**Air Force Institute of Technology**

**Graduate School of Engineering and Management**

**Department of Electrical and Computer Engineering**

**CSCE 532 Automata and Formal Languages**

**Winter 2019**

# Day 9

# Non-Context-Free and Deterministic Context-Free Languages

§2.3 Non-Context-Free Languages

### Team Effort (Sipser Problem 2.30a) (cont.)

Use the pumping lemma (for CFLs) to show that is not context-free.

### Solution

Assume is context-free. Then has a pumping length such that every with can be written in a way that satisfies the conditions of Theorem 2.34. Let . Then, because , we must have entirely within adjacent groups of s and s. This yields the following possibilities.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  | Contradiction |
|  |  |  |  |  |
|  |  |  |  | Case 1 |
|  |  |  |  | Case 2 |
|  |  |  |  |  |
|  |  |  |  | Case 2 |
|  |  |  |  |  |
|  |  |  |  | Case 2 |
|  |  |  |  |  |
|  |  |  |  | Case 1 |
|  |  |  |  | Case 1 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

In Case 1, where . Thus, either

* or
* .

Similarly, in Case 2: where . Thus, because either or or both.

We have shown that there is no way to write in a way that satisfies the conditions of the Pumping Lemma (for CFLs), which contradicts the assumption that is context-free.

### Practice (Sipser Problem 3.21)

Let be the language of all palindromes over containing equal numbers of s and s. Show that is not context-free.

### Solution

Proof: Assume is context-free. Then has a pumping length such that every with can be written in a way that satisfies the conditions of Theorem 2.34. Let . By the third condition , so we have the following:

Four cases:

1. is all zeroes

Pumping violates both constraints

1. is all ones

Pumping violates the condition that the number of ones and zeroes are equal

1. is ones followed by zeroes

Pumping violates the palindrome rule

1. is zeroes followed by ones

Pumping violates the palindrome rule

In every case, contradiction, so assumption is false. Therefore, not a CFL.

§3.1 Turing Machines

### Definition

A **Turing machine** is a 7-tuple , where are all finite sets and

1. is the set of states,
2. is the input alphabet not containing the blank symbol ␣,
3. is the tape alphabet, where ␣ and ,
4. is the transition function,
5. is the start state,
6. is the accept state, and
7. is the reject state, where .

### Terminology and Notation

A **configuration** of a TM specifies its current state, tape contents, and head location. Specifically, for state and strings , the configuration specifies that the machine is in state , that the tape contains , and that the head is positioned over the first symbol of .

The **start configuration** of a TM with input is .

Any configuration involving the state () is an **accepting (rejecting) configuration**.

A **halting configuration** is one that is accepting or rejecting.

A configuration **yields** configuration if the TM can legally go from to in one step.

A TM **accepts** input if a sequence of configurations exists, where

1. is the start configuration of on input ,
2. Each yields , and
3. is an accepting configuration.

The collection of strings that accepts is the **language of** , or the **language recognized by** , denoted .

In general TMs can accept, reject, or loop.

### Definition

Call a language **Turing-recognizable** (or **recursively enumerable**) if some TM recognizes it.

### Definition

Call a language **Turing-decidable** (or **decidable**, or **recursive**) if some TM decides it.