Automata Theory Assignment 1

Report

The following code takes a Non-Deterministic Finite Automaton (NFA) and converts it to a Deterministic Finite Automaton (DFA).

Reading input.json and Setting up DFA

The following snippet of code reads the input.json file and opens it in a variable called nfa. The nfa is used to set up the basic values of the DFA like number of states, etc in a new variable called dfa.

```
''' Reading file '''
with open('input.json', 'r') as inpjson:
    nfa = json.loads(inpjson.read())

''' Setting up DFA '''
dfa={}
numstates=1
for i in range(nfa["states"]):
    numstates=numstates*2
dfa['states']=numstates
dfa['letters']=nfa['letters']
dfa['t_func']=[]
set1=[]
states = nfa['states']
for i in range(nfa['states']):
    set1.append(i)
```

Creating Power set

```
''' Powerset Functions '''
def power_set(nfa):
    powerset=[[]]
    for i in nfa:
        for sub in powerset:
            powerset=powerset+[list(sub)+[i]]
    return powerset
```

The powerset function is used to create all the states of the DFA. It is a simple function iterating through the sets of the NFA and appending them to the powerset.

Converting NFA relation to DFA function

The function takes each character of the alphabet, goes through the states of each relation for that character and appends the union of these states to form the function in the DFA.

```
''' Converting NFA relation to DFA function'''
for inp in nfa['letters']:
    for states in set2:
        temp=[]
        for state in states:
            for func in nfa['t_func']:
                if state==func[0] and inp==func[1]:
                    for i in func[2]:
                        temp.append(i)
        dfa['t_func'].append([states,inp,temp])
```

Setting DFA's start and final states and writing to output.json

The start state of the NFA and DFA remain same. The final states of the DFA are any states in the DFA which contain a final state of the NFA. The dictionary is then written to output.json.

```
''' setting DFA start and final states '''
dfa['start']=nfa['start']
dfa['final']=[]
for states in set2:
    for state in states:
        if state in nfa['final'] and states not in dfa['final']:
            dfa['final'].append(states)

''' Writing to output.json'''
with open('output.json', 'w') as outjson:
    outjson.write(json.dumps(dfa,indent=4))
```