

A dark blue vertical bar runs down the left side of the page. A blue arrow points to the right from this bar, containing the date.

1/22/2019

Lab Report

CSE 2206

Yamin Haque

ROLL: 1603007

SECTION: A

Several thin, curved lines in dark blue and light grey originate from the bottom left and sweep upwards and to the right, creating a dynamic, abstract design.

Theory:

We extend the class of NFAs by allowing instantaneous (ϵ) transitions:

1. The automaton may be allowed to change its state without reading the input symbol.
2. In diagrams, such transitions are depicted by labeling the appropriate arcs with ϵ .
3. Note that this does not mean that ϵ has become an input symbol. On the contrary, we assume that *the symbol ϵ does not belong to any alphabet*.

ϵ -NFAs add a convenient feature but (in a sense) they bring us nothing new: they do not extend the class of languages that can be represented. Both NFAs and ϵ -NFAs recognize exactly the same languages.

ϵ -transitions are a convenient feature: try to design an NFA for the even or divisible by 3 language that does not use them!

Exercise 2.5.1: Consider the following ϵ -NFA.

	ϵ	a	b	c
$\rightarrow p$	\emptyset	$\{p\}$	$\{q\}$	$\{r\}$
q	$\{p\}$	$\{q\}$	$\{r\}$	\emptyset
$*r$	$\{q\}$	$\{r\}$	\emptyset	$\{p\}$

- a) Compute the ϵ -closure of each state.
- b) Give all the strings of length three or less accepted by the automaton.

Code:

```
#include<bits/stdc++.h>
using namespace std;
int main()
{
    string s;
    cout<<"Input a string consists of a,b,c and e (as an epsilon) : ";
    cin>>s;
    int flag=0,i,m;
    int l=s.size();
    vector<int>v;
    int c=0;
    v.push_back(0);

    for( i=0; i<l; i++)
    {
        if(s[i]=='a'&& c==0)
        {
            v.push_back(0);
        }
        else if(s[i]=='b'&& c==0)
        {
            v.push_back(1);
        }
    }
}
```

```
        c=1;

    }
    else if(s[i]=='c'&& c==0)
    {
        v.push_back(2);
        c=2;
        flag=1;
    }
    else if(s[i]=='e'&& c==0)
    {
        v.push_back(0);

    }

    else if(s[i]=='a'&& c==1)
    {
        v.push_back(1);

    }
    else if(s[i]=='b'&& c==1)
    {
        v.push_back(2);
        c=2;
        flag=1;
    }

    else if(s[i]=='e'&& c==1)
    {
        v.push_back(1);
        c=0;
    }


    else if(s[i]=='a'&& c==2)
    {
        v.push_back(2);

    }
    else if(s[i]=='e'&& c==2)
    {
        v.push_back(1);
        c=1;
    }
    else if(s[i]=='c'&& c==2)
    {
        v.push_back(0);
        c=0;
    }
}
```

```

    }
    m=v.size();

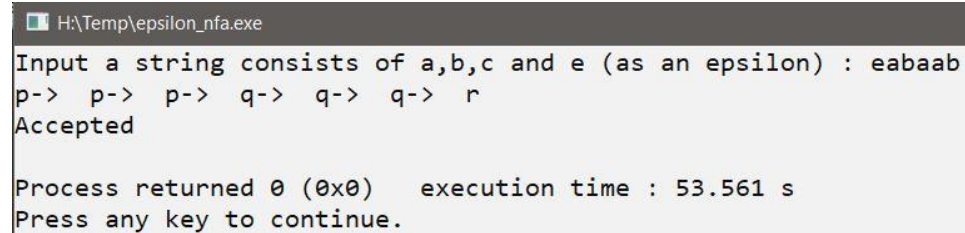
    for(i=0;i<m;i++)
    {
        if(v[i]==0)
            cout<<"p->";
        else if(v[i]==1)
            cout<<"q->";
        else if(v[i]==2)
            cout<<"r->";
    }
    cout<<endl;

    if(flag==1)
        cout<<"Accepted"<<endl;
    else
        cout<<"Not Accepted"<<endl;

    return 0;
}

```

Input & Output:



The screenshot shows a Windows command prompt window with the title "H:\Temp\epsilon_nfa.exe". The user has entered the string "eabaab" as input. The program's output shows the string being processed character by character, with 'p' for 'e', 'q' for 'a', and 'r' for 'b'. The final output is "Accepted". Below this, the program reports "Process returned 0 (0x0)" and "execution time : 53.561 s", followed by the prompt "Press any key to continue.".

```

H:\Temp\epsilon_nfa.exe
Input a string consists of a,b,c and e (as an epsilon) : eabaab
p-> p-> p-> q-> q-> q-> r
Accepted

Process returned 0 (0x0)   execution time : 53.561 s
Press any key to continue.

```