TOC - 2018-19

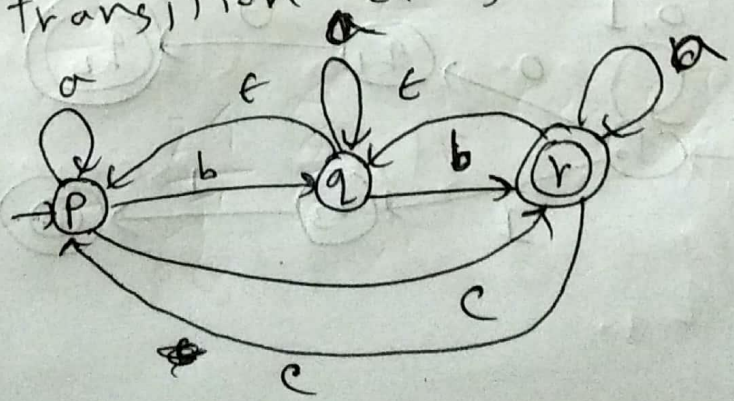
1.a) To determine whether the Dominating Set problem is in NP, we will try to verify the given certificate. If we can verify the certificate in polynomial time, Dominating Set problem is in NP. IF it cannot be verified in polinomial time, it is not in NP.

1.b) i) This opinion is wrong. Because pumping lemma cannot be used to prove a language regular. ~~Because~~ Because following the pumping lemma is not enough for a language to be regular. So, it should only used

for ~~proovi~~ proving a language non-regular. Also only one string cannot prove that a language follows pumping lemma. Because if ~~sting only~~ at least one string in a language violates pumping lemma, the language will be non-regular.

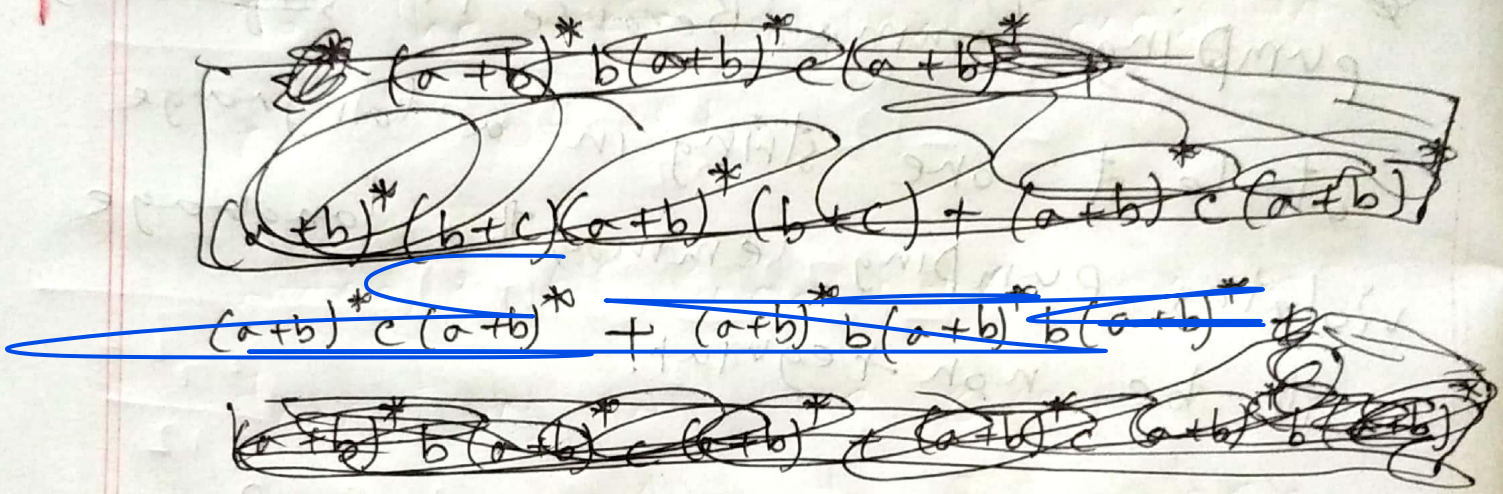
ii) Here choice of y is wrong. Because $|xy| \leq p$, so, y ~~cannot~~ can ~~only~~ only contain 0s and no 1s.

1. c) The transition diagram -



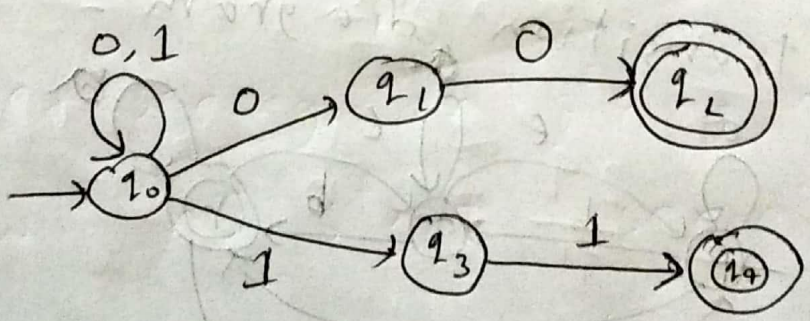
at least

If the string contains ~~only~~ one c, or at least two b's. ~~one b and one c~~ then the NFA will accept. So, general form is: $\Sigma^* c \Sigma^* + \Sigma^* b \Sigma^* b \Sigma^*$



2. a) This FDA accepts all the strings whose last two characters are same.

2. b)



2. c) $\Sigma^*(01+10)+0+1+\epsilon$

3. a)

i) $\emptyset^* \cdot \emptyset^* \cdot \emptyset^* \cdot \emptyset^*$

$= \{\epsilon\} \cdot \{\epsilon\} \cdot \{\epsilon\} \cdot \{\epsilon\}$

$= \{\epsilon\}$

ii) $\emptyset^* \cup \{\epsilon\} = \{\epsilon\} \cup \{\epsilon\} = \{\epsilon\}$

iii) $(\emptyset^* \cup 1)\{0,1\} = \{\epsilon \cup 1\}\{0,1\}$

$= \{0, 1, 10, 11\}$

iv) $(11)^* \emptyset (00)^* = \emptyset$

v) $(\emptyset^* \cup 0)\{0, 1, 11\} = \{\epsilon, 0\}\{0, 1, 11\}$

$= \{0, 1, 11, 00, 01, 011\}$

3.b)

When, ~~$\delta(q, a)$~~ $q \in Q_1$ and $q \notin F_1$ the transitions will be according to the ~~old~~ original transition rules of N_1.

When, $q \in F_1$ and $a \neq \epsilon$, again transitions will be according to the original rules. But when, $a = \epsilon$, along with the original transitions, we will add a transition to q_1 (the start state of N_1). Because it means, we are guessing we have finished seeing an string in L_1 and we will not start seeing another from L_1.

when, $q = q_0$ and $a = \epsilon$ ~~we~~

there will be transition to q_1.

Because q_0 is the new start state added to accept the empty string. So, we will add an empty transition to q_1. But if $a \neq \epsilon$, there will be no transition, because q_0 is added only to accept empty string.

4. a) This regular expression describes the binary strings with no consecutive 1's. So, there will be at least one 0 between any two 1's.

4.b)

Start state, $Q_0 = E(q_0)$

$= \{q_0, q_1, q_2, q_4, q_7\}$

$\delta(Q_0, 0) = \{q_1, q_2, q_4, q_3, q_6, q_7, q_8\}$

[Let, Q_1]

$\delta(Q_0, 1) = \{q_5, q_6, q_7, q_1, q_2, q_4\}$

[Let, Q_2]

$\delta(Q_1, 0) = \{q_3, q_6, q_1, q_2, q_4, q_7, q_8\}$

$= Q_1$

$\delta(Q_1, 1) = \{q_5, q_6, q_7, q_1, q_2, q_4, q_9\}$

[Let, Q_3]

$\delta(Q_2, 0) = \{q_3, q_6, q_7, q_1, q_2, q_4, q_8\}$

$= Q_1$

$$\delta(Q_2, 1) = \{q_5, q_6, q_7, q_1, q_2, q_4\}$$

$$= Q_2$$

$$\delta(Q_3, 0) = \{q_3, q_6, q_7, q_1, q_2, q_4, q_8\}$$

$$= Q_1$$

$$\delta(Q_3, 1) = \{q_5, q_6, q_7, q_1, q_2, q_4, q_{10}\}$$

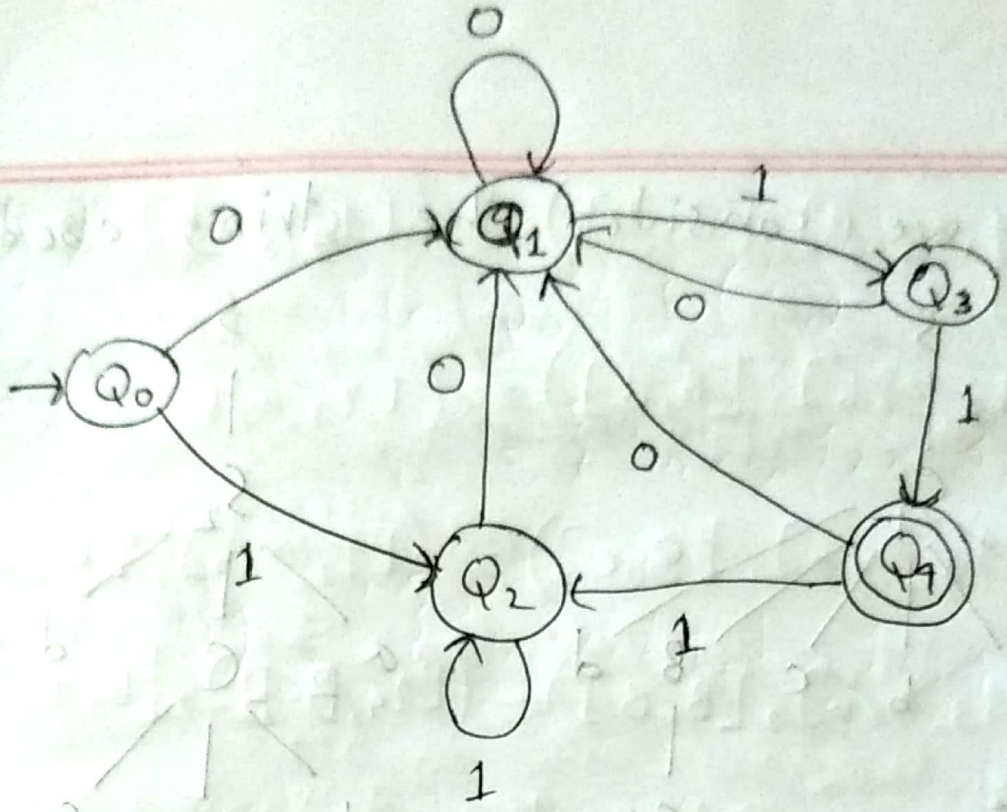
[Let, Q_4]

$$\delta(Q_4, 1) = \{q_5, q_6, q_7, q_1, q_2, q_4\}$$

$$= Q_2$$

$$\delta(Q_4, 0) = \{q_3, q_6, q_7, q_1, q_2, q_4, q_8\}$$

$$= Q_1$$



5. a) $S \rightarrow X0X1X \mid X1X0X$

$X \rightarrow 0X \mid 1X \mid \epsilon$

b) $S \rightarrow 1S0 \mid 0S1 \mid SS \mid \epsilon$

6. $S \rightarrow S_1 \mid S_2$

$S_1 \rightarrow aAbcBd$

$A \rightarrow aAb \mid \epsilon$

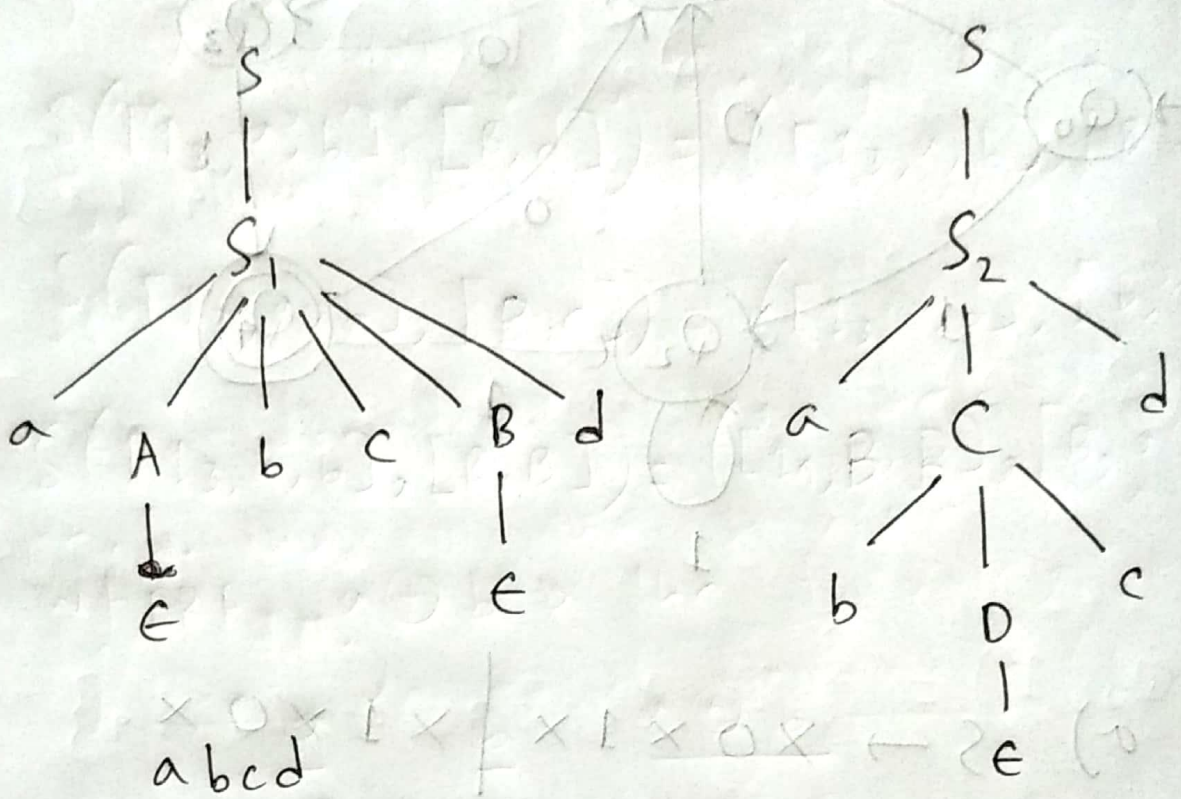
$B \rightarrow cBd \mid \epsilon$

$S_2 \rightarrow aCd$

$C \rightarrow aCd \mid bDc$

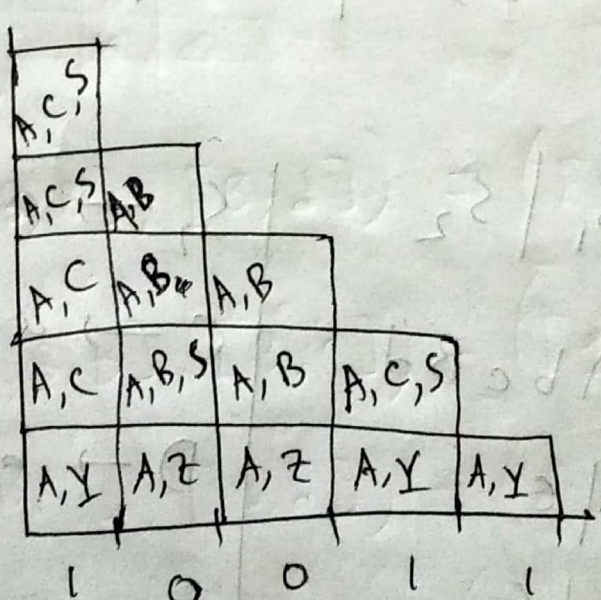
$D \rightarrow bDc \mid \epsilon$

If we consider the string abcd



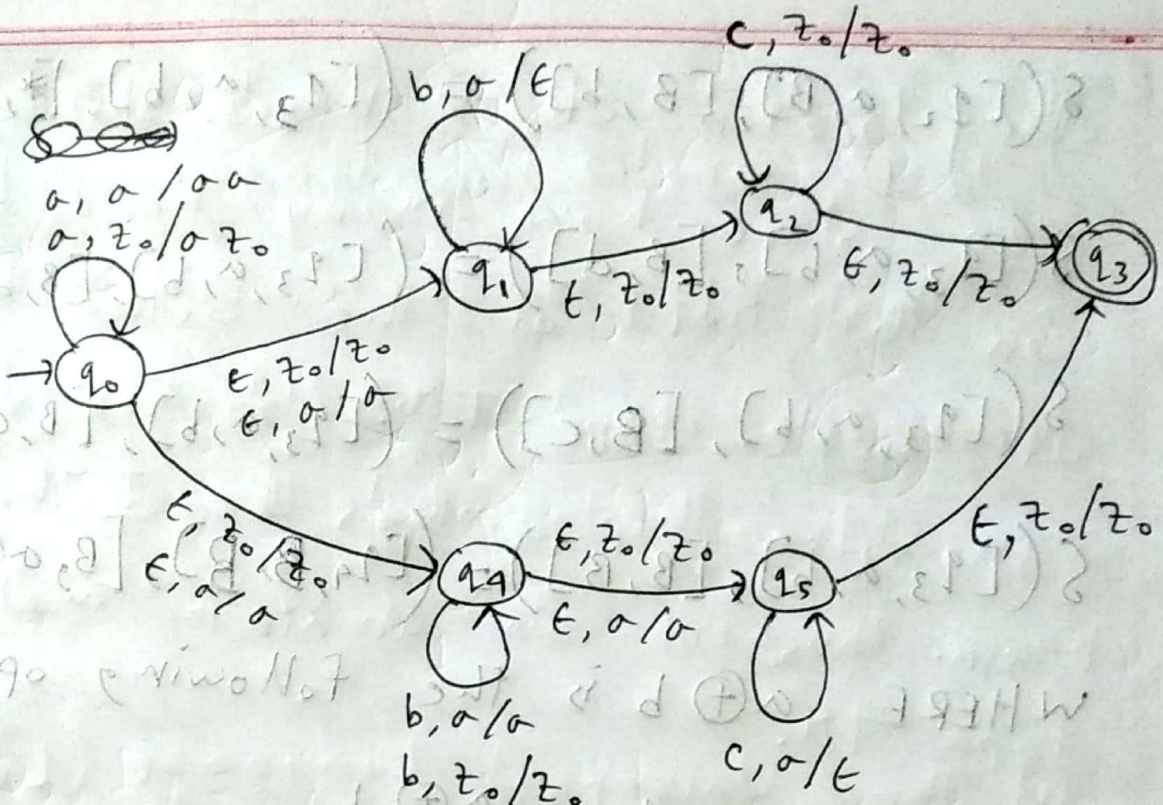
So, the grammar is ambiguous

7.



As S is present in the topmost cell, 10011 is in L(G).

8.



9. The TM will be a TM with two tracks and two symbol storage.

$\delta([q_0, B, B], [B, a]) = ([q_1, a, B], [*, a], R)$

where a can be any bit.

$\delta([q_1, a, B], [B, b]) = ([q_1, a, B], [B, b], R)$

$\delta([q_1, a, B], [B, c]) = ([q_2, a, B], [B, c], R)$

$\delta([q_2, a, B], [*, b]) = ([q_2, a, B], [*, b], R)$

$$\delta([q_2, a, B], [B, b]) = ([q_3, a, b], [*, b], R)$$

$$\delta([q_3, a, b], [B, d]) = ([q_3, a, b], [B, d], R)$$

$$\delta([q_3, a, b], [B, c]) = ([q_3, a, b], [B, c], R)$$

$$\delta([q_3, a, b], [B, B]) = ([q_4, B, B], [B, a \oplus b], L)$$

WHERE $a \oplus b$ is the following operation:

a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

Then,

$$\delta([q_4, B, B], [B, d]) = ([q_4, B, B], [B, d], L)$$

$$\delta([q_4, B, B], [B, c]) = ([q_5, B, B], [B, c], L)$$

$$\delta([q_5, B, B], [B, d]) = ([q_5, B, B], [B, d], L)$$

$$\delta([q_5, B, B], [*, d]) = ([q_5, B, B], [*, d], L)$$

$\delta([q_5, B, B], [B, c]) = ([q_6, B, B], [B, c], L)$

$\delta([q_6, B, B], [B, d]) = ([q_6, B, B], [B, d], L)$

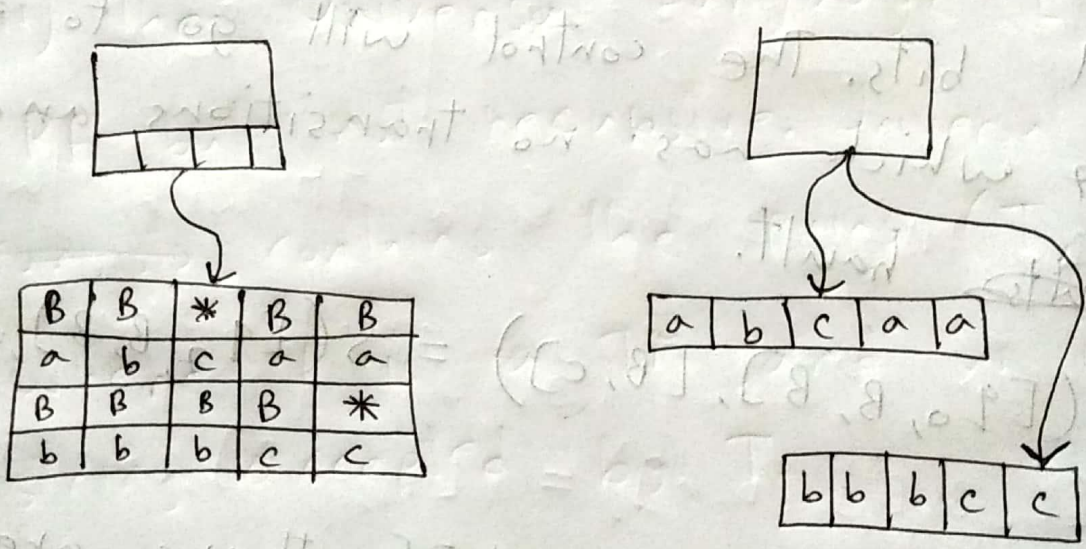
$\delta([q_6, B, B], [*, d]) = ([q_0, B, B], [*, d], R)$

The cycle will start again. If in q_0 state c is encountered we will know that we have done XOR operation on all bits. The control will go to state q_7 which has no transitions and will halt.

$\delta([q_0, B, B], [B, c]) = \delta([q_7, B, B], [B, c], R)$

10. i) It is true. If there are k tapes, we can use a 2k track TM, where half tracks will contain the Tape contents and other half will denote

the location of k heads. Storage can store how many heads are on the left of the actual head of single tape TM. Also the storage can store the symbols head has read. Thus, it can simulate multi tape TM.



ii) It is true. A deterministic multi-tape TM can simulate n steps of computer

in $O(n^3)$. Again, we can simulate a multi tape TM using a single tape TM in $O(n^2)$. So, a single tape TM can simulate n steps of computer in $O(n^6)$ step which is a polynomial.

11. a) Languages ~~that~~ for which a Touring machine can be constructed are called RE languages. ~~That~~ is, Touring recognizable languages are RE languages.

RE languages for which a TM can be constructed that gurantees to hault on any input with or without accepting ~~is call~~ are called recursive languages.

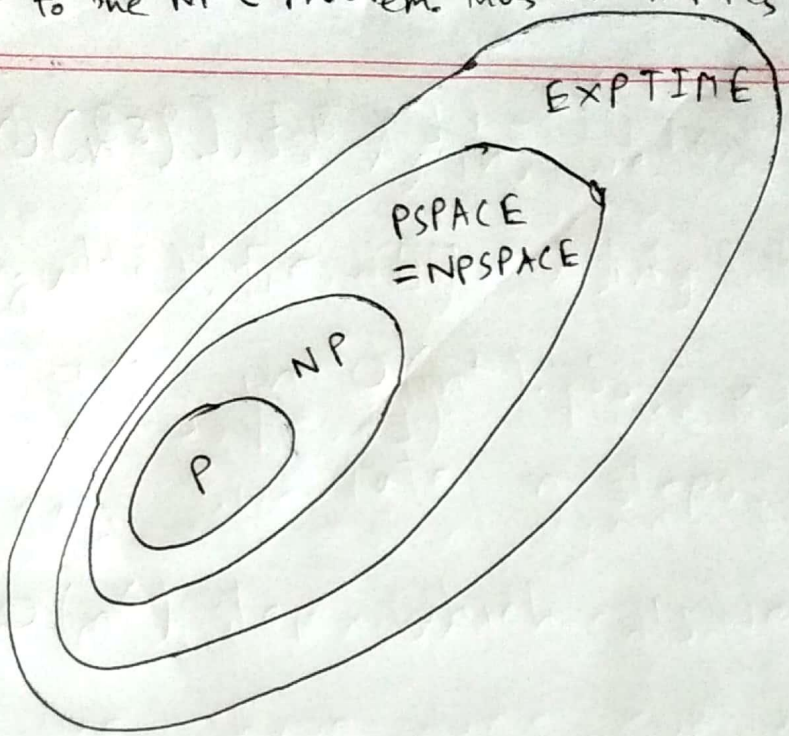
b) i) Diagonalization language (L_D) is a non RE language. Diagonalization language consists the strings w that is not accepted by the TM whose ~~code~~ binary code is w.

ii) The Universal Language (L_u) is RE but not recursive. Universal Langua ~~consists~~ contains the pairs (T, w) where T is a TM and w is a string accepted by T.

12. a) Every problem L in NP has a polynomial time reduction to satisfiability problem (and SAT is in NP). That is satisfiabili problem is NP-complete.

Every language in NP can be polynomially reduced to an NP-complete problem. So, if we can find a polynomial time algorithm for a NP-C problem, every problem in NP can be solved in polynomial time reducing it to the NP-C problem. Thus it implies P = NP.

b)



$P \subseteq NP$, because problems in P can be verified in polynomial time.

~~NP ⊆ PSPACE, because~~

$NP \subseteq NPSPACE$, because, in polynomial time, a TM cannot access more than polynomial ~~space~~ space.

But, NPSPACE = PSPACE

Also, exptime operations may take more than polynomial space.