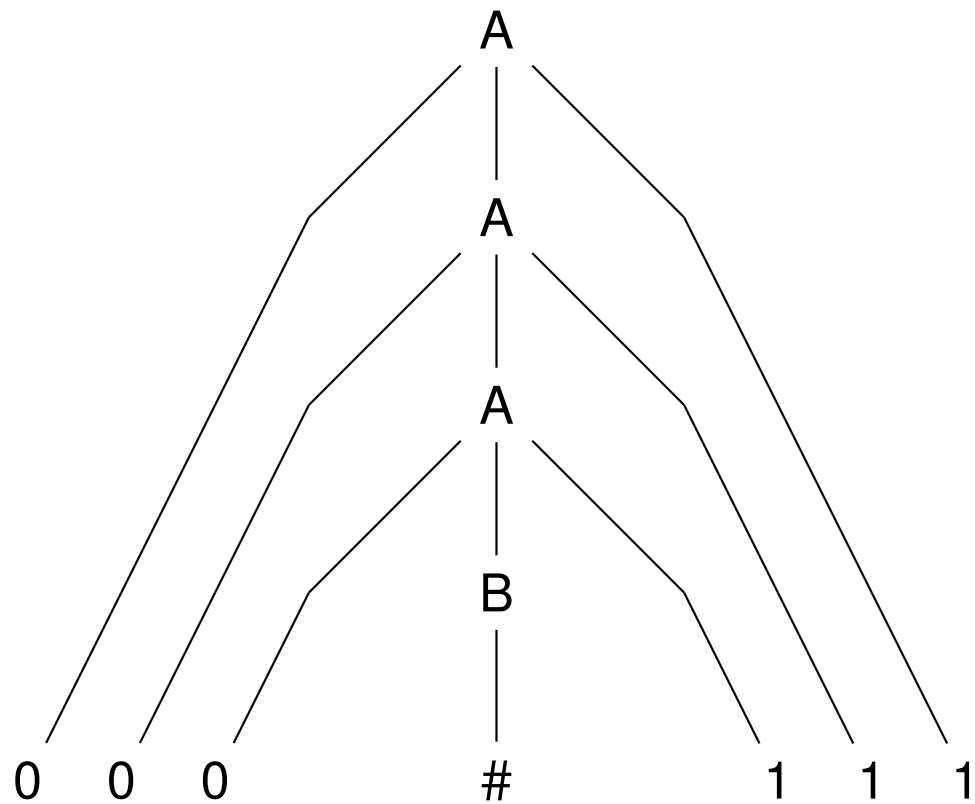
1. Write down the start variable; it is the *lhs* of the first rule, unless specified otherwise
2. Find a variable that is written down and a rule whose *lhs* is that variable. Replace the written down variable with the *rhs* of that rule.
3. Repeat step 2 until no variables remain in the string thus generated.

Note

The sequence of substitutions used to obtain a string using a CFG is called a *derivation* and may be represented by a tree called *derivation tree* or a *parse tree*

Example derivation tree

The derivation tree of the string $000\#111$ using CFG G_1 is:



Note

- ▷ All strings of terminals generated in this way constitute the language specified by the grammar

- ▷ We write $\mathscr{L}(G)$ for the language generated by the grammar G. Thus, $\mathscr{L}(G_1) = \{0^n\#1^n \mid n \geq 0\}$

- ▷ A language generated by a context-free grammar (CFG) is called a *Context-Free Language* (CFL).

CFG G_2

The CFG G_2 specifies a fragment of English:

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\langle \text{SENTENCE} \rangle \Rightarrow \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{ARTICLE}} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{ARTICLE}} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a} \langle \underline{\text{NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{ARTICLE}} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a} \langle \underline{\text{NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{VERB-PHRASE}} \rangle
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{ARTICLE}} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a} \langle \underline{\text{NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{VERB-PHRASE}} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{CP-VERB}} \rangle
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{ARTICLE}} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a} \langle \underline{\text{NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{VERB-PHRASE}} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{CP-VERB}} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{VERB}} \rangle
\end{array}
$$

Example derivation with G_2

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \rightarrow & \langle \text{NOUN-PHRASE} \rangle \langle \text{VERB-PHRASE} \rangle \\
\langle \text{NOUN-PHRASE} \rangle & \rightarrow & \langle \text{CP-NOUN} \rangle \mid \langle \text{CP-NOUN} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{VERB-PHRASE} \rangle & \rightarrow & \langle \text{CP-VERB} \rangle \mid \langle \text{CP-VERB} \rangle \langle \text{PREP-PHRASE} \rangle \\
\langle \text{PREP-PHRASE} \rangle & \rightarrow & \langle \text{PREP} \rangle \langle \text{CP-NOUN} \rangle \\
\langle \text{CP-NOUN} \rangle & \rightarrow & \langle \text{ARTICLE} \rangle \langle \text{NOUN} \rangle \\
\langle \text{CP-VERB} \rangle & \rightarrow & \langle \text{VERB} \rangle \mid \langle \text{VERB} \rangle \langle \text{NOUN-PHRASE} \rangle \\
\langle \text{ARTICLE} \rangle & \rightarrow & \text{a} \mid \text{the} \\
\langle \text{NOUN} \rangle & \rightarrow & \text{boy} \mid \text{girl} \mid \text{flower} \\
\langle \text{VERB} \rangle & \rightarrow & \text{touches} \mid \text{likes} \mid \text{sees} \\
\langle \text{PREP} \rangle & \rightarrow & \text{with}
\end{array}
$$

$$
\begin{array}{lcl}
\langle \text{SENTENCE} \rangle & \Rightarrow & \langle \underline{\text{NOUN-PHRASE}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{CP-NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \langle \underline{\text{ARTICLE}} \rangle \langle \text{NOUN} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a} \langle \underline{\text{NOUN}} \rangle \langle \text{VERB-PHRASE} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{VERB-PHRASE}} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{CP-VERB}} \rangle \\
& \Rightarrow & \text{a boy} \langle \underline{\text{VERB}} \rangle \\
& \Rightarrow & \text{a boy sees}
\end{array}
$$

Direct derivation

- If $u, v, w \in (V \cup \Sigma)^*$ (i.e. are strings of variables and terminals) and $A \to w \in R$ (i.e. is a rule of the grammar), then we say that uAv yields uwv, written

$$uAv \Rightarrow uwv$$

- We may also say that uwv is directly derived from uAv using the rule $A \to w$

Derivation

- We write $u \overset{*}{\Rightarrow} v$ if $u = v$ or if a sequence $u_1, \ldots, u_k \in (V \cup \Sigma)^*$ exists, for $k \geq 0$, and $u \Rightarrow u_1 \Rightarrow \cdots \Rightarrow u_k \Rightarrow v$

- We may also say that $u, u_1, \ldots, u_k, v$ is a derivation of v from u

Language specified by G

If $G = (V, \Sigma, R, S)$ is a CFG then the language specified by G (or the language of G) is a CFL

$$\mathscr{L}(G) = \{w \in \Sigma^* \mid S \stackrel{*}{\Rightarrow} w\}$$

More examples of CFGs

- Consider the grammar

$$G_3 = (\{S\}, \{\mathrm{a}, \mathrm{b}\}, \{S \to \mathrm{a}S\mathrm{b} \mid SS \mid \epsilon\}, S)$$

More examples of CFGs

- Consider the grammar

$$G_3 = (\{S\}, \{\mathrm{a}, \mathrm{b}\}, \{S \to \mathrm{a}S\mathrm{b} \mid SS \mid \epsilon\}, S)$$

- $\mathscr{L}(G_3)$ contains strings such as

$$\mathrm{abab}, \quad \mathrm{aaabbb}, \quad \mathrm{aababb}$$

Note

If one thinks of a and b as the symbols '(' and ')' then we can see that $\mathscr{L}(G_3)$ is the language of all strings of properly nested parenthesis

Important application

Context-free grammars are used as basis for compiler design and implementation

- ▷ Context-free grammars are used as specification mechanisms for programming languages

- ▷ Designers of compilers use such grammars to implement compiler's components, such as scanners, parsers, code generators, *code synthesizers*

- ▷ The implementation of almost any programming languages is preceded by a context-free grammar that specifies it

Example

▷ Consider the grammar $G_4 = (\{E, T, F\}, \{\mathrm{a}, +, *, (,)\}, R, E)$ in which R is:

$$\begin{array}{lcl} E & \rightarrow & E + T \mid T \\ T & \rightarrow & T * F \mid F \\ F & \rightarrow & (E) \mid \mathrm{a} \end{array}$$

Example

▷ Consider the grammar $G_4 = (\{E, T, F\}, \{\mathrm{a}, +, *, (,)\}, R, E)$ in which R is:

$$\begin{array}{lcl} E & \rightarrow & E + T \mid T \\ T & \rightarrow & T * F \mid F \\ F & \rightarrow & (E) \mid \mathrm{a} \end{array}$$

▷ $\mathscr{L}(G_4)$ is the language of arithmetic expressions

Application: program synthesis

Consider the grammar $G_S = (\{E, B\}, \{0, 1, x, y, +, \text{ite}, \leq, \wedge, \neg, \ , \ , (,)\}, R, E)$ in which R is:

$$\begin{array}{lcl} E & \rightarrow & 0 \mid 1 \mid x \mid y \mid (E + E) \mid \text{ite}(B, E, E) \\ B & \rightarrow & (\neg B) \mid (B \wedge B) \mid (E \leq E) \end{array}$$

What is a program generated with this grammar that solves the following problem:

$$prog(x, y) \geq x \wedge prog(x, y) \geq y$$

Application: program synthesis

Consider the grammar $G_S = (\{E, B\}, \{0, 1, x, y, +, \text{ite}, \leq, \wedge, \neg, \ , \ , (,)\}, R, E)$ in which R is:

$$\begin{array}{lcl} E & \rightarrow & 0 \mid 1 \mid x \mid y \mid (E + E) \mid \text{ite}(B, E, E) \\ B & \rightarrow & (\neg B) \mid (B \wedge B) \mid (E \leq E) \end{array}$$

What is a program generated with this grammar that solves the following problem:

$$prog(x, y) \geq x \wedge prog(x, y) \geq y$$

A solution is $prog(x, y) = \text{ite}(x \leq y, y, x)$, i.e. the max function.

Designing CFGs

- As with the design of automata, the design of CFGs requires creativity

- CFGs are even trickier to construct than finite automata since “we are more accustomed to programming a machine than to specify programming languages.”

Design Techniques

- ▷ Many CFGs are unions of simpler CFGs. Hence the suggestion is to construct smaller, simpler grammars first and then to join them into larger grammar

- ▷ The mechanism of grammar combination consists of putting all their rules together and adding the new rules

$$S \to S_1 \mid \cdots \mid S_k$$

where the nonterminals S_i, for $1 \leq i \leq k$, are the start variables of the individual grammars and S is the new variable

Example Grammar Design

Design a grammar for the language $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

Example Grammar Design

Design a grammar for the language $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

1. Construct the grammar $S_1 \to 0S_1 1 \mid \epsilon$ for the language

$$\{0^n 1^n \mid n \geq 0\}$$

Example Grammar Design

Design a grammar for the language $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

1. Construct the grammar $S_1 \to 0S_1 1 \mid \epsilon$ for the language

$$\{0^n 1^n \mid n \geq 0\}$$

   Construct the grammar $S_2 \to 1S_1 0 \mid \epsilon$ for the language

$$\{1^n 0^n \mid n \geq 0\}$$

Example Grammar Design

Design a grammar for the language $\{0^n 1^n \mid n \geq 0\} \cup \{1^n 0^n \mid n \geq 0\}$

1. Construct the grammar $S_1 \to 0S_1 1 \mid \epsilon$ for the language

$$\{0^n 1^n \mid n \geq 0\}$$

Construct the grammar $S_2 \to 1S_1 0 \mid \epsilon$ for the language

$$\{1^n 0^n \mid n \geq 0\}$$

3. Put them together adding the rule $S \to S_1 \mid S_2$, obtaining

$$\begin{array}{lcl} S & \to & S_1 \mid S_2 \\ S_1 & \to & 0S_1 1 \mid \epsilon \\ S_2 & \to & 1S_2 0 \mid \epsilon \end{array}$$

Second Design Technique

- ▷ Constructing a CFG for a regular language is easy if one can first construct a DFA for the language

- ▷ Conversion procedure:
  1. Make a variable R_1 for each state q_i of the DFA
  2. Add rules $R_i \to \mathrm{a}R_j$ to the CFG if $\delta(q_i, \mathrm{a}) = q_j$ is a transition in the DFA
  3. Add the rule $R_i \to \epsilon$ if q_i is an accept state of the DGA
  4. If q_0 is the start state of the DGA make R_0 the start variable of the CFG.

Theorem

Every regular language is context-free.

Third Design Technique

- ▷ Certain CFLs contain strings with two related substrings such as 0^n and 1^n in $\{0^n 1^n \mid n \geq 0\}$
- ▷ Example of relationship: to recognize such a language a machine would need to remember an unbounded amount of information about one of the substrings
- ▷ A CFG that handles this situation uses a rule of the form $R \rightarrow uRv$ which generates strings wherein the portion containing u's corresponds to the portion containing v's.

Example Application

Consider the CFG $G = (\{S, B\}, \{\mathrm{a}, \mathrm{b}\}, \{S \to \mathrm{a}S\mathrm{b} \mid B \mid \epsilon, B \to \mathrm{b}B \mid \mathrm{b}\}, S)$

- ▷ The following are derivations with G:

$$
\begin{array}{llllllll}
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}S\mathrm{b}BB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB\mathrm{b}B, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}BB
\end{array}
$$

  which shows that derivations in this grammar can be quite complex

- ▷ When rewriting the strings $\mathrm{aa}SBB$ we can consider further derivations of each of its symbols in isolation

Example Application

Consider the CFG $G = (\{S, B\}, \{\mathrm{a}, \mathrm{b}\}, \{S \to \mathrm{a}S\mathrm{b} \mid B \mid \epsilon, B \to \mathrm{b}B \mid \mathrm{b}\}, S)$

- ▷ The following are derivations with G:

$$
\begin{array}{lllllll}
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}S\mathrm{b}BB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB\mathrm{b}B, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}BB
\end{array}
$$

  which shows that derivations in this grammar can be quite complex

- ▷ When rewriting the strings $\mathrm{aa}SBB$ we can consider further derivations of each of its symbols in isolation

- ▷ Derivations from B are $B \Rightarrow \mathrm{b}B \Rightarrow \mathrm{bb}B \overset{*}{\Rightarrow} \mathrm{b}^{k-1}B \Rightarrow \mathrm{b}^k$, $k \geq 1$

Example Application

Consider the CFG $G = (\{S, B\}, \{\mathrm{a}, \mathrm{b}\}, \{S \to \mathrm{a}S\mathrm{b} \mid B \mid \epsilon, B \to \mathrm{b}B \mid \mathrm{b}\}, S)$

- ▷ The following are derivations with G:

$$
\begin{array}{lllllll}
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}S\mathrm{b}BB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB\mathrm{b}B, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}BB
\end{array}
$$

  which shows that derivations in this grammar can be quite complex

- ▷ When rewriting the strings $\mathrm{aa}SBB$ we can consider further derivations of each of its symbols in isolation
- ▷ Derivations from B are $B \Rightarrow \mathrm{b}B \Rightarrow \mathrm{bb}B \overset{*}{\Rightarrow} \mathrm{b}^{k-1}B \Rightarrow \mathrm{b}^k$, $k \geq 1$
- ▷ Therefore $S \Rightarrow \mathrm{a}SB \overset{*}{\Rightarrow} \mathrm{a}S\mathrm{b}^k$, $k \geq 1$

Example Application

Consider the CFG $G = (\{S, B\}, \{\mathrm{a}, \mathrm{b}\}, \{S \to \mathrm{a}S\mathrm{b} \mid B \mid \epsilon, B \to \mathrm{b}B \mid \mathrm{b}\}, S)$

- ▷ The following are derivations with G:

$$
\begin{array}{lllllll}
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}S\mathrm{b}BB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB\mathrm{b}B, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}SB, \\
S & \Rightarrow & \mathrm{a}S\mathrm{b} & \Rightarrow & \mathrm{aa}SBB & \Rightarrow & \mathrm{aa}BB
\end{array}
$$

  which shows that derivations in this grammar can be quite complex

- ▷ When rewriting the strings $\mathrm{aa}SBB$ we can consider further derivations of each of its symbols in isolation
- ▷ Derivations from B are $B \Rightarrow \mathrm{b}B \Rightarrow \mathrm{bb}B \overset{*}{\Rightarrow} \mathrm{b}^{k-1}B \Rightarrow \mathrm{b}^k$, $k \geq 1$
- ▷ Therefore $S \Rightarrow \mathrm{a}SB \overset{*}{\Rightarrow} \mathrm{a}S\mathrm{b}^k$, $k \geq 1$
- ▷ Hence, $\mathscr{L}(G) = \{\mathrm{a}^n\mathrm{b}^m \mid n \leq m\}$

Ambiguity

- If a CFG generates the same string in several different ways, we say that the string is derived *ambiguously* in that grammar

- If a CFG generates some string we say that the grammar is *ambiguous*

Example

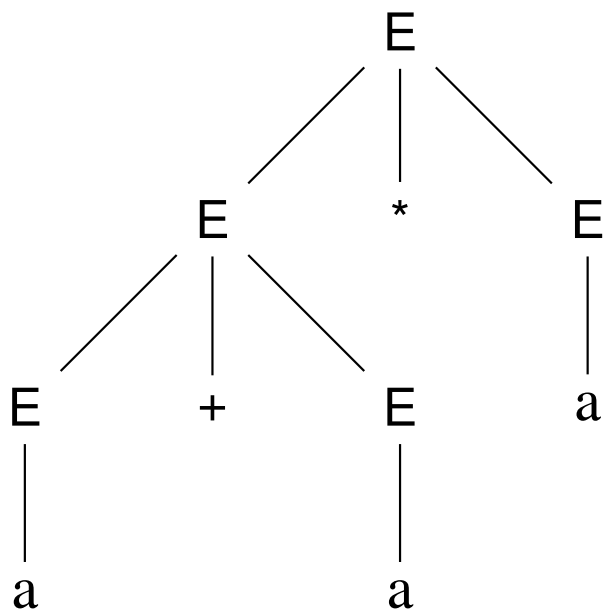
The grammar G_5, whose rules are

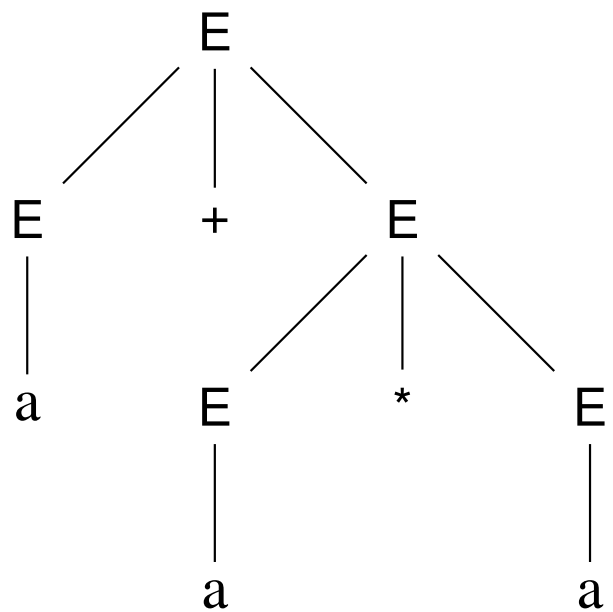
$$E \to E + E \mid E * E \mid (E) \mid \mathrm{a}$$

generates ambiguously some arithmetic expressions

Ambiguous expressions

Two different derivation trees for $\mathrm{a} + \mathrm{a} * \mathrm{a}$





Note

- The grammar G_5 does not capture the casual precedence relations and so groups the $+$ before $*$ and vice versa

- In contrast, the grammar G_4 generates the same language, but every generated string has a unique derivation tree
  $G_4 = (\{E, T, F\}, \{\mathrm{a}, +, *, (,)\}, R, E)$ in which R is:

$$\begin{array}{lcl} E & \rightarrow & E + T \mid T \\ T & \rightarrow & T * F \mid F \\ F & \rightarrow & (E) \mid \mathrm{a} \end{array}$$

- Hence, G_5 is ambiguous and G_4 is not, i.e. G_4 is *unambiguous*

Note

- When a grammar generates a string ambiguously it means that the string has two different derivation trees
- However, two different derivations may produce the same derivation tree because they may differ in the order in which they replace nonterminals, not in the rules they use
- To concentrate on the structure of the derivations we need to fix the order of rule application

Fixing rule application order

Definition (Leftmost derivation)

A derivation of a string w in a grammar G is a *leftmost derivation* if at every step the leftmost nonterminal is replaced.

Definition (Rightmost derivation)

A derivation of a string w in a grammar G is a *rightmost derivation* if at every step the rightmost nonterminal is replaced.

Inherent Ambiguity

- ▷ Some CFLs can have both ambiguous and unambiguous grammars.
- ▷ Some CFLs, however, can be generated only by an ambiguous grammar.
- ▷ A CFL that can be generated only by ambiguous grammars is called *inherently ambiguous*.

Example of inherently ambiguous language

$$\{0^i 1^j 2^k \mid i = j \vee j = k\}$$

Chomsky Normal Form

- ▷ It is often convenient to simplify CFGs so we can reason about them
- ▷ One of the simplest and most useful simplified forms of CFGs is called the Chomsky Normal Form
- ▷ Another normal form usually used in algebraic specifications is Greibach normal form

Definition

A context-free grammar G is in Chomsky normal form if every rule is of the form

$$\begin{aligned} A &\rightarrow BC \\ A &\rightarrow \mathrm{a} \end{aligned}$$

where a is a terminal, A, B, C are nonterminals, and B, C may not be the start variable

Note

The rule $S \rightarrow \epsilon$, where S is the start variable, is not excluded

Chomsky Normal Form characterizes CFLs

Theorem

Any context-free language is generated by a context-free grammar in Chomsky normal form

Proof ideas

- ▷ Show that any grammar G can be converted into Chomsky normal form
- ▷ Conversion procedure has several states where the rules that violate Chomsky normal form conditions are replaced with equivalent ones that satisfy these conditions
- ▷ Order of transformations:
  1. add a new start variable
  2. eliminate all ϵ-rules
  3. eliminate unit rules
  4. convert rules
- ▷ Check that the obtained grammar defines the same language as the initial one.

Conversion: 1 - introduce new start state

Add a new start symbol S_0 and the rule $S_0 \to S$ where S was the original start symbol

Note

This change guarantees that the start symbol does not occur on the *rhs* of any rule

Conversion: 2 - eliminate ϵ-rules

Repeat

1. Eliminate the ϵ rule $A \to \epsilon$ where A is not the start symbol
2. For each occurrence of A on the *rhs* of a rule, add a new rule with that occurrence of A deleted
   Example: To delete $A \to \epsilon$, replace $B \to uAv$ by $B \to uAv \mid uv$; replace $R \to uAvAw$ by $B \to uAvAw \mid uvAw \mid uAvw \mid uwv$
3. Replace the rule $B \to A$, (if it is present) by $B \to A \mid \epsilon$ unless the rule $B \to \epsilon$ has not been previously eliminated

until all ϵ rules are eliminated.

Conversion: 3 - remove unit rules

Repeat

1. Remove a unit rule $A \to B$
2. For each rule $B \to u$ that appears, add the rule $A \to u$, unless it was a unit rule previously removed

until all unit rules are eliminated.

Note

u is a string of variables and terminals

Conversion: 4 - Convert all remaining rules

Repeat

1. Replace a rule $A \to u_1, \ldots, u_k$, $k \geq 3$, where each u_i, $1 \leq i \leq k$, is a variable or terminal, by:

$$A \to u_1 A_1, A_1 \to u_2 A_2, \ldots A_{k-2} \to u_{k-1} u_k$$

   where $A_1, \ldots, A_{k-2}$ are new variables

2. If $k \geq 2$ replace any terminal u_i with a new variable U_i and add the rule $U_i \to u_i$

until no rules of the form $A \to u_1, \ldots, u_k$ with $k \geq 3$ remain.

Example CFG conversion

Consider the grammar

$$\begin{aligned} S &\rightarrow ASA \mid \mathrm{a}B \\ A &\rightarrow B \mid S \\ B &\rightarrow \mathrm{b} \mid \epsilon \end{aligned}$$

After the first step of transformation we get

$$\begin{aligned} S_0 &\rightarrow S \\ S &\rightarrow ASA \mid \mathrm{a}B \\ A &\rightarrow B \mid S \\ B &\rightarrow \mathrm{b} \mid \epsilon \end{aligned}$$

Removing ϵ rules

Removing $B \to \epsilon$:

$$
\begin{array}{lcl}
S_0 & \to & S \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \\
A & \to & B \mid S \mid \epsilon \\
B & \to & \mathrm{b}
\end{array}
$$

Removing ϵ rules

Removing $B \to \epsilon$:

$$\begin{array}{lcl} S_0 & \to & S \\ S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \\ A & \to & B \mid S \mid \epsilon \\ B & \to & \mathrm{b} \end{array}$$

Removing $A \to \epsilon$

$$\begin{array}{lcl} S_0 & \to & S \\ S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \mid S \\ A & \to & B \mid S \\ B & \to & \mathrm{b} \end{array}$$

Removing unit rule

Removing $S \to S$

$$
\begin{array}{lcl}
S_0 & \to & S \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
A & \to & B \mid S \\
B & \to & \mathrm{b}
\end{array}
$$

Removing unit rule

Removing $S \to S$

$$
\begin{array}{lcl}
S_0 & \to & S \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
A & \to & B \mid S \\
B & \to & \mathrm{b}
\end{array}
$$

Removing $S_0 \to S$

$$
\begin{array}{lcl}
S_0 & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
A & \to & B \mid S \\
B & \to & \mathrm{b}
\end{array}
$$

More unit rules

Removing $A \to B$

$$
\begin{array}{lcl}
S_0 & \to & S \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
A & \to & S \mid \mathrm{b} \\
B & \to & \mathrm{b}
\end{array}
$$

More unit rules

Removing $A \to B$

$$
\begin{array}{lcl}
S_0 & \to & S \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
A & \to & S \mid \mathrm{b} \\
B & \to & \mathrm{b}
\end{array}
$$

Removing $A \to S$

$$
\begin{array}{lcl}
S_0 & \to & ASA \mid \mathrm{a}B \mid SA \mid AS \\
S & \to & ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid AS \\
A & \to & \mathrm{b} \mid ASA \mid \mathrm{a}B \mid \mathrm{a} \mid SA \mid SA \\
B & \to & \mathrm{b}
\end{array}
$$

Converting the remaining rules

$$
\begin{array}{lcl}
S_0 & \rightarrow & AA_1 \mid UB \mid \mathrm{a} \mid SA \mid AS \\
S & \rightarrow & AA_1 \mid UB \mid \mathrm{a} \mid SA \mid AS \\
A & \rightarrow & \mathrm{b} \mid AA_1 \mid UB \mid \mathrm{a} \mid SA \mid AS \\
A_1 & \rightarrow & SA \\
U & \rightarrow & \mathrm{a} \\
B & \rightarrow & \mathrm{b}
\end{array}
$$