CS 3350 Automata, Computability, and Formal Languages
Fall 2018, Test 2

Last 4 digits of your UTEP ID number: ___ ___________

General comments:

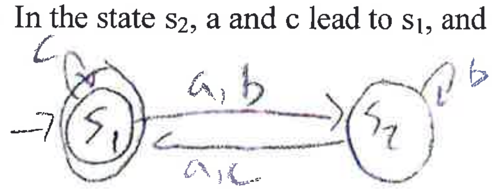
- you are allowed up to 5 pages of handwritten notes;
- if you need extra pages, place the last 4 digits of ID number on each extra page;
- the main goal of most questions is to show that you know the corresponding algorithms; so, if you are running of time, just follow the few first steps of the corresponding algorithm;
- each question will be graded on its own merit; so, for example, if when answering to the first part of the question, you got a wrong automaton, but on the second part, you correctly traced the new automaton, you will get full credit for the second part.

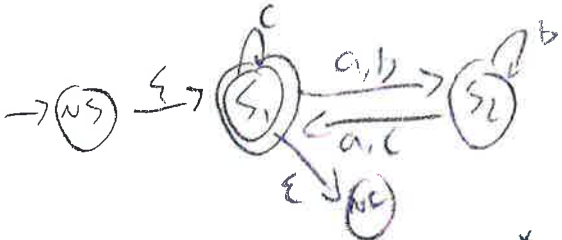
Good luck!

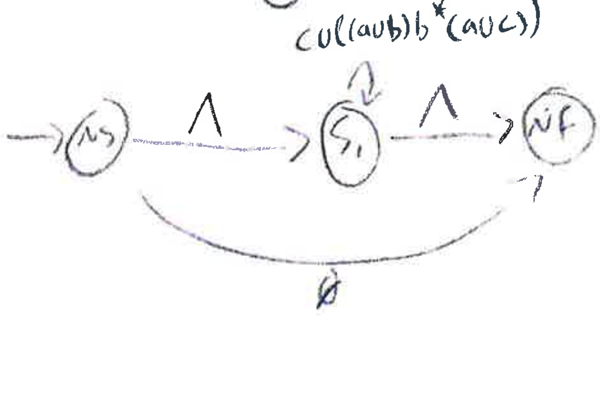
20|20

1-2. Use a general algorithm to transform the following finite automaton into the corresponding regular expression.

- This automaton has two states: a state s_1 which is a start state and a final state, and another state s_2.
- In the state s_1, a and b lead to s_2, and c leads to s_1.
- In the state s_2, a and c lead to s_1, and b leads to s_2.







$R_{NS1}' = R_{NS1} \cup (R_{NS2} R_{22}^* R_{21})$

$\Lambda \cup (\emptyset\ b^*(a \cup c))$

Λ

$R_{NSNF}' = R_{NSNF} \cup (R_{NS2} R_{22}^* R_{2NF})$

$\emptyset \cup (\emptyset \quad)$

$\emptyset$

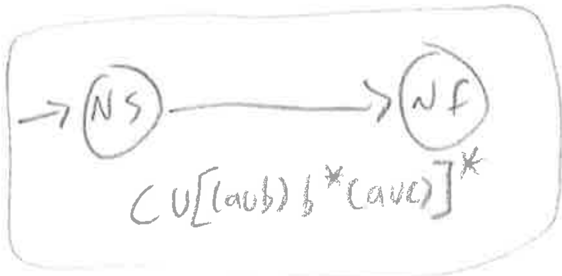
$R_{11}' = R_{11} \cup (R_{12} R_{22}^* R_{21})$

$(c \cup ((a \cup b) b^* (a \cup c))$

$R_{1NF}' = R_{1NF} \cup (R_{12} R_{22}^* R_{2NF})$

$\Lambda \cup (\ldots \emptyset)$

Λ



$R'_{NSNF} = R_{NSNF} \cup (R_{NS1} R_{11}^* R_{1NF})$

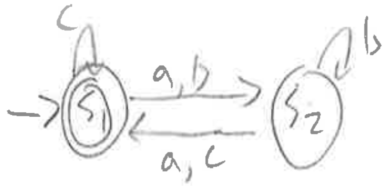
$\emptyset \cup (\Lambda [\epsilon \cup ((a \cup b) b^* (a \cup c)]^* \Lambda)^*$

$[\epsilon \cup [(a \cup b) b^* (a \cup c)]]^*$

3. Automaton from Problem 1-2 accepts the word abc.

10/10

- Trace, step-by-step, that this word is indeed accepted by the given automaton.
- Use this tracing to find the parts x, y, and z of this word corresponding to the Pumping Lemma.
- Trace that the words xyyz and xyyyz are also accepted by this automaton.



| a | b | c |

s_1 s_2 s_2 s_1 ✓

x y z

x = a

y = b

z = c

xyyz = | a | b | b | c |

s_1 s_2 s_2 s_2 s_1 ✓

xyyyz = | a | b | b | b | c |

s_1 s_2 s_2 s_2 s_2 s_1 ✓

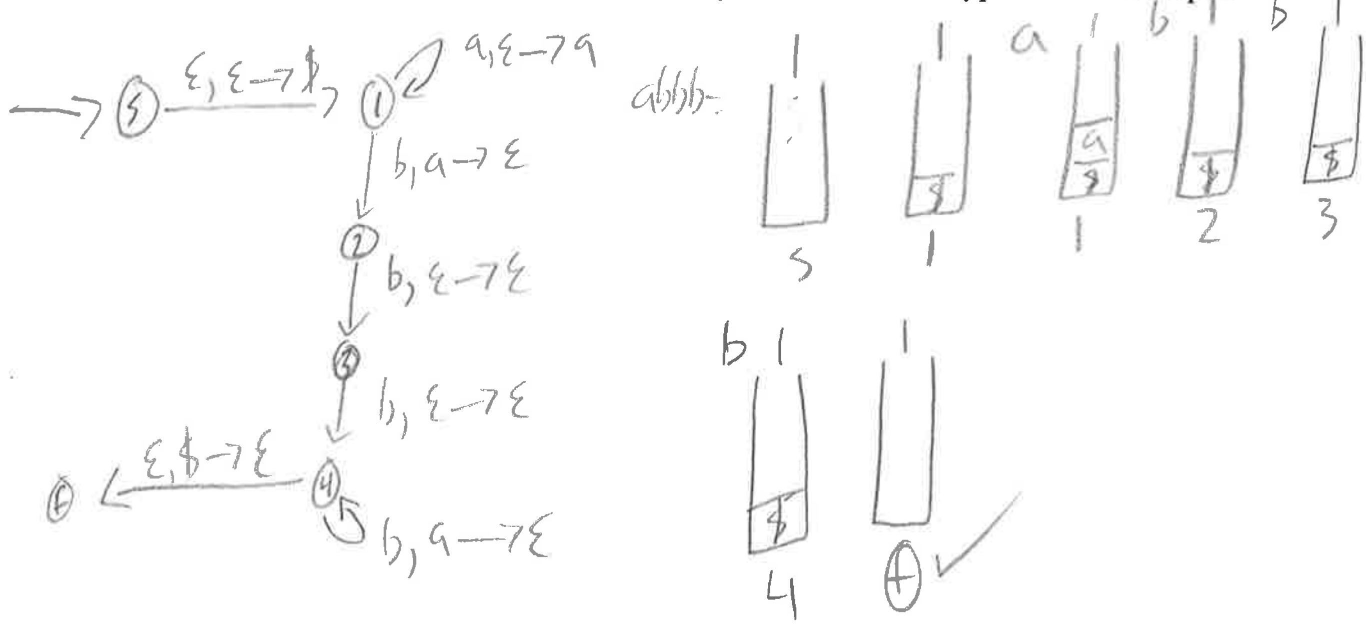
10/10

4. Prove that the language $\{a^n b^{2n}\} = \{\Lambda, abb, aabbbb, aaabbbbbb, ...\}$ is not regular.

LET'S ASSUME $L = \{a^N b^{2N}\}$ IS A RL (REGULAR LANGUAGE) & LET'S DERIVE A CONTRADICTION FROM THIS ASSUMPTION. SINCE L IS A RL BY PL (PUMPING LEMMA), THERE EXISTS AN INTEGER 'P' SUCH THAT EVERY WORD 'W' FROM L WHOSE $len(w) \geq P$ CAN BE REPRESENTED AS 'XYZ' WHERE $LEN(Y) > 0$, $LEN(XY) \leq P$, & $\forall i (xy^i z \in L)$. LET'S TAKE $w = a^P b^{2P}$ $(\underbrace{a..a}_{P}\underbrace{b...b}_{2P})$. $LEN(XY) \leq P$, $W = XYZ$ STARTS WITH 'XY' SO 'Y' IS IN a's, IN XYYZ, WE ADD a's, BUT NOT b's. WE STARTED WITH A WORD THAT HAS TWICE AS MANY b's THAN a's, BUT $XYYZ \notin L$, BUT BY PL $XYYZ \in L$. WE GET A CONTRADICTION. SO OUR ASSUMPTION THAT L IS A RL IS FALSE. THUS, L IS NOT REGULAR

10/10

5. Design a pushdown automaton that would recognize the words of the type $a^n b^{n+2}$, i.e., the language L = {bb, abbb, aabbbb, ...}. Show, step by step, how your pushdown automaton will recognize the word abbb. *Hint*: use a pushdown automaton for recognizing the words of the type $a^n b^n$ as a sample.



10/10

6. Design a context-free grammar that would generate all the words of the type a^nb^{n+2}, i.e., the language L = {bb, abbb, aabbbb, ...}. Show, step by step, how your grammar will generate the word abbb. *Hint*: use a context-free grammar for generating the words of the type a^nb^n as a sample. There, we had rules $S \rightarrow \varepsilon$ and $S \rightarrow aSb$, now a slight modification is needed.

$S \rightarrow bb$

$S \rightarrow aSb$

OR

$S \rightarrow \varepsilon$

$S \rightarrow bSb$

$S \rightarrow aSb$

$S \rightarrow aSb \mid S \rightarrow abbb$

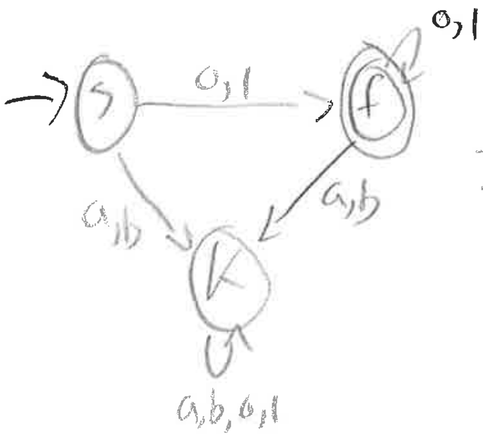
$S \rightarrow aSb \mid S \rightarrow abSbb \mid S \rightarrow abbb$

10/10

7. Use a general algorithm that we had in class -- for transforming a finite automaton into a context-free grammar -- to generate a context-free grammar that corresponds to the following finite automaton for recognizing unsigned integers. For simplicity, assume that we only allow letters a and b and digits 0 and 1. This automaton has three states: the starting state s, the final state f, and the sink state k.

- From s, any letter (a or b) lead to k, while any digit leads to f.
- From f, any digit leads to f, and letter leads to k.

Show, step by step, how your grammar generates a word 010.



$\Rightarrow$

$S \to 0F$
$S \to 1F$
$S \to ak$
$S \to bk$

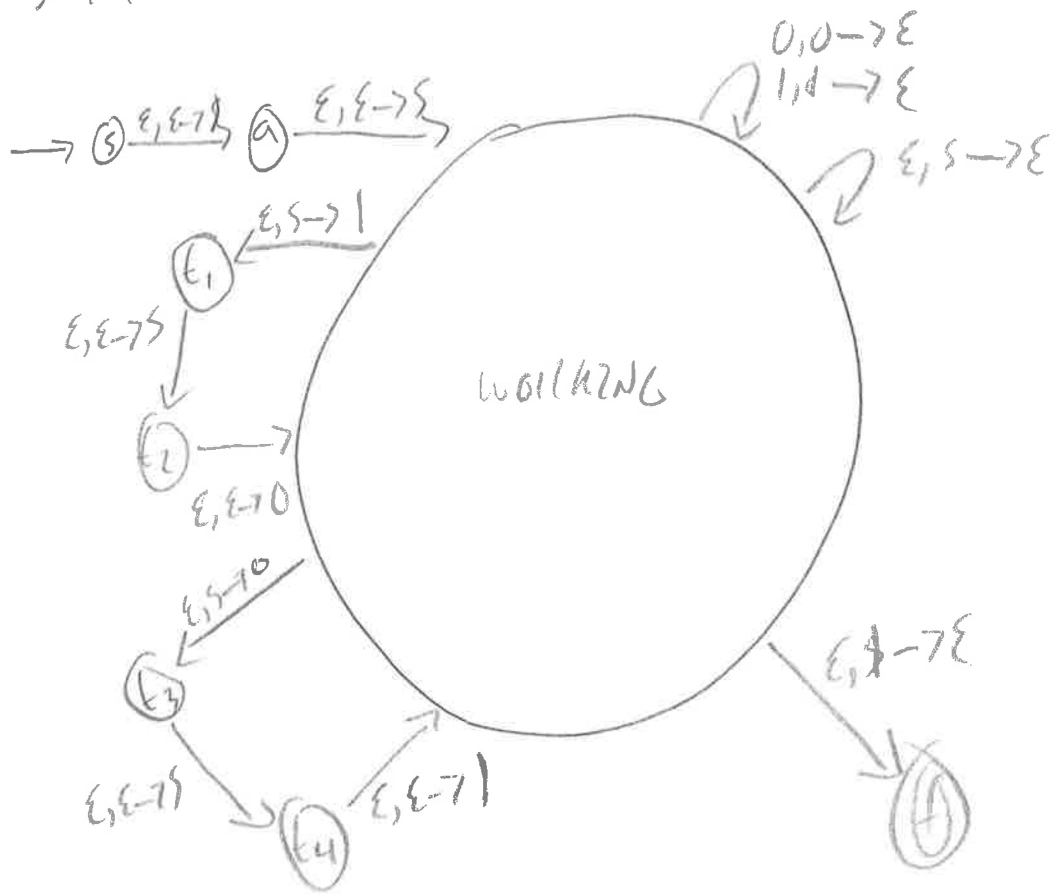
$K \to ak$
$K \to bk$
$K \to 0k$
$K \to 1k$

$F \to 0F$
$F \to 1F$
$F \to ak$
$F \to bk$
$F \to \Lambda$

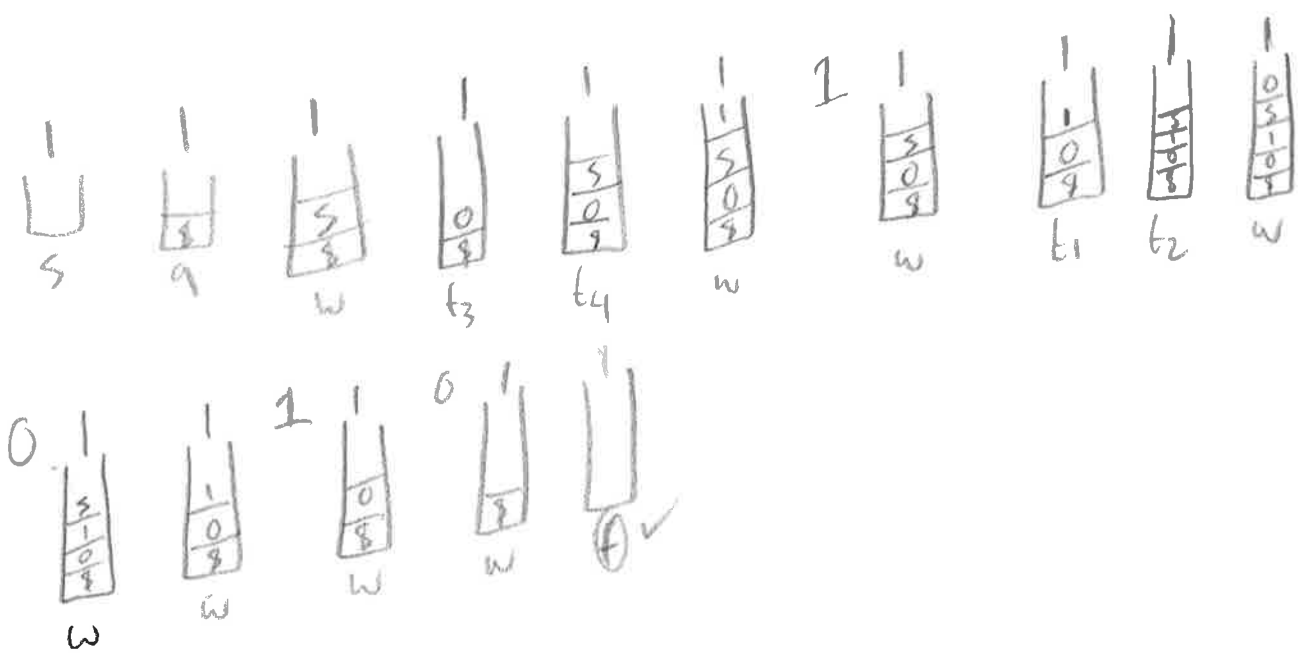
10/10

8. Use a general algorithm for transforming CFG into PDA to design a pushdown automaton which is equivalent to the the grammar with rules S → ε, S → 0S1, and S → 1S0. Show, step by step, how the word 1010 will be accepted by the resulting pushdown automaton.

$S \to \varepsilon$
$S \to 0S1$
$S \to 1S0$



1010:



9/10

9. (For extra credit) Prove that the following language is not context-free: $\{a^{2n}b^nc^{2n}\} = \{\Lambda$, aabcc, aaaabbcccc, aaaaaabbbcccccc, ...$\}$.

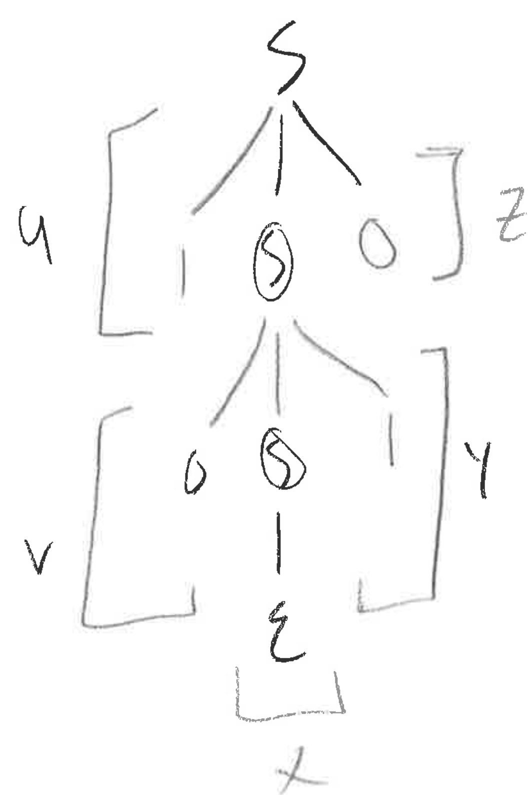
LET'S ASSUME L IS CFG. LET'S TAKE $w = a^{2P}b^Pc^{2P} = \underbrace{a..a}_{2P}\underbrace{b..b}_{P}\underbrace{c..c}_{2P}$ WHERE $len(w) = \cancel{X_P}\,5P \geq P$. THUS $\exists u,v,x,y,z$, $w = uvxyz$ & $len(vxy) < P$ & $len(vy) > 0$ & $uvvxyyz \in L$ BY PL. WE CAN'T HAVE 'VXY' COVERING a's, b's, & c's SINCE $len(vxy) \geq P$. SO WE HAVE THE CASES:

1. 'VXY' IS IN b's. WHEN WE PUMP, WE ADD b's, BUT NOT a's & c's
2. 'VXY' IS IN SOME COMBINATION OF a's & b's WHERE $len(vxy) < P$. WHEN WE PUMP, WE PUMP a's & b's, BUT NOT c's
3. 'VXY' IS IN SOME COMBINATION OF b's & c's WHERE $len(vxy) < P$. WHEN WE PUMP, WE PUMP b's & c's, BUT NOT a's.
4. 'VXY' IS IN a's SUCH THAT $len(vxy) < P$. WHEN WE PUMP, WE PUMP a's, BUT NOT b's & c's.
5. 'VXY' IS IN c's SUCH THAT $len(vxy) < P$. WHEN WE PUMP, WE PUMP c's, BUT NOT a's OR b's.

IN ALL POSSIBLE CASES, WE GET A CONTRADICTION, SO L IS NOT CFG.

10/10

10. (For extra credit) On the example of the word 1010 generated by a grammar from Problem 8, show how this word will be represented as uvxyz according to the Pumping Lemma. Show how the words uxz and uvvxyyz will be generated by this grammar.



u = 1

v = 0

x = Λ

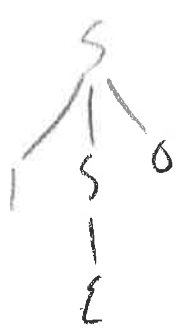
y = 1

z = 0

uxz = 10

uvvxyyz = 100110

uxz =



uvvxyyz =