19CS10039
Kaushal Banthia
1

1

(a)

⊢	a	b	b	a	b	a	c	c	c	d	d	d	⊔	...

~~• Start from left end marker and count number of a's and number of b's and remember them using finite control. This is done by moving right and changing states one at a time, until a 'c' or a 'd' is encountered.~~

~~• start from the ~~last~~ 'a' or 'b' and count the number of 'c's by moving to the right~~

① • Start from left end marker.

② • whenever an 'a' is encountered, mark it as $\bar{a}$ and move right, until an unmarked 'c' is encountered. Then mark it as $\bar{c}$. Then go left ~~to the~~ till you land on a marked element $\bar{a}$ or $\bar{b}$. Start from the next element (on right)
Similarly, whenever a 'b' is encountered, mark is as $\bar{b}$ and move right until an unmarked 'd' is encountered. Then mark it as $\bar{d}$. Then go back left till you land on a marked element $\bar{a}$ or $\bar{b}$. start from the next element (on the right)

③ • If in previous step, an 'a' is encountered and on moving right, in search of an ~~un~~ unmarked 'c', we land upon 'd' or '⊔', then we go to reject state and halt.

④ • If step ② successfully works, then we go to accept state and halt. It successfully works when, while ~~searc~~ searching for an unmarked 'a' or 'b', we land upon '⊔' (or the right end marker, if used).

(b)

$S \longrightarrow ASX$

$S \longrightarrow BSY$

$S \longrightarrow \epsilon$

$YX \longrightarrow XY$

$Y \longrightarrow \cancel{B}\, dZ$

$X \longrightarrow \cancel{A}\, cZ$

$Yd \rightarrow ddZ$

$Xd \rightarrow cdZ$

$Xc \rightarrow ccZ$

$A \longrightarrow aZ$

$B \longrightarrow bZ$

$Z \rightarrow \epsilon$

19CS10039
Kaushal Banthia

3

(a) A universal Turing Machine U is considered, which has input as TM N and input ε. This simulates the case ~~...~~ DILEMMA$_{\varepsilon}$

For all the possible cases of transitions, if two or more possibilities occur (a dilemma occurs), then we can move to the accept state t and halt the machine. (This is checked from the input supplied to U about N. From that information, it ~~...~~ the no. of transitions can be determined). If this is not the case, the ~~...~~ it will loop indefinitely and thus be implicitly rejected. (without ever reaching the accept or reject states). Since it can go ~~...~~ into looping condition, it is ~~...~~ NOT RECURSIVE.

But since it can either accept or reject (implicitly), thus its Recursively Enumerable.

4 (a) Simulate G and check whether F is included in its set of languages.

By simulating, we mean to go through the sentential forms one by one, using the productions given in G.

(b) Similarly, we can simulate G to check if F is included in its set of languages

If F ~~...~~ $= L(G)$ then $L \neq \Sigma^* - F$.

If $F \neq L(G)$ then $L = \Sigma^* - F$.