Module 3 Overview

Context-Free Languages (Chapter 2)

Module Intro

In Chapter 1 we introduced two different, though equivalent, methods of describing languages: finite automata and regular expressions. We showed that many languages can be described in this way but that some languages cannot.

In this chapter we present context-free grammars, a more powerful method of describing languages. Such grammars can describe certain features that have a recursive structure, which makes them useful in a variety of applications. Context-free grammars were first used in the study of human languages. One way of understanding the relationship of terms such as noun, verb, and preposition and their respective phrases leads to a natural recursion because noun phrases may appear inside verb phrases and vice versa. Context-free grammars help us organize and understand these relationships.

An important application of context-free grammars occurs in the specification and compilation of programming languages. A grammar for a programming language often appears as a reference for people trying to learn the language syntax. Designers of compilers and interpreters for programming languages often start by obtaining a grammar for the language. Most compilers and interpreters contain a component called a parser that extracts the meaning of a program prior to generating the compiled code or performing the interpreted execution. A number of methodologies facilitate the construction of a parser once a context-free grammar is available. Some tools even automatically generate the parser from the grammar.

Objectives

Upon completion of this module, students will be able to:

1. Define the relationship between context-free languages and regular languages
2. Define the relationship between context-free languages and PDAs
3. Define the relationship between PDAs and finite automata
4. Define and create a PDA for given languages
5. Define and create context-free grammars, CFGs, for given languages
6. Determine what strings a given CFG or PDA recognizes
7. Define what it means for a language to be closed under an operation on that language
8. Determine whether a given CFG is ambiguous or not
9. Place a given CFG into Chomski Normal Form

Readings and Resources

You can explore this module's information in multiple ways:

- View slides
- Access this information from the textbook (**Chapter 2)** Course Textbook (https://virginiacommonwealth.instructure.com/courses/119232/pages/course-textbook)
- Solve practice activities

Module at a Glance

Below is an overview of this module. Pay particular attention to items with **points values** and **due dates,** as these are graded assignments. (Note: module overview will not display if you are accessing Canvas on a mobile device.)

Module 3: Context-Free Languges (Chapter 2)

Module 3 Overview: Context-Free Languages (Chapter 2)

M3 Chapter 2: Notes - Context-Free Languages 2.1

M3: 2.1 Practice
0 pts

M3 Chapter 2: Notes - Properties of Context-Free Languages 2.2

M3: 2.2 Practice
0 pts

M3 Chapter 2: Notes - Pushdown Automata 2.3

M3: 2.3 Practice
0 pts

M3 Chapter 2: Notes - Equivalence of PDAs and CFGs 2.4

M3: 2.4 Practice
0 pts

M3: Challenge Problem Set
0 pts

M3 Chapter 2 Review Key

Reflection 3 - Exam 2
Oct 19, 2025, 11:59 PM 10 pts This assignment was locked Oct 20 at 11:59pm.

Will unlock Sep 1 at 12:00AM