Date: __/__/____

HW6

1. $C \to ACA \mid E$
$E \to 0G1 \mid 1G0$
$G \to AGA \mid A \mid \varepsilon$
$A \to 0 \mid 1$

CFG to PDA

Transition rule for PDA

Each non-terminal

$\delta(q, \hat{}, A) = \{(q, \alpha) \mid A \to \alpha \text{ in prod}^n p\}$
$\therefore \hat{} \to \text{null}$

Each non-terminal $\{C, E, G, A\}$

$\delta(q, \hat{}, C) = \{(q, ACA), (q, E)\}$

$\delta(q, \hat{}, E) = \{(q, 0G1), (q, 1G0)\}$

$\delta(q, \hat{}, G) = \{(q, AGA), (q, A), (q, \varepsilon)\}$

$\delta(q, \hat{}, A) = \{(q, 0), (q, 1)\}$

For each terminal

$\delta(q, a, a) = (q, \hat{}) \mid q \in \Sigma$

given terminal : $\{0, 1\}$

$\delta(q, 0, 0) = \{(q, \hat{})\}$
$\delta(q, 1, 1) = \{(q, \hat{})\}$

Date: __/__/____

2. An unambiguous CFG is defined as a CFG for which all justifiable string has an individual leftmost derivation.

A DCFG is a proper subset of deterministic CFG's, it can be derived from deterministic finite automata and can be used to generate deterministic context free language.

DCFG's always show unambiguous behavior and an unambiguous CFG is on a super class of DCFG

To prove, for every pushed down automata M, there exists an equivalent context free grammar G.

So M recognizes L.$ → G, generates L$.
M deterministic → G is unambiguous. so if we replace $ by ε in G, G → G generates L.

Every DCFG is an unambiguous CFG.

~~R→ start state, T, W, Z→ final state, n=9~~

3

	R	S	T	U	V	W	X	Y	Z	
R	X	X	X	X	X	X	X	X	X	
S	X	X	X	X	X	X	X	X	X	
T	X	X	X	X	X	X	X	X	X	
U	X	X	X	X	X	X	X	X	X	2
V	X	.	X	X	X	X	X	X	X	
W	X	X	.	X	X	X	X	X	X	2
X	.	X	X	.	X	X	X	X	X	
Y	X	.	X	X	.	X	X	X	X	
Z	X	X	X	X	X	.	X	X	X	

Date: __/__/____

From the transition table

R → start state,

T, W, 2 → final state, n = 9

Explanation of the table.

1. Cross out all diagonal cells (R, R) and all the cells to the right

2. Consider final state T, W, 2.

Cross the cells where it is final - non-final
eg (R, T). (S, W)

3. for each unmarked cell, transition on 0, 1.

(S, V) : $(S, V)^0$ → T, W
$(S, V)^1$ → W, 2

if TW and W2 are unmarked, leave (S, V)

or if any of them are crossmarked, then cross (S, V).

4. Unmarked states : SV, TW, RX, UX, SY, VY, W2

RX, SV, SY, VY, UX, TW, W2

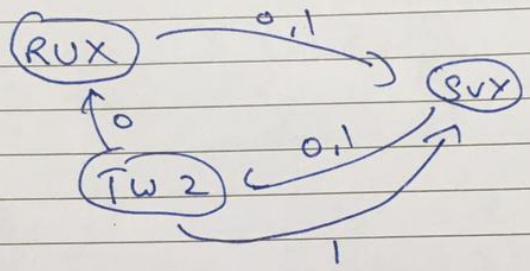
SV, SY and VY can be considered a single state
→ (SVY)

Date: __/__/____

RX, UX → (RUX)

TW, W2 → (TW2)

Now the DFA can be drawn.



The above is the minimum state equivalent DFA.

4. $L = \{ a^n b^n c^n \mid n \geq 0 \}$

This language will accept strings which have the same numbers of a, b, c in a continuous manner.

Our Method:

→ When 'a' is read, then 'x' is marked and we move right

→ When 'b' is read, we mark it as 'y' and move right.

→ when 'c' is read, we mark 'z' and keep moving

Date: __/__/____

left till 'x' is found

Repeat the steps till all a, b, c are marked

Transitions :

$\delta(q_0, a) = (q_1, x, R)$

$\delta(q_1, a) = (q_1, a, R)$
$\delta(q_1, y) = (q_1, y, R)$
$\delta(q_1, b) = (q_2, y, R)$

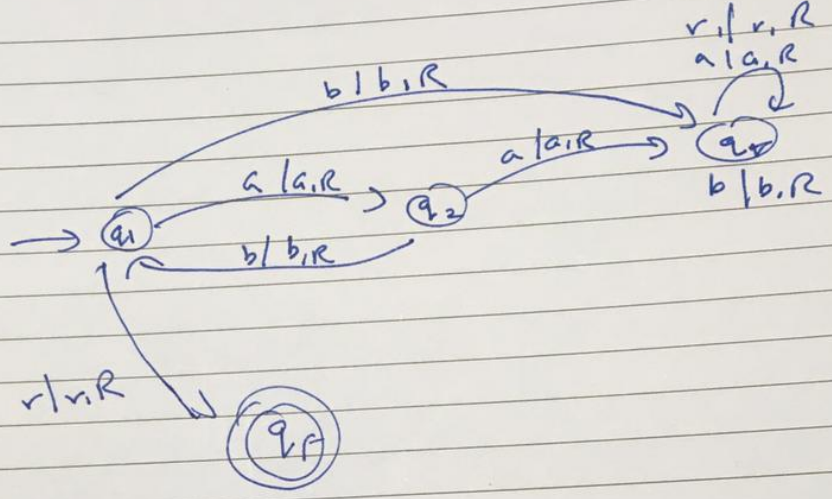
$\delta(q_2, b) = (q_2, b, R)$
$\delta(q_2, z) = (q_2, z, R)$
$\delta(q_2, c) = (q_3, z, L)$

$\delta(q_3, a) = (q_3, a, L)$
$\delta(q_3, b) = (q_3, b, L)$
$\delta(q_3, y) = (q_3, y, L)$
$\delta(q_3, z) = (q_3, z, L)$
$\delta(q_3, x) = (q_0, x, R)$

$\delta(q_0, B) = (q_0, B, S)$ -> for n = 0
$\delta(q_0, y) = (q_4, y, R)$
$\delta(q_4, y) = (q_4, y, R)$
$\delta(q_4, z) = (q_4, z, R)$
$\delta(q_0, B) = (q_0, B, S)$

Date: __/__/____

5



q_1 → initial state

q_r → infinite loop , q_f → final state

This behavior might be undesirable because when the expression enters an infinite loop, it may take a really long time to come to a conclusion on whether the string is accepted or rejected and computer resources may be used up for no outcome