29/11/23

EXP 2 : Conversion from RE to NFA

AIM: To write a program for converting regular expression to NFA

ALGORITHM:

1. Start
2. Get the input from the user
3. Initialize separate variables & functions for postfix.
4. Create separate methods for different operators like +, *, ..
5. By using switch case initialize different cases for the input.
6. For '.' operator initialize a separate method by using various stack functions, do the same for the other operators like '*' and '+'.
7. RE is in the form like a.b (or) a+b.
8. Display the output
9. Stop.

PROGRAM:

```
transition_table = [[0]*3 for in range(20)]
re = input("Enter the regular expressions:")
re = re + " "
i = 0
j = 1
while (i < len(re)):
if re[i] == 'a'
try:
if re[i+1] != '|' and re[i+1] != '*':
transition_table[i][0] = j+1
j += 1
elif re[i+1] == '|' and re[i+2] == 'b':
transition_table[j][2] = ((j+1)*10) + (j+3)
i += 1
```

```
transition_table[j][0] = j+1
    j+=1
transition_table[j][2] = j+3
    j+=1
transition_table[j+1] = j+1
    j+=1
transition_table[j][2] = j+1
    j+=1
    i = i+2
elif re[i+1] == '*':
transition_table[j][2] = ((j+1)*10 + (j+3)
    j+=1
transition_table[j][0] = j+1
    j+=1
transition_table[j][2] = ((y+1)*10) + (j-1)
    j+=1
except:
transition_table[j][0] = j+1
elif re[i] = 'b':
try:
if re[i+1] != '|' and re[i+1] != '+':
transition_table[j][2] = ((j+1)*10) + (j+3)
    j+=1
transition_table[j][1] = j+1
    j+=1
transition_table[j][2] = j+3
    j+=1
transition_table[j][0] = j+1
    j+=1
transition_table[j][2] = j+1
```

```
        jt = 1
        i = i + 2
    elif re[i+1] == '*':
        transition_table[j][2] = ((j+1) * 10) + (j+3)
        jt = 1
    transition_table[j][i] = j+1
        jt = 1
    transition_table[j][2] = ((j+1) * 10) + (j-1)
        jt = 1
    except:
    transition_table[j][i] = j+1
    elif re[i] == 'e' and re[i+1] != '1' and
            re[i+1] != '*':
    transition_table[j][2] = j+1
        jt = 1
    elif re[i] == ) and re[i+1] == '+':
        transition_table[0][2] = ((j+1) * 10) + 1
        transition_table[j][2] = ((j+1) * 10) + 1
            jt = 1
        it = 1
    print ("Transition function")
    for i in range(j):
        if (transition_table[i][0] != 0):
```

Start with a and with ba

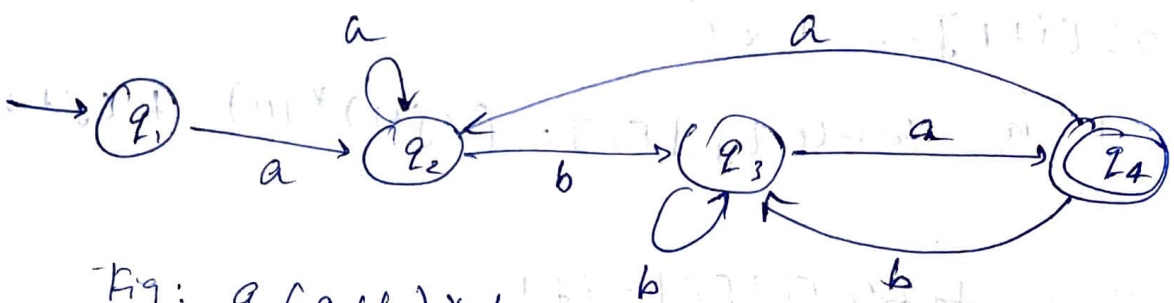


Fig: a(a+b)*ba

Given regular expression : a abba

Transition table

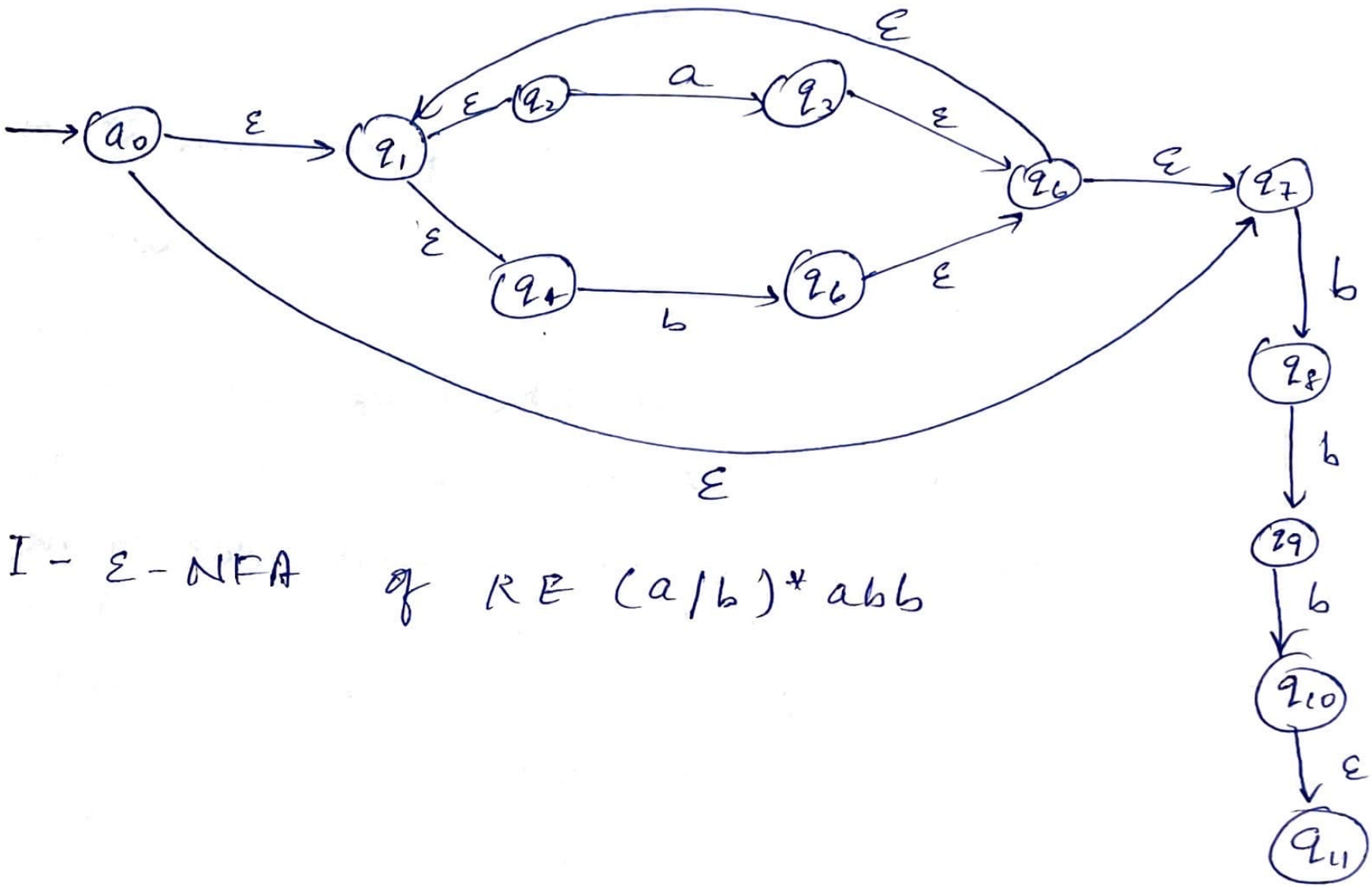
Current state	Input	Next state
q[1]	|a|	q[2]
q[2]	|a|	q[2]
q[3]	|b|	q[3]
q[4]	|b|	q[3]
q[4]	|a|	q[2]

OUTPUT:

Enter the regular expression (a/b)* abb

transition function:

Current state	Input	Next state
q[0]	|e|	q[7], q[1]
q[1]	|e|	q[2], q[4]
q[2]	|a|	q[3]
q[3]	|e|	q[6]
q[4]	|b|	q[5]
q[5]	|e|	q[6]
q[6]	|e|	q[7], q[1]
q[7]	|a|	q[8]
q[8]	|b|	q[9]
q[9]	|b|	q[10]



I - ε-NFA of RE $(a/b)^* abb$