BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI
CS F351 (Theory of Computation) Mid Semester Exam, 2023 – 24
[Open Book]

October 13, 2023 **90 Minutes** **MM: 60**

Note: There are six questions. Answer all parts of a question together. ε denotes empty string.

Q1 [10M]. For two languages L_1 and L_2 over Σ, we define the quotient of L_1 and L_2 to be the language L_1/L_2 = { x | for some y ∈ L_2, xy ∈ L_1 }. Answer the following:

- **a)** If L_1 = {fish, dog, carrot}, L_2 = {rot, cheese}, what is L_1 / L_2?
- **b)** TRUE/FALSE: "In general, if L_2 contains ε, then L_1/L_2 will contain L_1". Justify in one statement.
- **c)** TRUE/FALSE: "In general, if L_2 = P U Q, then L_1/L_2 = (L_1/P) U (L_2/Q)". Justify in one statement.
- **d)** TRUE/FALSE: "In general, if L_1 = P U Q, then L_1/L_2 = (P/L_2) U (Q/L_2)". Justify in one statement.
- **e)** Prove that regular languages are closed under quotient operation.

Q2 [10M]. Consider the following three CFG's, G_1 to G_3, and answer the questions that follow:

G_1 = ({S_1, A, a, b}, {a, b}, {S_1 -> aabA | Aba , A -> aA | bA | a | b | ε}, S_1)
G_2 = ({S, Y, a, b} , {a, b} , { S → aSb|bY |Ya , Y → bY |aY |ε}, S)
G_3 = ({S_2, A, a, b}, {a, b}, {S_2 -> AaaA , A -> bA | ε}, S_2)

- **a)** What is the language of G_1, G_2, and G_3?
- **b)** Write "simplest possible" CFG G_4 such that L(G_4) is the complement of L(G_2)?
- **c)** Write a CFG G_5 such that L(G_5) = L(G_1) U L(G_3)*. Writing CFG G_5 should depict some algorithmic process for which understanding the language is not required.

Q3 [12M]. Answer the following:

- **a)** Prove/Disprove: "If L is a non-regular language, then L* must also be a non-regular language".
- **b)** A CFG is known as strongly *right linear* if all the rules in CFG are of the form P → aQ or P → ∈, where P,Q are variables and a is a terminal symbol. Prove/disprove: "α is the language of strongly right linear grammar if and only if α is a regular language".
- **c)** Consider a language L over an alphabet Σ. String *x* is known as a prefix of string *y* if *y* = *xz* for some string *z*. For e.g., the prefixes of *abbab* are *a*, *ab*, *abb*, *abba*, *abbab*. By using *L*, it is possible to create the language dupcPre(*L*) by duplicating the prefixes of strings in *L*.

  dupcPre(*L*) = {*xy* | *y* ∈ *L*, and *x* is a prefix of *y*}.

  Prove/Disprove: If *L* is regular, then dupcPre(*L*) must also be regular.

Q4 [8M]. Consider the following CFG G = ({S, A, a, b}, {a, b}, R, S), where R is given as:

S -> aS | bA | b
A -> aA | bS | ε

Also, it is given that G is a regular language. Convert G to NFA and NFA to DFA. Making of NFA and corresponding DFA should depict some algorithmic process for which understanding the language is not required.

Q5 [10M]. Over ∑ = {0, 1}, let L = {w ∈ ∑* | w has even number of 1's}.

- **a)** Make a DFA with two states accepting L.
- **b)** Using pumping theorem, consider the following proof which proves L is not a regular language:

Proof: Let string w = 0110 which satisfies the condition |w| ≥ p, where p is the pumping length. We can break string w into three parts x, y, and z, such that w = xyz, as follows: x = 0, y = 1, z = 10. This satisfies the condition |y| > 0, and |xy| ≤ p. However, now $xy^2z \notin L$. Hence, L is not a regular language.
What is wrong with the above proof? Your answer should be complete.

Q6 [10M]. Over ∑ = {(,), int, +, *}, let L = {w ∈ ∑* | w is a legal arithmetic expression}. For example, the following are two legal arithmetic expressions.

- int + int * int
- ((int + int) * (int + int)) + (int)

Assuming $ as the bottom marker symbol of the stack and # as the end marker symbol of the input string, following is a deterministic PDA (DPDA) for L with four states. The PDA has few missing transitions. Complete it. Redraw the complete DPDA (with four states) for L.