EXPERIMENT - VI
NFA TO DFA CONVERSION

September 13, 2020

ADITHYA D RAJAGOPAL
ROLL NO : 9
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
COLLEGE OF ENGINEERING TRIVANDRUM

AIM

To write a program to convert NFA to DFA.

THEORY

Deterministic Finite Automata

A deterministic finite automata (DFA) N over an alphabet set Σ is a structure of the form:

$$N = (Q, s, \delta, F)$$

where,

- Q is a finite set of states.
- s ∈ Q is the start state.
- Δ : Q X Σ → Q is the transition function.
- F ⊆ Q is the set of final states.

Non-Deterministic Finite Automata

A non-deterministic finite automata (NFA) M over an alphabet set Σ is a structure of the form:

$$M = (Q, S, \Delta, F)$$

where,

- Q is a set of finite states.
- S ⊆ Q is the set of start states.
- Δ : Q X Σ → 2^Q is the transition function.
- F ⊆ Q is the set of final states.

Conversion of NFA to DFA

Let M=(Q_M,S_M,Δ_M,F_M) be an NFA over the alphabet set Σ. Then the equivalent DFA N=(Q_N,s_N,δ_N,F_N) over the same alphabet set Σ can be defined as follows :

- $Q_N = 2^{Q_M}$
- $s_N = S_M$
- δ_N(A,b) = $\bigcup_{q \in A}$ (Δ(q,b))
- F_N = { A $\subseteq Q_M$ | A $\cap$ $F_M \neq \Phi$ }

ALGORITHM

Algorithm 1 Algorithm to Convert NFA to DFA

```
Start
Input the NFA (nfa).
Initialize an empty object of type DFA (dfa).
dfa.noofalphabets = nfa.noofalphabets
i=0
Initialize a set lazySet which stores subsets of Q and store 0 in it.
Create a new row of size dfa.noofalphabets and insert into dfa.table.
Initialize all values of the row to -1.
while i < lazySet.size() do
    for each input symbol j in Σ do
        Initialize an empty set of states reachable.
        Initialize next = -1.
        for each element in lazySet do
            Push into reachable the set nfa.table[i][j].
        end for
        if reachable is already in lazySet then
            Get the value of next from lazySet.
        else
            Insert into lazySet the set reachable.
            Set next = lazySet.size().
            if any element in reachable is a final state of nfa then
                Insert next into dfa.finalstates.
            end if
            Create a new row of size dfa.noofalphabets and insert into dfa.table.
            Initialize all values of the row to -1.
            dfa.table[i][j]=next
        end if
    end for
    Increment i.
end while
Stop
```

SOURCE CODE

```
def ConvertToDFA(nfa):
        print()
        print("Generating DFA...")
        delta=[]
        S=nfa[1]
        F=[]
        lazySet=[]
        lazySet.append(S)
        tt=nfa[2]
        i=0
        while i<len(lazySet):
                row=[]
                for j in range(len(sigma)):
                        reachable=[]
                        next=-1
                        for k in lazySet[i]:
                                states=tt[k][j].split(",")
                                for x in states:
                                        y=int(x)
                                        if y not in reachable:
                                                reachable.append(y)
                        reachable.sort()
                        if reachable not in lazySet:
                                lazySet.append(reachable)
                                next=len(lazySet)
                                row.append(next-1)
                        else:
                                next=lazySet.index(reachable)
                                row.append(next)
                delta.append(row)
                i=i+1
        print()
        print("Start state of DFA : 0")
```

```
        print()
        print("STATE",end="\t")
        for i in range(s-1):
                print(sigma[i],end="\t")
        print(sigma[s-1])
        for i in range(len(lazySet)):
                print(i,end="\t")
                for j in range(s-1):
                        print(delta[i][j],end="\t")
                print(delta[i][s-1])
        print()
        for x in lazySet:
                for y in x:
                        if y in nfa[3]:
                                F.append(lazySet.index(x))
                                break
        print("Final States of DFA : ",end="")
        F.sort()
        for i in range(len(F)-1):
                print(F[i],end=",")
        print(F[len(F)-1])

global s
global n
global sigma
n=int(input("Enter number of states in the NFA:"))
Q=[i for i in range(n)]
s=int(input("Enter number of input symbols:"))
print("State",end="\t")
x=('a','b','c','d','e','f','g','h','i','j')
sigma=[]
for i in range(s):
        print(x[i],end="\t")
        sigma.append(x[i])
print()
```

```
nfa=[]
nfa.append(Q)
delta=[]
for i in range(n):
        print(i,end="\t")
        t=input().split()
        delta.append(t)
m=input("Enter the start states of the NFA:").split(",")
f=input("Enter the final states of the NFA:").split(",")
S=[]
F=[]
for i in range(len(m)):
        S.append(int(m[i]))
for i in range(len(f)):
        F.append(int(f[i]))
nfa.append(S)
nfa.append(delta)
nfa.append(F)
ConvertToDFA(nfa)
```

SAMPLE OUTPUT

```
user@adithya-d-rajagopal:~/s7/cd$ python3 p6.py
Enter number of states in the NFA:3
Enter number of input symbols:2
State   a       b
0       0,2     1
1       1,2     2
2       0,2     1
Enter the start states of the NFA:0
Enter the final states of the NFA:0,2

Generating DFA...

Start state of DFA : 0

STATE   a       b
0       1       2
1       1       2
2       3       4
3       5       3
4       1       2
5       5       3

Final States of DFA : 0,1,3,4,5
user@adithya-d-rajagopal:~/s7/cd$
```

RESULT

A program to convert NFA to DFA has been implemented using Python and the outputs have been verified.