01204211 Discrete Mathematics
Lecture 11a: Context-free languages and grammars (1)[1]

Jittat Fakcharoenphol

August 29, 2024

[1] Based on lecture notes of *Models of Computation* course by Jeff Erickson.

How to program a computer?

High-level programming languages

- 1954 - FORTRAN

High-level programming languages

- 1954 - FORTRAN
- 1958 - LISP

High-level programming languages

- 1954 - FORTRAN
- 1958 - LISP
- 1958 - COBOL

High-level programming languages

- 1954 - FORTRAN
- 1958 - LISP
- 1958 - COBOL
- 1960 - ALGO60

Backus-Naur form

Python grammar

Building up languages

Regular languages:

- Contenation
- Union
- Kleene star

Building up languages

Regular languages:

- Contenation
- Union
- Kleene star

Context-free grammar:

- Contenation
- Union
- Recursion

Example

$$
\begin{aligned}
S &\rightarrow 0S1 \\
S &\rightarrow \varepsilon
\end{aligned}
$$

Example

$$\begin{aligned} S &\rightarrow 0S1 \\ S &\rightarrow \varepsilon \end{aligned}$$

You can use "|" to write production rules more succinctly.

$$S \rightarrow 0S1 \mid \varepsilon$$

Definition

A **context-free grammer** consists of the following components:

- a finite set Σ, a set of *symbols* (or *terminals*),
- a finite set Γ disjoint from Σ, a set of *non-terminals* (you can think of them as variables),
- a finite set R of *production rules* of the form $A \to w$ where $A \in \Gamma$ and $w \in (\Sigma \cup \Gamma)^*$ is a string of symbols and variable, and
- a *starting* non-terminal (usually the non-terminal of the first production rule).

Another example

$$\begin{array}{rcl} S & \rightarrow & A \mid B \\ A & \rightarrow & 0A \mid 0C \\ B & \rightarrow & B1 \mid C1 \\ C & \rightarrow & \varepsilon \mid 0C1 \end{array}$$

Here $\Sigma = \{0, 1\}$ and $\Gamma = \{S, A, B, C\}$.

Applying the rules

If you have strings $x, y, z \in (\Sigma \cup \Gamma)^*$ and the production rule

$$A \to y,$$

You can apply the rule to the string xAz. This yields the string

$$xyz.$$

We use the notation

$$xAz \rightsquigarrow xyz$$

to describe this application.

Derivation

We say that z derives from x if we can obtain z from x by production rule applications, denoted by $x \rightsquigarrow^* z$.
Formally, for any string $x, z \in (\Sigma \cup \Gamma)^*$, we say that $x \rightsquigarrow^* z$ if either

- $x = z$, or
- $x \rightsquigarrow y$ and $y \rightsquigarrow^* z$ for some string $y \in (\Sigma \cup \Gamma)^*$.

$L(w)$

The *language* $L(w)$ of string $w \in (\Sigma \cup \Gamma)^*$ is the set of all strings in Σ^* that derive from w, i.e.,

$$L(w) = \{x \in \Sigma^* \mid w \rightsquigarrow^* x\}.$$

$L(w)$

The *language* $L(w)$ of string $w \in (\Sigma \cup \Gamma)^*$ is the set of all strings in Σ^* that derive from w, i.e.,

$$L(w) = \{x \in \Sigma^* \mid w \leadsto^* x\}.$$

The language **generated by** a context-free grammar G, denoted by $L(G)$ is the language of its starting non-terminal.

$L(w)$

The *language* $L(w)$ of string $w \in (\Sigma \cup \Gamma)^*$ is the set of all strings in Σ^* that derive from w, i.e.,

$$L(w) = \{x \in \Sigma^* \mid w \rightsquigarrow^* x\}.$$

The language **generated by** a context-free grammar G, denoted by $L(G)$ is the language of its starting non-terminal.
A language L is **context-free** if there exists some context-free grammar G such that $L(G) = L$.

Grammar G_1

$$
\begin{aligned}
S &\rightarrow NP\,VP \\
NP &\rightarrow CN | CN\ PP \\
VP &\rightarrow CV | CV\ PP \\
PP &\rightarrow PREP\ CN \\
CN &\rightarrow ART\ N \\
CV &\rightarrow V | V\ NP \\
ART &\rightarrow \texttt{a} | \texttt{the} \\
N &\rightarrow \texttt{boy} | \texttt{girl} | \texttt{flower} \\
V &\rightarrow \texttt{touches} | \texttt{likes} | \texttt{sees} \\
PREP &\rightarrow \texttt{with}
\end{aligned}
$$

Small English grammar

$$
\begin{array}{rcl}
S & \rightarrow & NP\,VP \\
NP & \rightarrow & CN|CN\ PP \\
VP & \rightarrow & CV|CV\ PP \\
PP & \rightarrow & PREP\ CN \\
CN & \rightarrow & ART\ N \\
CV & \rightarrow & V|V\ NP \\
ART & \rightarrow & \texttt{a}|\texttt{the} \\
N & \rightarrow & \texttt{boy}|\texttt{girl}|\texttt{flower} \\
V & \rightarrow & \texttt{touches}|\texttt{likes}|\texttt{sees} \\
PREP & \rightarrow & \texttt{with}
\end{array}
$$

- Examples of strings in $L(G_2)$ are:
  - `a boy sees`

Small English grammar

$$
\begin{aligned}
S &\rightarrow NP\,VP \\
NP &\rightarrow CN | CN\ PP \\
VP &\rightarrow CV | CV\ PP \\
PP &\rightarrow PREP\,CN \\
CN &\rightarrow ART\ N \\
CV &\rightarrow V | V\ NP \\
ART &\rightarrow \texttt{a} | \texttt{the} \\
N &\rightarrow \texttt{boy} | \texttt{girl} | \texttt{flower} \\
V &\rightarrow \texttt{touches} | \texttt{likes} | \texttt{sees} \\
PREP &\rightarrow \texttt{with}
\end{aligned}
$$

- Examples of strings in $L(G_2)$ are:
  - `a boy sees`
  - `the boy sees a flower`

Small English grammar

$$
\begin{array}{rcl}
S & \rightarrow & NP\ VP \\
NP & \rightarrow & CN | CN\ PP \\
VP & \rightarrow & CV | CV\ PP \\
PP & \rightarrow & PREP\ CN \\
CN & \rightarrow & ART\ N \\
CV & \rightarrow & V | V\ NP \\
ART & \rightarrow & \texttt{a} | \texttt{the} \\
N & \rightarrow & \texttt{boy} | \texttt{girl} | \texttt{flower} \\
V & \rightarrow & \texttt{touches} | \texttt{likes} | \texttt{sees} \\
PREP & \rightarrow & \texttt{with}
\end{array}
$$

- Examples of strings in $L(G_2)$ are:
  - `a boy sees`
  - `the boy sees a flower`
  - `a girl with a flower likes the boy`

Parse tree

$$
\begin{aligned}
S &\rightarrow A \mid B \\
A &\rightarrow \mathtt{0}A \mid \mathtt{0}C \\
B &\rightarrow B\mathtt{1} \mid C\mathtt{1} \\
C &\rightarrow \varepsilon \mid \mathtt{0}C\mathtt{1}
\end{aligned}
$$

Parse tree

- 00011

$$
\begin{aligned}
S &\rightarrow A \mid B \\
A &\rightarrow 0A \mid 0C \\
B &\rightarrow B1 \mid C1 \\
C &\rightarrow \varepsilon \mid 0C1
\end{aligned}
$$

Parse tree

- 00011
- 01111

$$\begin{aligned} S &\rightarrow A \mid B \\ A &\rightarrow 0A \mid 0C \\ B &\rightarrow B1 \mid C1 \\ C &\rightarrow \varepsilon \mid 0C1 \end{aligned}$$

Parse tree

- 00011
- 01111
- 111110

$$
\begin{aligned}
S &\rightarrow A \mid B \\
A &\rightarrow 0A \mid 0C \\
B &\rightarrow B1 \mid C1 \\
C &\rightarrow \varepsilon \mid 0C1
\end{aligned}
$$

Parse tree

- 00011
- 01111
- 111110

$$
\begin{array}{lcl}
S & \rightarrow & A \mid B \\
A & \rightarrow & 0A \mid 0C \\
B & \rightarrow & B1 \mid C1 \\
C & \rightarrow & \varepsilon \mid 0C1
\end{array}
$$

Parse tree

`a girl with a flower likes the boy`

$$
\begin{aligned}
S &\rightarrow NP\,VP \\
NP &\rightarrow CN | CN\ PP \\
VP &\rightarrow CV | CV\ PP \\
PP &\rightarrow PREP\ CN \\
CN &\rightarrow ART\ N \\
CV &\rightarrow V | V\ NP \\
ART &\rightarrow \texttt{a} | \texttt{the} \\
N &\rightarrow \texttt{boy} | \texttt{girl} | \texttt{flower} \\
V &\rightarrow \texttt{touches} | \texttt{likes} | \texttt{sees} \\
PREP &\rightarrow \texttt{with}
\end{aligned}
$$

Ambiguity

- $1 + 1 * 1$

$$S \quad \rightarrow \quad 1 \mid S + S \mid S * S$$

Ambiguity

- $1 + 1 * 1$
- $1 + 1 + 1 + 1 + 1$

$$S \quad \rightarrow \quad 1 \mid S + S \mid S * S$$

Ambiguity

- $1 + 1 * 1$
- $1 + 1 + 1 + 1 + 1$

$$S \quad \rightarrow \quad 1 \mid S + S \mid S * S$$

- A string w is **ambiguous** with respect to a grammar G if more than one parse tree for w exists.
- A grammar G is **ambiguous** if some string is ambiguous with respect to G.