CSc 30400 Introduction to Theory of Computer Science

3rd Homework Set

Due Date: 3/25

Instructions:

- This homework set covers Chapter 2 (CF languages).
- Submit your solutions by email or hand them in class **before the beginning of the class!**
- Prepare your solution either in electronic format or on paper. Using JFLAP earns a 5% credit. Use JFLAP to prepare any PDAs that you probably create. **Do not use JFLAP's automatic features for CNF and CYK!** (you can only use them to verify correctness).
- If your electronic solutions consist of more than one files, compress all your files into one file, name it using your first and last name and the homework set number and send the zipped file in an attachment. You should also include a word or pdf report.
- Exercises are divided into three sections: Practice, Easy and Hard. Practice questions just help you understand the material and are a good way to get easy points. "Easy questions" should be relatively easy but require more effort or understanding than the practice ones (not all easy questions have the same difficulty)! Hard questions are a bit tougher (again, not all of them have the same difficulty). All the questions earn 10 points if not otherwise stated.

1 Practice questions

P 1. Consider the following grammar:

$$E \to E + T|T$$

$$T \to T \times F|F$$

$$F \to (E)|N$$

$$N \to 0|1|2|3|4|5|6|7|8|9$$

Give parse trees and derivations for each string

a. 2

b. $5 + 3$

c. $3 + 7 + 5$

d. $4 + 6 \times 3$

e. $(2 + 5) \times 6$

2 Easy questions

E 1. Give context free grammars for the following languages over the alphabet $\Sigma = \{0, 1\}$. Give right linear grammars for those that are regular (failing to recognize a regular language results in getting only 1 out of 2 points partial credit).

$L_1 = \{w : w \text{ contains at least three 1s}\}$

$L_2 = \{0^n 1^{2n} : n \geq 0\}$

$L_3 = \{w : |w| \geq 2 \text{ and } w \text{ starts and ends with the same symbol}\}$

$L_4 = \{w : w = w^R, \text{ that is } w \text{ is palindrome}\}$

$L_5 = \{0^n 1^k 0^n : n \geq 0, k \equiv 0 \mod 3\}$

E 2. (6 points) The following languages over $\Sigma = \{0, 1\}$ are regular but giving a DFA for them might be tiresome. Give PDAs for each one (use the additional power of the stack to simplify your automaton).

$L_1 = \{wtw^R : w, t \in \Sigma^*, |w| = 4\}$

$L_2 = \{w : w \text{ contains exactly six 0s and at least five 1s}\}$

E 3. Consider the following grammar in Chomsky Normal Form:

$$S \to BA$$

$$A \to BB|0$$

$$B \to AA|1$$

For each of the following strings, decide whether the above grammar produces the string or not running the CYK algorithm. If the answer is yes write a derivation (derivation gains 2 points).

a. 0100

b. 1100

E 4. Convert the following Context Free grammar into Chomsky Normal Form.

$S \to aSb|bSa|\varepsilon$

E 5. (4 points) Parsing is a part of the compilation process. Using a CF grammar which is built in the compiler, the rules match with valid expressions consisted by the tokens which were created during the lexical analysis. If a matching is found for the whole program, a structure called an *abstract syntax tree* is built, which is used later on in the evaluation procedure and further semantical analysis of the program. It is crucial for the evaluation procedure that the grammar is as unambiguous as possible in order to avoid misinterpretation of what a programmer was intending to write and what the program actually computes.

Consider the following part of such a built-in CF grammar:

$$\begin{aligned} <STMT> \quad \to \quad & if <COND> then <STMT> | \\ & if <COND> then <STMT> else <STMT> \end{aligned}$$

Is it ambiguous? Why? What do we usually do to fix this?

3 Hard questions

H 1. An **Extended Right Linear Grammar** is a grammar with productions of the form:

$$A \to \varepsilon$$
$$A \to w$$
$$A \to wB$$

where $A, B \in V$ and $w \in \Sigma^*$. Show that Extended Right Linear Grammars produce the Regular Languages.

Hint: Show that Right Linear Grammars and Extended Right Linear Grammars are equivalent. The first direction is immediate (why?). The "difficult part" is for every Extended Right Linear Grammar to construct an equivalent Right Linear one.

H 2. The following languages are not context free:

$$\begin{aligned} L_1 &= \{a^n b^n c^n, n \geq 0\} \\ L_2 &= \{ww : w \in \{0,1\}^*\} \\ L_2' &= \{w\#w : w \in \{0,1\}^*\} \end{aligned}$$

This means that they cannot be recognized by a PDA (this fact can be proven using a similar argument with Pumping Lemma but for PDAs -see section 2.3 if interested). However, an enhanced PDA with two stacks (instead of just one), is capable of recognizing all the above languages. This automaton is called a 2-PDA.

Construct 2-PDAs for L_1 and L_2. If L_2 seems too difficult try L_2' for partial credit.

Hint: In order to define formally a 2-PDA you should take care of both stacks, so in the transition you need to add an extra $a \to b$ to denote the change in the second stack.

H 3. (a) Show that $L_a = \{a^k b^n c^n : n, k \geq 0\}$ and $L_c = \{a^n b^n c^k : n, k \geq 0\}$ are both context free.

(b) Show (using part a and H 2) that context free languages are not closed under intersection.

(c) Show (using part b) that the context free languages are not closed under complement.