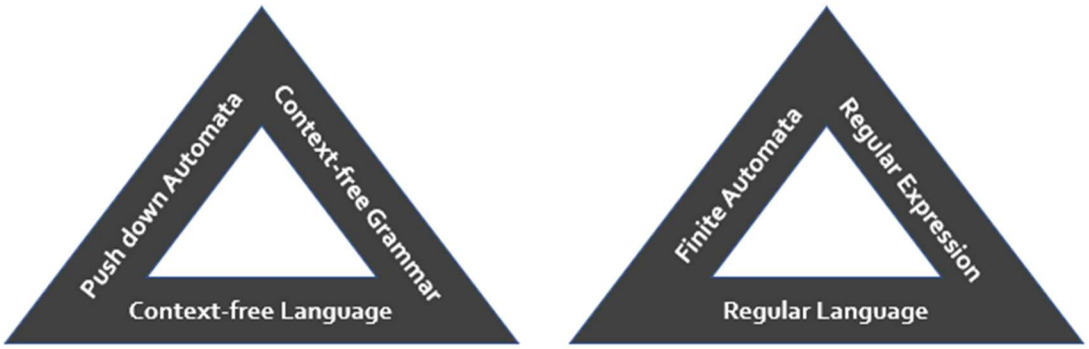
CSE331 Automata and Computability

# Lecture 13.2: Introduction to Context-free Grammar - 2

*Presenter: Azwad Anjum Islam (AAI)* *Scribe: Mujtahid Al-Islam Akon (AKO)*

In this lecture, you will learn about more about the Context-free Grammar and how they work.

- We know that any regular language can be described by a method of description called Regular Expression. Similarly, any context-free language (CFL) can be described by a method of description called Context-free Grammar (CFG).
- We know that for any regular language, there exists an abstract machine that recognizes that particular language called Finite Automaton (FA). Similarly, for any CFL, there exists an abstract machine that recognizes that particular language called Pushdown Automaton (PDA).
- The following triangles portray the above relations-




## Context-free Grammar

In a context free grammar, we express a language by a set of rules (also called **productions**). These rules describe how a string of the language can be generated. This can be best illustrated using the following example. Suppose, the following production rules are given for a language:

$1. S \rightarrow 0S1$

$2. S \rightarrow \epsilon$

Let us show how to generate a string using the rules given. Start with first symbol $S$, then replace any symbol with the right-hand-side of a rule. Continue this until no symbol appears.

| | |
|---|---|
| $S$ | (Start with $S$) |
| $\rightarrow \mathbf{0S1}$ | (Replace with $0S1$ using rule-1) |
| $\rightarrow 01$ | (Replace $S$ with $\epsilon$ using rule-2) |

So, we get $01$.
Let's try again.

$S$ (Start with $S$)
$\rightarrow 0S1$ (Rule-1)
$\rightarrow 00S11$ (Rule-1 again)
$\rightarrow 0011$ (Rule-2)
This time we get, $0011$.

Is there any pattern arising? Let's try for the last time.
$S$ (Start with $S$)
$\rightarrow 0S1$ (Rule-1)
$\rightarrow 00S11$ (Rule-1 again)
$\rightarrow 000S111$ (Rule-1 once more)
$\rightarrow 000111$ (Rule-2)

Here we get $000111$.

Now, it is obvious that the above grammar (CFG to be specific) is generating only the set of strings consisting of some 0's followed by equal number of 1's. So, the above two production rules i.e. the above CFG, generating the language, $L = \{w | w = 0^n 1^n, n \geq 0\}$. In previous lectures, we learnt that this language is not regular. So, it can't be described by either a regular expression or a finite automaton.

## Context free grammar notations

We have already seen that a context free grammar is expressed by a set of rules. Beside the rules, there are some implicit notations that you need to understand:

- The symbols that appear on the left side of a rule are called **non-terminals** or **Variables**. The non-terminal symbol needs to be replaced. There can't be any non-terminal symbol in the final string because these symbols are not a member of the alphabet $\Sigma$.
- The symbols that appear only on the right side and never appear on the left side of a rule are called **terminal** symbols. The terminal symbols are never replaced as they are not present on the left side of a rule. The terminal symbols are members of $\Sigma$. The final string generated consists of only terminal symbols.
- The non-terminal symbol that appears on the left side of the first rule is called the **start symbol** of the grammar. The production of a string will start with this symbol (see our examples above).