CENG 213 Theory Of Computation

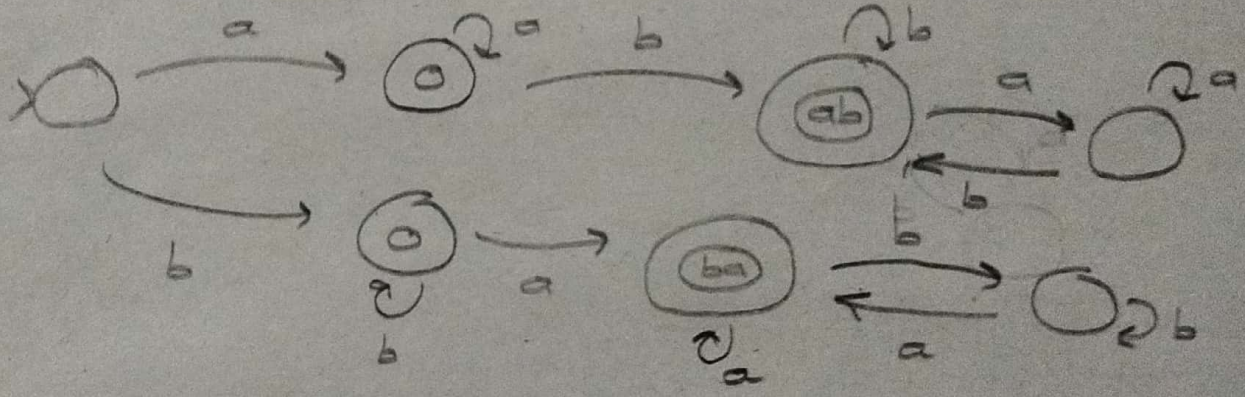
ASSIGMENT 1#

Name: Gökay
Surname: Gülsoy
Student ID: 270201072

Question - 1)

Solution:

part a) DFA that recognizes L is as in the below



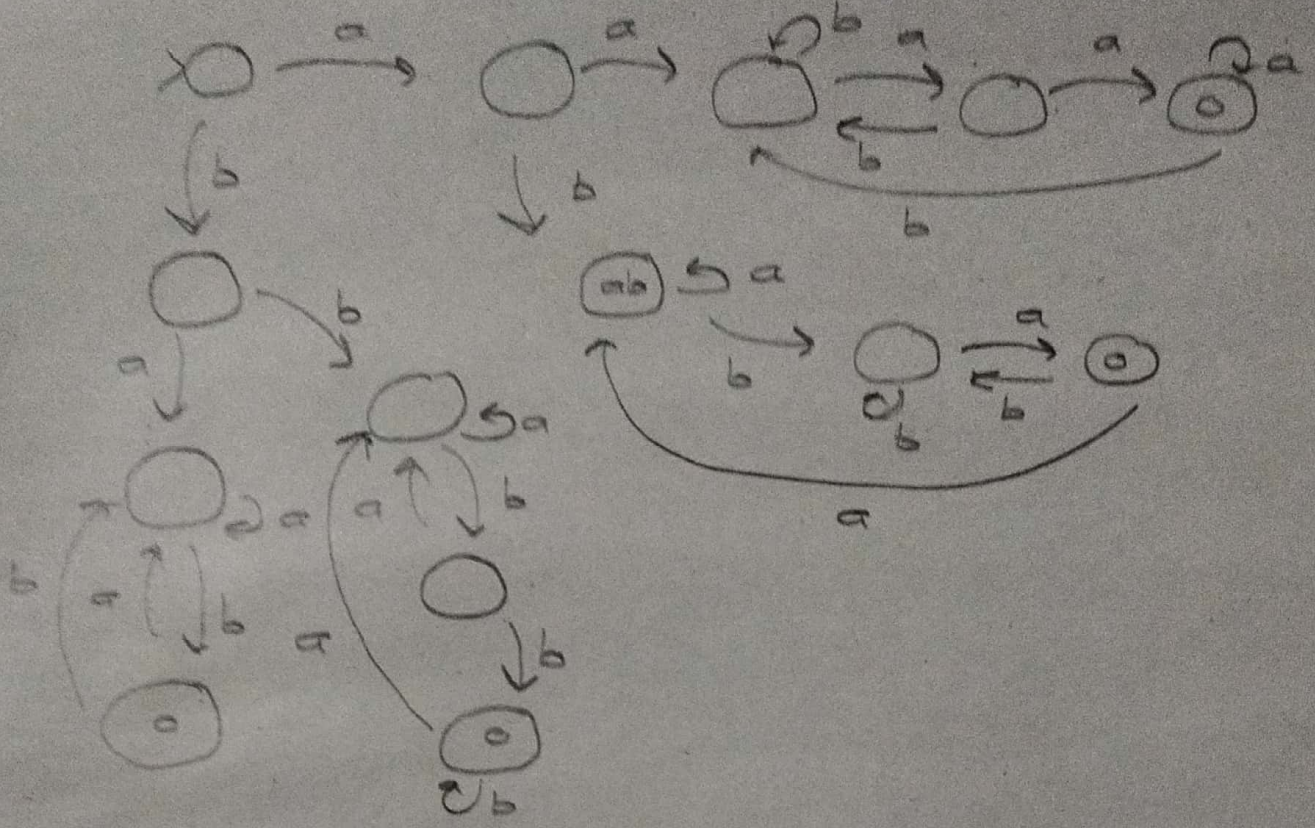
part b) equivalent regular expression For DFA is

$$(aa^*b\,[b^*aa^*bb^*]^*) \cup (bb^*a\,[a^*bb^*aa^*]^*)$$

For Example:

String abab will be accepted by our DFA but String aba will not be accepted by DFA.

Question 2-)



part a) above is the DFA that recognizes L

For example:

String aababaa will be accepted by DFA, but string abbabbab will not be accepted by DFA.

part b) Python code to decide whether the given String is in L

```
import re
# we assume that variable w holds the String
# checking given palindrome situations
begin1 = re.Findall("^aa", w)
end1 = re.Findall("aa$", w)

begin2 = re.Findall("^ab", w)
end2 = re.Findall("ab$", w)
```

continuing on next page =>

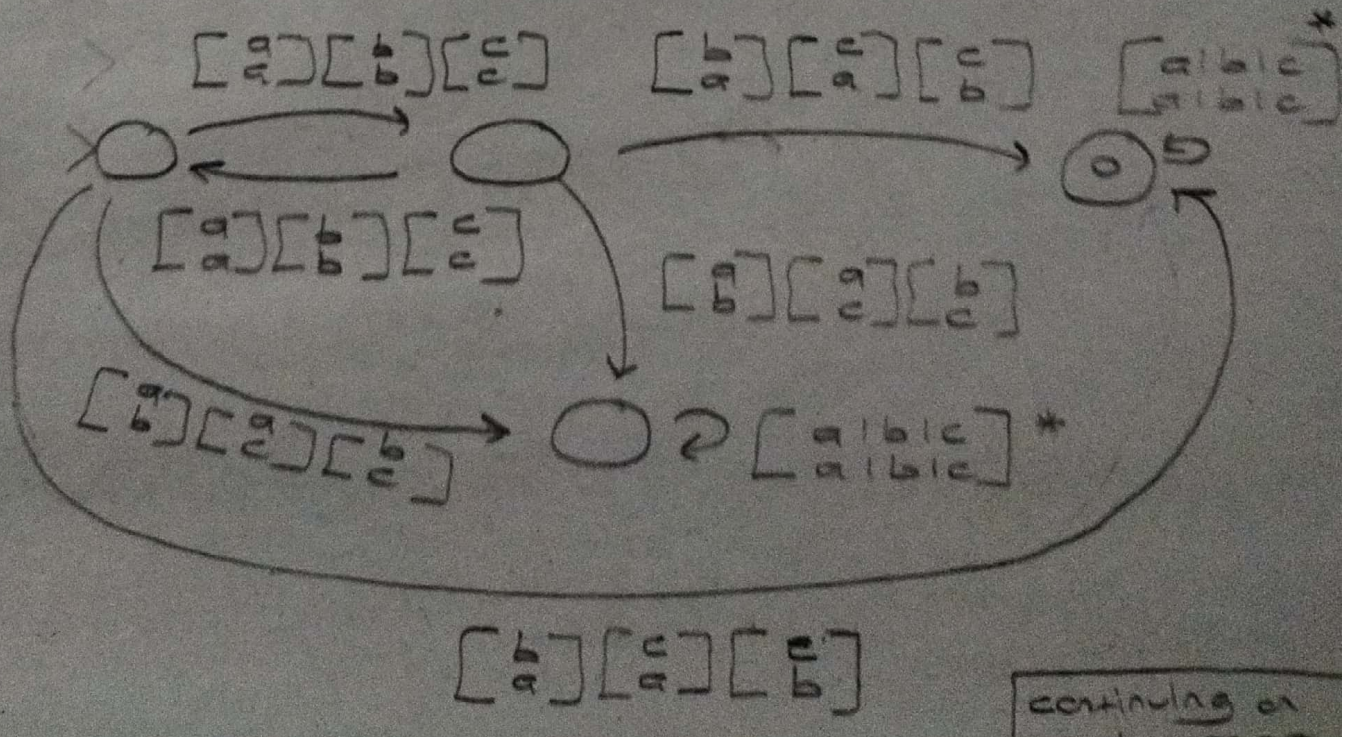
```
begin3 = re.Findall("^ba", w)
end3 = re.Findall("ab$", w)

begin4 = re.Findall("^bb", w)
end4 = re.Findall("bb$", w)

# Using conditionals to check whether the language
definition is satisfied or not

IF (begin1 and end1):
    print("The string " + w + " is in L")
elif (begin2 and end2):
    print("The string " + w + " is in L")
elif (begin3 and end3):
    print("The string " + w + " is in L")
elif (begin4 and end4):
    print("The string " + w + " is in L")
else:
    print("The string " + w + " is not in L")
```

Question - 3)



continuing on next page

above is the DFA that recognizes L

$\begin{bmatrix} a|b|c \\ a|b|c \end{bmatrix}^*$ means all possible columns of a's b's and c's, so as a convention in above DFA instead of writing all possible 9 columns I used $\begin{bmatrix} a|b|c \\ a|b|c \end{bmatrix}^*$ to denote all possibilities after reaching final state or dead state.

part b) Equivalent regular expression is

$$\left[\left(\begin{bmatrix} a \\ a \end{bmatrix} \cup \begin{bmatrix} b \\ b \end{bmatrix} \cup \begin{bmatrix} c \\ c \end{bmatrix} \right)^* \left(\begin{bmatrix} b \\ a \end{bmatrix} \cup \begin{bmatrix} c \\ a \end{bmatrix} \cup \begin{bmatrix} c \\ b \end{bmatrix} \right) \left(\begin{bmatrix} a|b|c \\ a|b|c \end{bmatrix}^* \right) \right]$$

$$\cup \left[\left(\begin{bmatrix} b \\ a \end{bmatrix} \cup \begin{bmatrix} c \\ a \end{bmatrix} \cup \begin{bmatrix} c \\ b \end{bmatrix} \right) \left(\begin{bmatrix} a|b|c \\ a|b|c \end{bmatrix} \right)^* \right]$$

For example:

$$\begin{bmatrix} a \\ a \end{bmatrix} \begin{bmatrix} b \\ b \end{bmatrix} \begin{bmatrix} c \\ a \end{bmatrix} \begin{bmatrix} c \\ c \end{bmatrix} \in L$$

because (abcc > abaa)

and if we check also DFA accepts it and it can be generated by regular expression.

Question - 4) Regular expression for the given language is as follows

$$\left(\left[a|b|c \right) a \right] \cup \left[b|a|c \right) \right] \cup \left[c|a|b \right) \right] (a \cup b)^*$$

$$\left(\left[ab|a|c \right) \right] \cup \left[ba(a|c) \right] \cup \left[|a|c)ba \right] \cup \left[|a|c)ab \right] \right)^*$$

For example:

If we check string ababbacb can be generated by given regular expression but strings abccba, cbbbbab, and abaaaac cannot be generated by regular expression because they do not satisfy language definition.

part b-)

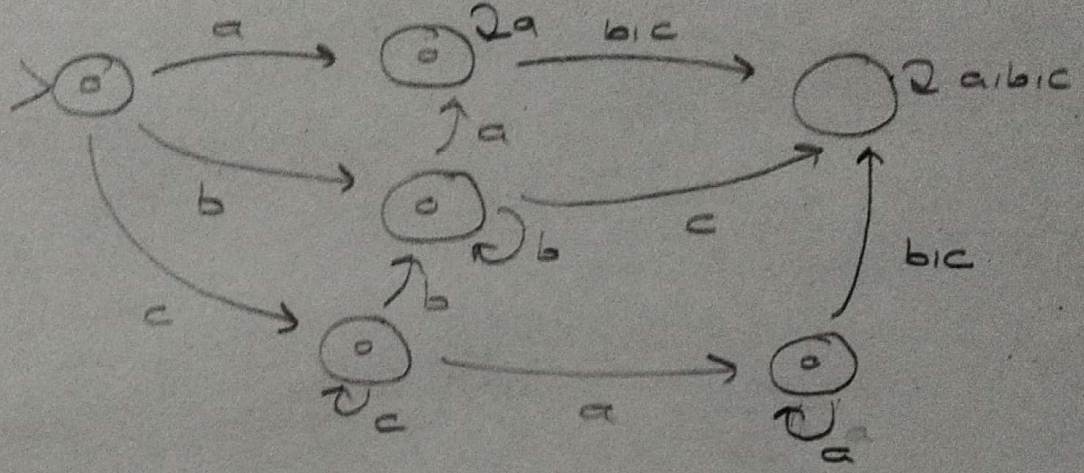
```
# w is the variable that holds the string
substring1 = "aaa"
substring2 = "bbb"
substring3 = "cc"
# Using conditionals to check whether string contains substrings
if (substring1 in w):
    print("The string " + w + " is not in L")
elif (substring2 in w):
    print("The string " + w + " is not in L")
elif (substring3 in w):
    print("The string " + w + " is not in L")
else:
    print("The string " + w + " is in L")
```

Question-5) Regular expression for the language is part a-)

$$a^* \cup (bb^*aa^*) \cup [c[(bb^*aa^*) \cup (aa^*)]$$

For example: strings baabc, abc, and bbcab cannot be generated by regular expression, but bbbac can be generated by regular expression.

part-b) DFA for the given regular expression is as follows



Question-6)

part-a) $R = \{(a,b), (b,c), (c,d), (b,a)\}$

relation on $\{a,b,c,d\}$

we know that $R^+ = R \cup R^2 \cup R^3 \ldots$

so we will firstly find R^2

$R^2 = R \circ R$ $\quad R^2 = \{(a,c), (a,a), (b,d), (b,b)\}$

$R^3 = R^2 \circ R$ $\quad R^3 = \{(a,d), (a,b), (b,a), (b,c)\}$

$R^4 = R^3 \circ R$ $\quad R^4 = \{(a,a), (a,c), (b,d), (b,b)\}$

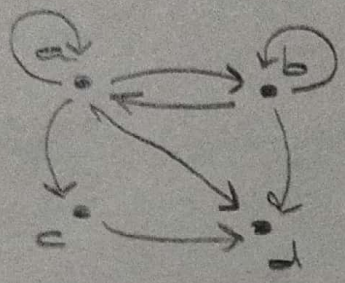
after R^4 as to terms will repeat each other we can finitely represent R^+ as follows

$R^+ = R \cup R^2 \cup R^4$

continuing on next page

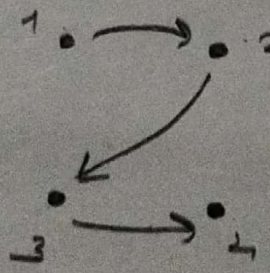
$R^+ = \{ (a,a), (a,c), (a,b), (a,d), (b,a), (b,b), (b,c), (b,d), (c,d) \}$

also graphical representation of it is as follows



part-b) let $R = \{(1,2), (2,3), (3,4)\}$ be a relation on $\{1,2,3,4\}$

by definition of transitive closure the must be a path from a to c if there is a path from a to some b and there is a path from b to c.

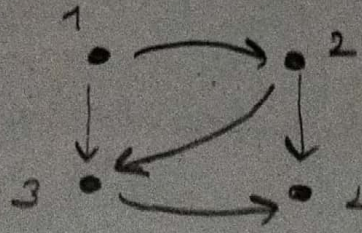


$(1,2), (2,3) \longrightarrow (1,3)$

path path must be a path

$(2,3), (2,4) \longrightarrow (2,4)$

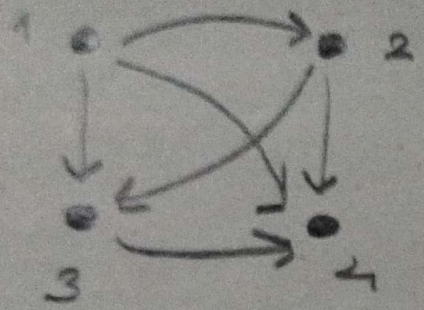
path path must be a path



continuing on next page

(1,2) , (2,4) → (1,4)

path path must be a path



$R^+ = R \cup R^2 \cup R^3$