TOC Quiz 2

Aritra
MCS20230[redacted]4

50/50

(3) definition of DPDA :-

DPDA is a M with 7 tuples

$M = (Q, \Sigma, \Gamma, \delta, q_0, Z, F)$

states, inp alph, stack alphabet, transition, starting state, bottom, set of final states

st $\delta(q, \epsilon, \gamma) \neq \phi$ $\forall a \in \Sigma$ $\delta(q, a, \gamma) = \phi$

$q \in Q$, $\gamma \in \Gamma$

and No such $q \in Q$ exists st

$\delta(q, a, \gamma)$ contains more than one element.

$a \in \Sigma \cup \{\epsilon\}$

$\gamma \in \Gamma$

Yes.

Let, $M = (Q, \Sigma, \Gamma, \delta, q_0, Z, F)$ be the DPDA for L

we will create $M'$ following way:

Intuition, whenever $w$ is accepted by L (st no proper prefix of $w$ say $y \notin L$) then all the strings of the form $ww'$ where $ww' \notin L$ will not belong to $L'$

$\therefore \{ w \notin L'$ st $\exists y \in \Sigma^*$, $y$ is proper pref of $w$

and $y \in L \}$

So, whenever we are in final state,

after reading input string $\alpha$ ~~we can~~ and some alphabets are still to process we can reject them as they shouldnot be in $L'$

So, $M' = (Q, \Sigma, \Gamma, \delta', q_0, Z, F)$ st

$$\boxed{\delta'(q_f, a, X) = \phi} \quad ①$$

st $q_f \in F$

$X \in \Gamma^*$

10

Proof: let, $ww' \in L$ st $w \in L$ and no proper prefix of $w \in L$

Now $(q_0, w, Z_0) \vdash^* (q_f, \epsilon, \gamma)$

$\gamma \in \Gamma^*$

But $(q_0, ww', Z_0) \vdash^* (q_f, w', \gamma)$ So, $ww'$ will not be accepted

and $\delta(q_f, a, X\gamma') = \phi$ from the construction ①

st $(X\gamma' = \gamma)$

$X \in \Gamma$

$\gamma' \in \Gamma^*$

Proved

(4) Subroutine for computing $x_{i+1}$ from $x_i$.

we will create a two way infinite TM for computing $x_{i+1}$ from $x_i$

At first we will have # 0 # as written on input tape of the subroutine and at each step it will be changed to

... B # 1 # B ... → ... B # 10 # B → B # 11 # B ...

The transition diagram is as follow

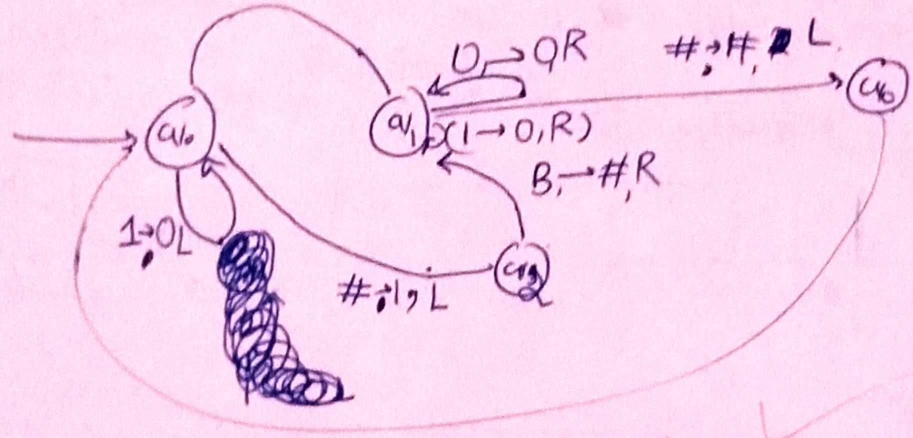
$\delta(q, a) = (P, b, D)$

we shown as

$a \to b, D$

Ⓠ Ⓟ

D is the direction enum of {L, R}




15

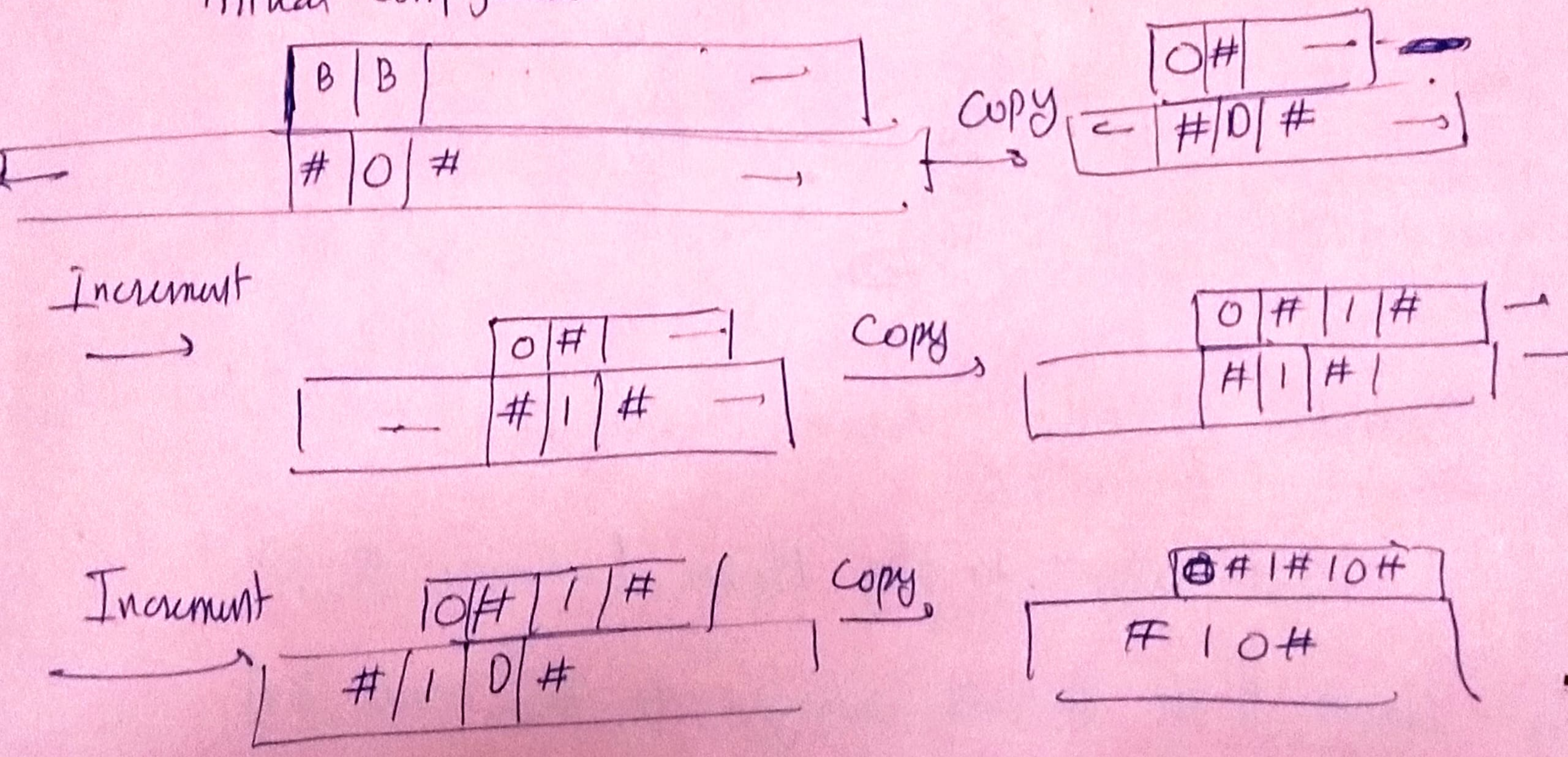
Description of M': whenever it sees 0, it will replace it by 1 and return to the Marker, whenever it gets 1 it converts it to 0 and carry 1 to the next bit

for the input. # 1111 # it carry 1 to the # marker and then replace # with 1 and replace the Blank with # marker

We will use the previous subroutine in this description.

We will be creating a TM with one semi-infinite and one infinite tape (Both side). The first tape (output tape) will contain the required output ($x_1 \# x_2 \# x_3 \ldots$) and the second tape we will use to call the subroutine that increase $x_i$ by 1. We will be assuming that there is a (copy) subroutine that allows to copy the $x_{i+1}$ to the output tape.

initial configuration



★ ①

Claim: At any moment stack can be represented as $\gamma w$ where $\gamma \in \Gamma^*$ and $w \in \Sigma^*$

Proof By induction on number of steps :-

Base case. $(q, w, S) \vdash^{0} (q, w, S)$ — start symbol of the grammar

true

Induction case: $(q, w\alpha w', S) \vdash^{*} (q, \alpha w', A\rho)$, $\rho \in \Sigma^*$

Now $A \rightarrow \alpha B, \beta$

So. $(q, w\alpha, S) \vdash^{*} (p, \alpha w', A\rho) \vdash (p, \alpha w', B\beta\rho)$

and $A \rightarrow \alpha$. $(q, w\alpha, S) \vdash^{*} (p, \alpha w', A\rho) \vdash (p, w', \rho)$

$\rho = \gamma\rho$

$\gamma = \epsilon$ and $\rho \in \Sigma^*$

Hence true

So, our PDA M will be such a PDA st

~~M = (Q, Σ, Γ,~~

$N(M) = \{ w \mid w \in L(G)$ and $(q, w, S) \vdash^{*} (q, \varepsilon, \varepsilon)\}$

and let $G = (V, T, P, S)$

So, ~~M = (Q, Σ, Γ, δ, q~~

$M = (\{q\}, T, (V \cup T), \delta, q, S, \phi)$

① High level intuition :-

for every production of the form

$$A \to \alpha B \beta$$

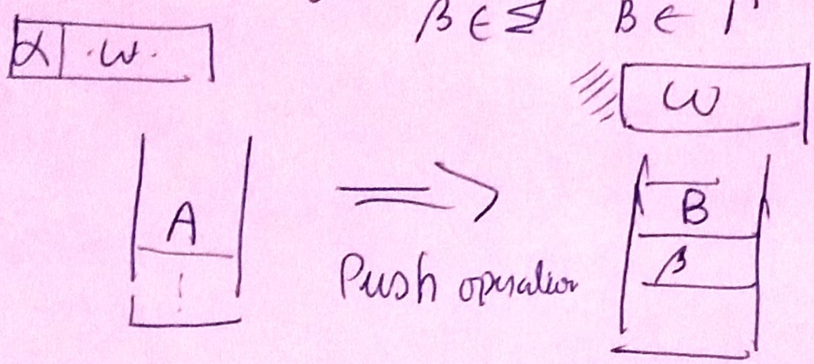
we will add following transition to our PDA

$$\delta(q, \alpha, A) \ni (P, B\beta)$$

$q \in Q$, $\alpha \in \Sigma$, $A \in \Gamma$

$\beta \in \Sigma$ $B \in \Gamma$

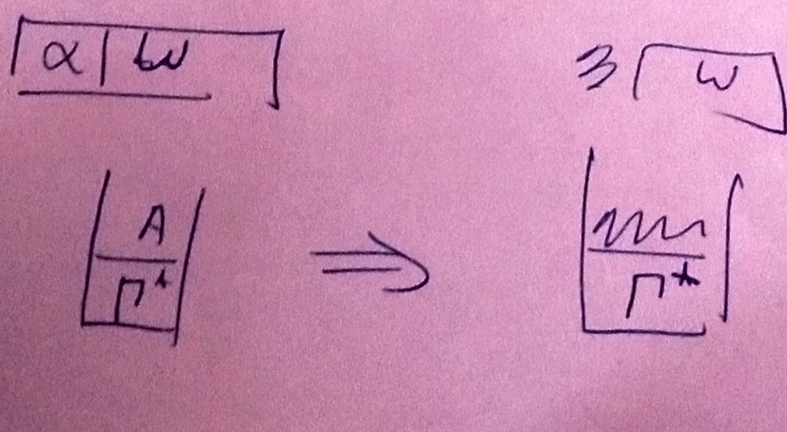
So,




and, every production of the form

$$A \to \alpha$$

$$\delta(q, \alpha, A) \ni (P, \varepsilon)$$



δ will be.

for every production

$$A \to \alpha B \beta$$

$$\delta(q, \alpha, A) \ni (q, B\beta)$$

and for all $A \to \alpha$

$$\delta(q, \alpha, A) \ni (q, A)$$

<u>and</u>, for all $t \in T$

$$\delta(q, t, t) \ni (q, \epsilon)$$

<u>Proof</u>: The <u>PDA will be one-turn</u>

<u>we will use claim ① to prove this</u>

According to the claim, stack can only one variable of the grammar at the top or no variable at all and once no variable is at the top, no way we can introduce a variable

So, whenever there is a variable. lets say

$A$ and $A \rightarrow ~~\gamma~~ \alpha B \beta$

stack top will be replaced by B and $\beta$ will be squeezed in between

So, push operation.

Now, $A \rightarrow \alpha$

will remove the only variable from top and push nothing (start of pop operation)

and so, no variable is in the stack

only transition are of form

$t \in T$.

15

$\delta(q, r, t) \ni (q, \epsilon)$

So, all pop operations till the stack being emptied (if $w \in L(G)$)

Hence proved

(2) L be a linear context free language and so we have a one-turn PDA $M$ for it.

R is a regular language and $\exists$ DFA $M'$ for it.

$$M = (Q, \Sigma, \Gamma, \delta, q_0, Z, \phi + \{q\})$$

$Q = \{q\}$

$$M' = (Q', \Sigma, \delta', q_0', F')$$

we will create an one turn PDA for it.

$$\delta(q, a, \gamma) \ni (q, \gamma')$$

$\gamma \in \Gamma$ $\qquad$ $\gamma' \in (\Sigma \cdot \Gamma) \cup (\Sigma) \cup (\epsilon)$

$q \in Q$

$q' \in Q'$

$p' \in Q'$

$\delta'(q', a) = (p')$

st $\delta_f([q, q'], a, \gamma) \ni ([q, p'], \gamma')$

10

$\{q\} \times F$

So, $M_f = (Q \times Q', \Sigma, \Gamma, \delta_f, \{q, q_0'\}, Z, )$

st $w \in L(M_f)$

$$([q, q_0], w, Z) \vdash^* ([q, q_f], \epsilon, S)$$

st $(q_f \in F)$

So, it will both be $L(M)$ and $N(M)$

Intuition: while simulating the one turn PDA, we will also simulate the DFA with it

for every $\delta(q, a, \gamma) \ni (q', \gamma')$

if $\delta'(q', a) = p'$

the following transition in new one turn PDA will be $\delta([q, q'], a) \ni$ ~~$(q', \gamma')$~~ $([q, p'], \gamma')$

It will only accept if for some word both stack goes empty and the DFA enters the final state.

Moreover, as we are not using $\varepsilon$ moves in PDA there is no case of non-determinism handling in the DFA part

So, each step, PDA will non-deterministically choose next step and DFA will keep simulating deterministically and accepts iff stack goes empty and reaches final