Compiler Design Lab WEEK 3

Experiment:

Implementation of NFA to DFA

Aim:

To implement the conversion of NFA to DFA.

Algorithm:

Suppose there is an NFA N < Q, ∑, q0, δ, F > which recognizes a language L. Then the DFA D < Q’, ∑, q0, δ’, F’ > can be constructed for language L as:

1. Initially Q’ = ɸ.
2. Add q0 to Q’.
3. For each state in Q’, find the possible set of states for each input symbol using transition function of NFA. If this set of states is not in Q’, add it to Q’.
4. Final state of DFA will be all states with contain F (final states of NFA)

**Code**

import pandas as pd

nfa = {}

n = int(input("No. of states : "))

t = int(input("No. of transitions : "))

for i in range(n):

state = input("state name : ")

nfa[state] = {}

for j in range(t):

path = input("path : ")

print("Enter end state from state {} travelling through path {} : ".format(state, path))

reaching\_state = [x for x in input().split()]

nfa[state][path] = reaching\_state

print("\nNFA :- \n")

print(nfa)

print("\nPrinting NFA table :- ")

nfa\_table = pd.DataFrame(nfa)

print(nfa\_table.transpose())

print("Enter final state of NFA : ")

nfa\_final\_state = [x for x in input().split()]

new\_states\_list = []

#-------------------------------------------------

dfa = {}

keys\_list = list(

list(nfa.keys())[0])

path\_list = list(nfa[keys\_list[0]].keys())

dfa[keys\_list[0]] = {}

for y in range(t):

var = "".join(nfa[keys\_list[0]][

path\_list[y]])

dfa[keys\_list[0]][path\_list[y]] = var

if var not in keys\_list:

new\_states\_list.append(var)

keys\_list.append(var)

while len(new\_states\_list) != 0:

dfa[new\_states\_list[0]] = {}

for \_ in range(len(new\_states\_list[0])):

for i in range(len(path\_list)):

temp = []

for j in range(len(new\_states\_list[0])):

temp += nfa[new\_states\_list[0][j]][path\_list[i]]

s = ""

s = s.join(temp)

if s not in keys\_list:

new\_states\_list.append(s)

keys\_list.append(s)

dfa[new\_states\_list[0]][path\_list[i]] = s

new\_states\_list.remove(new\_states\_list[0])

print("\nDFA :- \n")

print(dfa)

print("\nPrinting DFA table :- ")

dfa\_table = pd.DataFrame(dfa)

print(dfa\_table.transpose())

dfa\_states\_list = list(dfa.keys())

dfa\_final\_states = []

for x in dfa\_states\_list:

for i in x:

if i in nfa\_final\_state:

dfa\_final\_states.append(x)

break

print("\nFinal states of the DFA are : ", dfa\_final\_states)

**OUTPUT**



