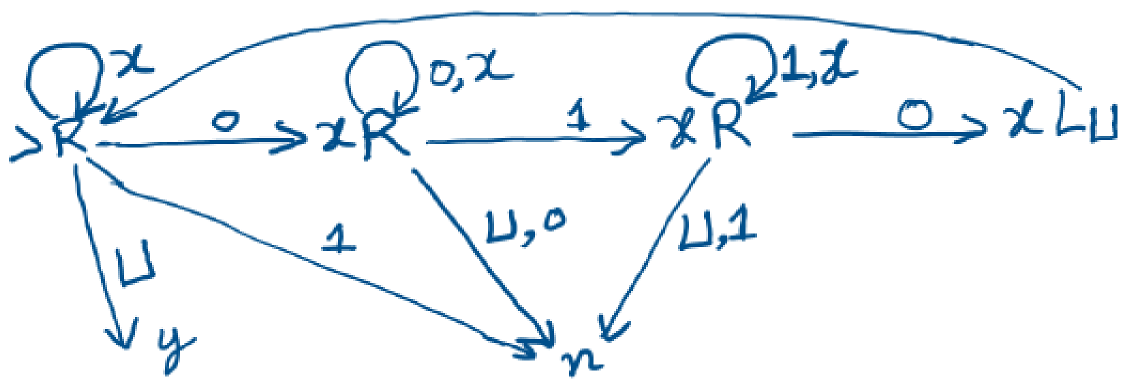
Birla Institute of Technology and Science, Pilani
First Semester 2020 - 21
Test-4, OPEN BOOK, November 20, 2020
Theory of Computation (CS F351)

Marking Scheme is given wherever required.

Q1. [*5 Marks*] Consider the Turing Machine (TM) shown below. The initial configuration of this TM is: ⊳ $\underline{\sqcup}$w. Does this TM decides the language L = $\{0^n1^n0^n \mid n \geq 0\}$? If so, just write "YES". Otherwise, write two incorrect strings that this TM accepts. y is the accepting halting state and n is the non-accepting one.



Q2. [*5 Marks*] For a single tape Turing machine T = ({q0, q1, q2, h}, {a, b}, δ, q0, {h}), following is the encoding (i.e. "T"). Assume that the machine starts in initial configuration.
(q00, a000, q01, a011), (q01, a000, q11, a010), (q01, a100, q10, a000), (q10, a100, q10, a011), (q10, a000, q11, a010), (q10, a101, q10, a011)
[**Note**: Other than the standard encoding for symbols as per the textbook, following are the encoding for rest of the symbols: q1 -> q01 ; q2 -> q10 ; a -> a100 ; b -> a101].

Answer the following questions:

a) What is the language accepted by the given TM? Write a regular expression for it.
b) Which is the correct encoding of the halting state of the given TM?
c) When "T" and its input "w" are given as input to a Universal TM, is it (i.e. Universal TM) guaranteed to always halt? Why yes or why no?

Sol:

State	Input	State to move to	Action
Q0	blank	Q1	→
Q1	Blank	halt	←
Q1	a	Q2	Blank
Q2	a	Q2	→
Q2	blank	halt	←
Q2	b	Q2	→

a) e U a(a U b)* Marking Scheme → 1Mark: deduct 0.5 Marks if e is not there.
b) halting state encoding: q11 → 0/1Mark
c) No, the U_{TM} is not guaranteed to always halt because the given TM "T" does not decides the language.
→ If someone writes only YES/NO then zero marks. Else, 1M for yes/No; and 2M for reason.

Q3. [*5 Marks*] You know that a single tape Turing Machine is defined as (K, ∑, δ, s, H). Suppose the initial configuration of the TM is ⊳ ⊔ w . If we put a restriction that a TM cannot write anything on the portion of the tape where the input (i.e. ⊔ w) is present, what all types of languages (i.e. Regular Languages, CFL's, and/or Type-0 languages) does it accepts. Justify briefly.

Sol: The given restricted TM accepts Regular Language → 1M

Justification→ 4M

Q4. [*5 Marks*] Let T1 = (K1, ∑1, δ1, s1, {Y1, N1}) and T2 = (K2, ∑2, δ2, s2, {Y2, N2}) be two single tape Turing Machines that decide languages L1 and L2, respectively. Y1 and Y2 are accepting halting states. Similarly, N1 and N2 are non-accepting halting states. Assuming both T1 and T2 starts in initial configuration, we need to design a single tape Turing Machine T = (K, ∑, δ, s, H) that decides L = L1 U L2.

Is the following construction of T correct? Justify your answer briefly.
T = (K, ∑, δ, s, H), where
K = K1 U K2 U {qn} , ∑ = ∑1 U ∑2 , s = s1 , H = H1 U H2
∀x ∈ ∑1, ∀y ∈ ∑1 – {⊳},
δ = **δ1** U **δ2** U **{(N1, x)(qn, x)}** U **{(qn, y)(qn, ←)}** U **{(qn, ⊳)(s2, →)}**
[**Note**: Transition of the form (q, x)(p, y) means in state q if the symbol read is x, write y in place of x and move to state p.]
The evaluation Criteria:
Incorrect (1 mark)
H = {Y1, Y2, N2} (1.5 mark)
A copy of input should be preserved before simulating T1 on the given input (1.5 marks)
If the input starts with a symbol in ∑2 -∑1, T1 may get stuck, similarly T2 (1 mark).

Q5. [*5 Marks*] Let G1 be a CFG and string x ∈ L(G1). Also, let G2 be a CFG such that L(G2) = L(G1) – {x}. Prove that (by giving an algorithm) computing grammar G2 from grammar G1 is decidable.

[**Hint**: If x is a string, $\{x\}^c$ is a regular language.]

Sol: Since x^c is regular, making a DFA for it is decidable. Also, L(G2) = L(G1) $\cap$ x^c

Computing grammar G2 is decidable because:

1. Every CFG can be converted to a PDA accepting the same language.
2. Intersection of a CFL and Regular language is a CFL. Therefore, it is decidable to get a PDA for L(G1) $\cap$ x^c
3. For every PDA, there exists a CFG accepting the same language.

Q6. [*5 Marks*] Let L = {G | G is a CFG and G generates at least 51 strings}. Prove that L is decidable. You can give an algorithm here as a high-level description of corresponding TM.

[**Hint**: Think about the outcome of the question just before this one.]

Sol: We can decide whether the L(G) is phi or not.

If the L(G) is not phi, generate a string x. Then, generate CFG G2 = L(G) – {x}. This step can be repeated 51 times.