

Automata Theory and Formal Language

Lab Session 4

Carlos Gallardo Polanco

mail@mail.com

<https://github.com/gpcarlos95/TALFPractice>

November 26, 2017

1 Exercises

Given the RE $E = (((00) * + (00) * 0)10 + ((11) * + (11) * 1)10) *$

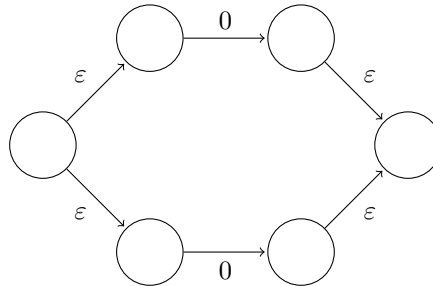
1. Build a ε - NFA that accepts exactly the same language, following the method explained in class to convert from RE to ε - NFA .
2. Generate the equivalent DFA
3. Implement it in a programming language (Python, C/C++, Java) following the table method.

2 Solutions

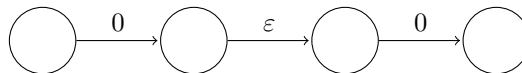
1. Build a ε - NFA that accepts exactly the same language, following the method explained in class to convert from RE to ε - NFA .

To convert a RE to an ε - NFA we'll follow the following equivalences:

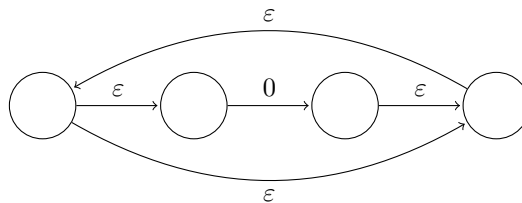
- Union.
If $E1 = 0$ and $E2 = 0$ then $E1 \cup E2$ ($E1 + E2$):



- Concatenation.
If $E1 = 0$ and $E2 = 0$ then $E1E2$:



- Closure.
If $E = 0$ then $E*$:

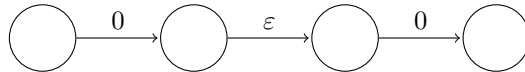


We'll divide the main RE into smaller REs :

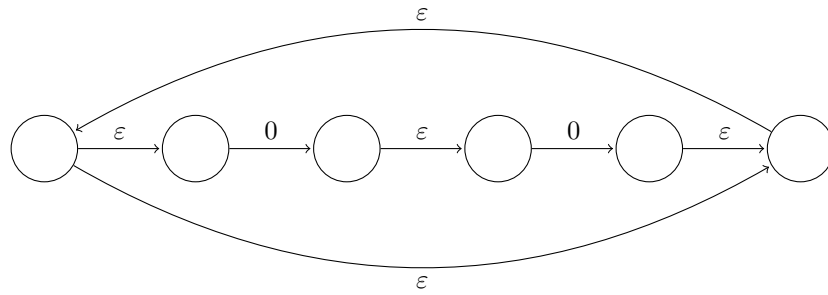
$$E = (((\underbrace{(00)}_{E1})^* + (\underbrace{(00)}_{E1})^* 0)10 + ((\underbrace{(11)}_{E2})^* + (\underbrace{(11)}_{E2})^* 1)10)^*$$

$\underbrace{\hspace{10em}}_{E3}$
 $\underbrace{\hspace{10em}}_{E4}$
 $\underbrace{\hspace{10em}}_{E5}$
 $\underbrace{\hspace{10em}}_{E6}$
 $\underbrace{\hspace{10em}}_{E7}$
 $\underbrace{\hspace{10em}}_{E8}$
 $\underbrace{\hspace{10em}}_{E9}$
 $E9^* = E$

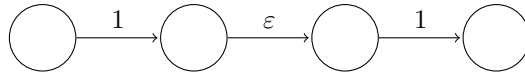
- $E1 = 00$



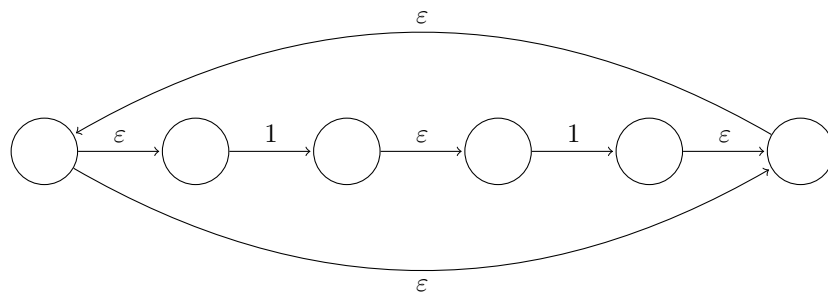
- $E1^* = (00)^*$



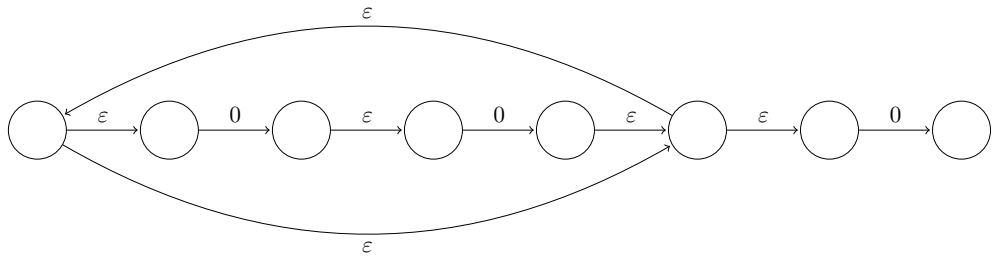
- $E2 = 11$



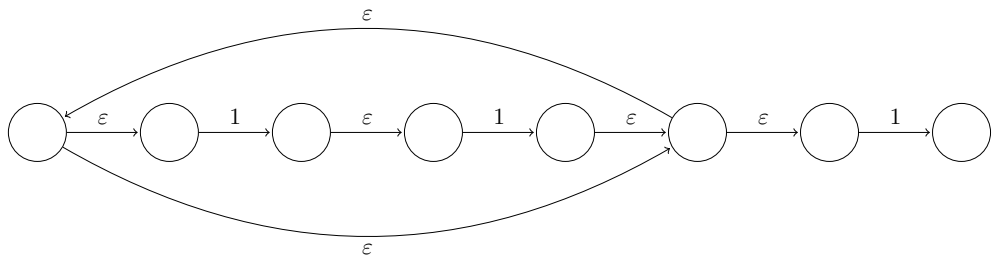
- $E2^* = (11)^*$



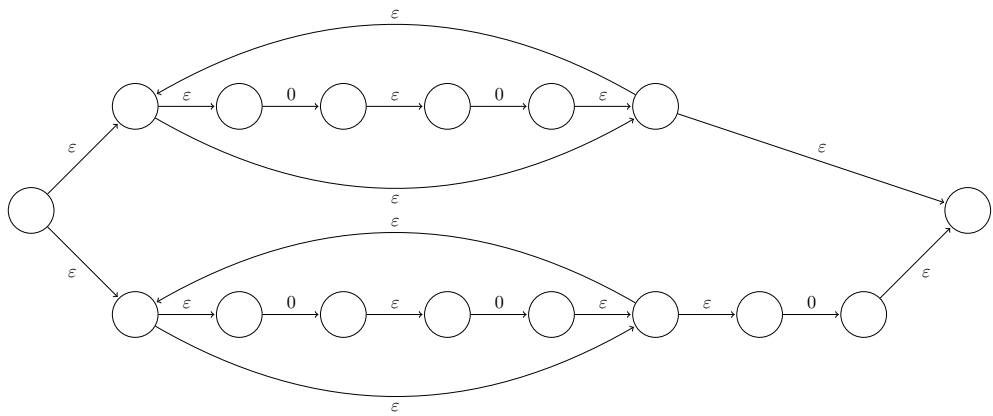
- $E3 = (00) * 0$



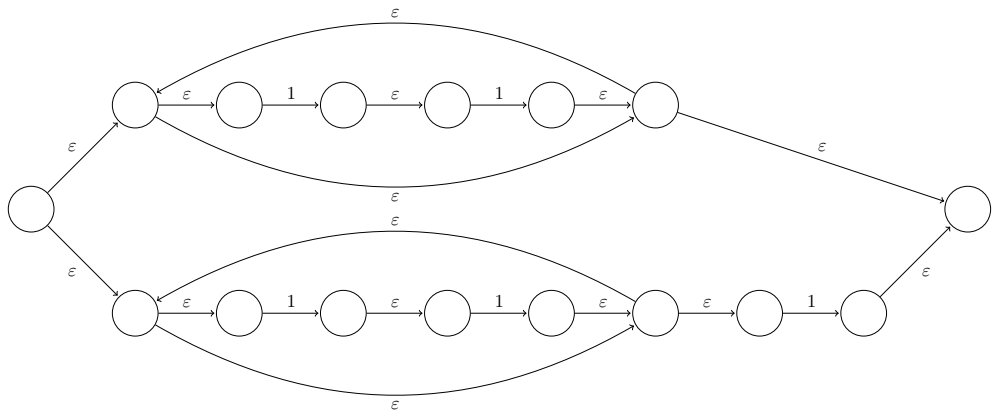
- $E4 = (11) * 1$



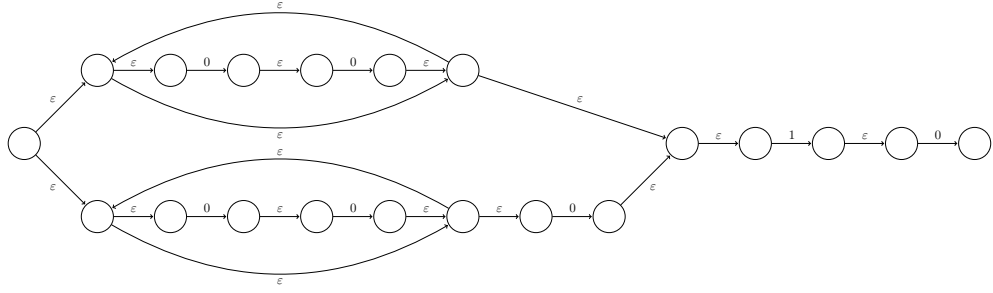
- $E5 = (00) * +(00) * 0$



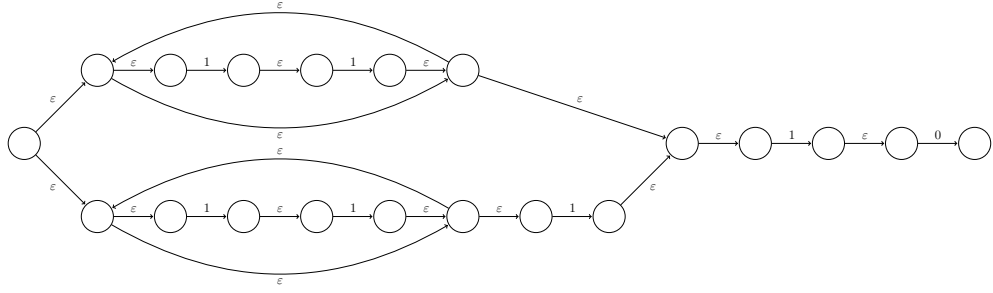
- $E6 = (11) * +(11) * 1$



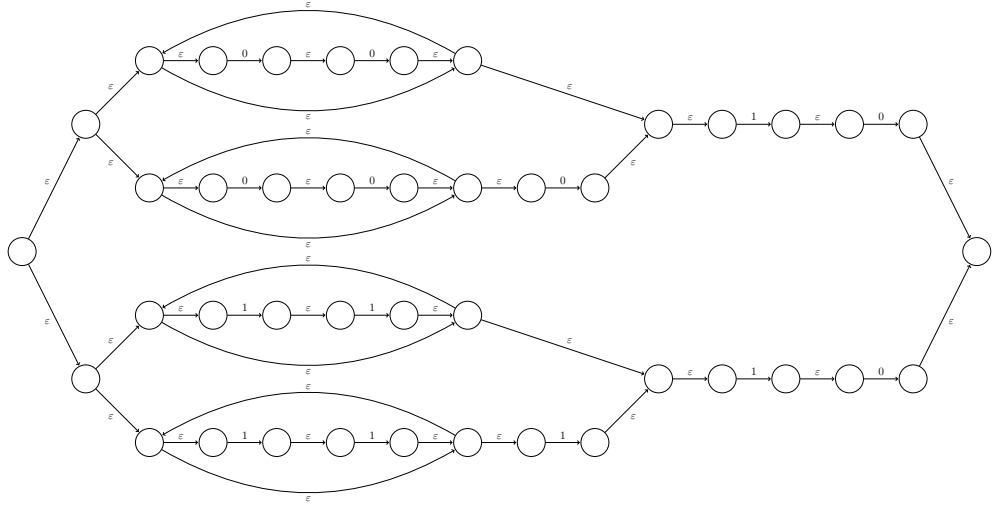
- $E7 = ((00) * +(00) * 0)10$



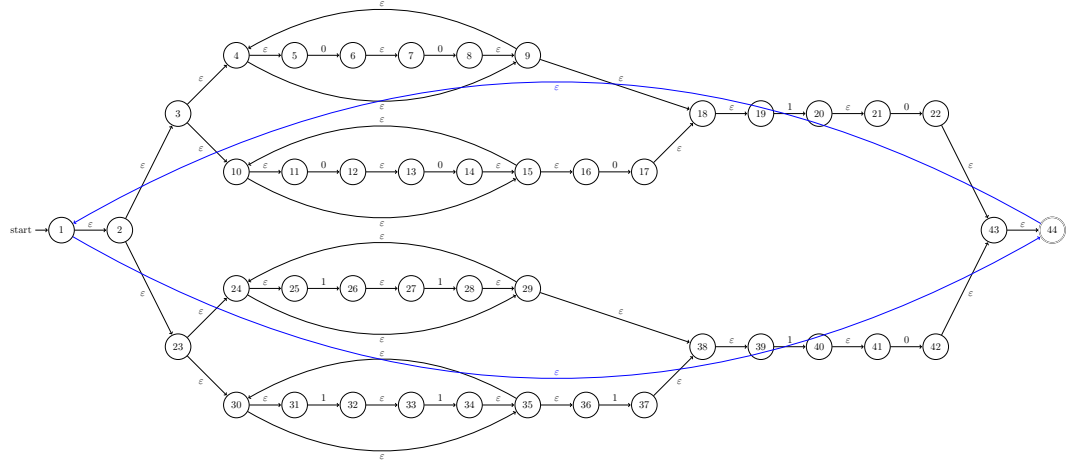
- $E8 = ((11) * +(11) * 1)10$



- $E9 = (((00) * +(00) * 0)10 + ((11) * +(11) * 1)10)$



- $E9^* = (((00)^* + (00)^* 0)10 + ((11)^* + (11)^* 1)10)^* = E$



2. Generate the equivalent DFA

To generate the equivalent DFA, we'll follow the **subset construction method**, so we'll need the equivalent NFA. In the table 1(a) we have the transition table of $\mathcal{E}-NFA$ and in the table 1(b) the closures.

Table 1: Transition Table of $\mathcal{E}-NFA$

(a) Transition Table of $\mathcal{E}-NFA$

States	0	1	ε
1			2 44
2			3 23
3			4 10
4			5 9
5	6		
6			7
7	8		
8			9
9			4 18
10			11 15
11	12		
12			13
13	14		
14			15
15			10 16
16	17		
17			18
18			19
19		20	
20			21
21	22		
22			43
23			24 30
24			25 29
25		26	
26			27
27		28	
28			29
29			24 38
30			31 35
31		32	
32			33
33		34	
34			35
35			30 36
36		37	
37			38
38			39
39		40	
40			41
41	42		
42			43
43			44
44			1

(b) Closures of $\mathcal{E}-NFA$

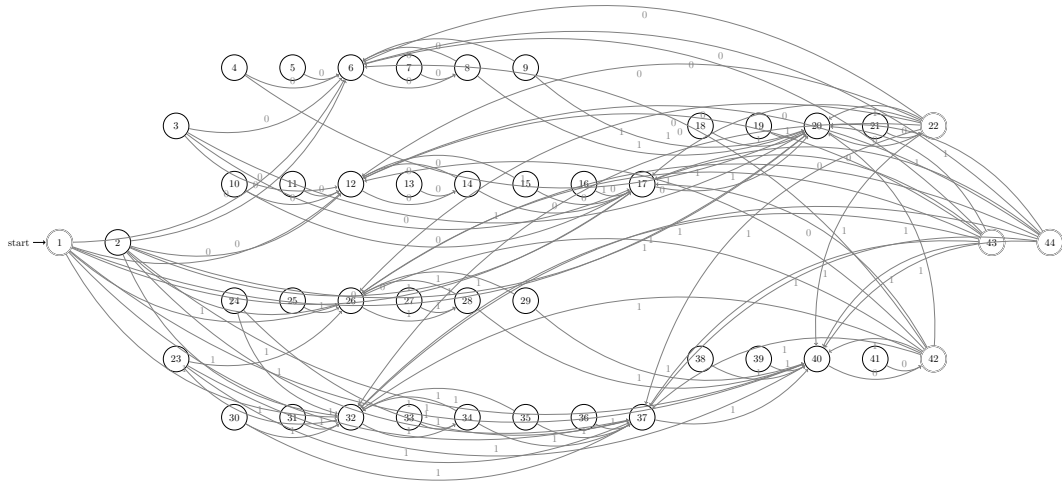
States	Closures
→ *	1 1 2 44 3 23 4 10 24 30 5 9 18 19 11 15 25 29 31 35 16 38 36 39
	2 2 3 23 4 10 24 30 5 9 11 15 25 29 31 35 18 16 38 36 19 39
	3 3 4 10 5 9 11 15 18 16 19
	4 4 5 9 18 19
	5 5
	6 6 7
	7 7
	8 8 9 4 18 5 19
	9 9 4 18 5 19
	10 10 11 15 16
	11 11
	12 12 13
	13 13
	14 14 15 10 16 11
	15 15 10 16 11
	16 16
	17 17 18 19
	18 18 19
	19 19
	20 20 21
	21 21
*	22 22 43 44 1 2 3 23 4 10 24 30 5 9 18 19 11 15 25 29 31 35 16 38 36 39
	23 23 24 25 29 38 39 30 31 35 36
	24 24 25 29 38 39
	25 25
	26 26 27
	27 27
	28 28 29 24 38 25 39
	29 29 24 38 25 39
	30 30 31 35 36
	31 31
	32 32 33
	33 33
	34 34 35 30 36 31
	35 35 30 36 31
	36 36
	37 37 38 39
	38 38 39
	39 39
	40 40 41
	41 41
*	42 42 43 44 1 2 3 23 4 10 24 30 5 9 18 19 11 15 25 29 31 35 16 38 36 39
*	43 43 44 1 2 3 23 4 10 24 30 5 9 18 19 11 15 25 29 31 35 16 38 36 39
*	44 44 1 2 3 23 4 10 24 30 5 9 18 19 11 15 25 29 31 35 16 38 36 39

Removing the ε -transition, we obtain the transition table 2 of NFA.

States	0	1
→ *	6 12 17	20 26 32 37 40
2	6 12 17	20 26 32 37 40
3	6 12 17	20
4	6	20
5	6	
6	8	
7	8	
8	6	20
9	6	20
10	12 17	
11	12	
12	14	
13	14	
14	12 17	
15	12 17	
16	17	
17		20
18		20
19		20
20	22	
21	22	
* 22	6 12 17	20 26 32 37 40
23		26 32 37 40
24		26 40
25		26
26		28
27		28
28		26 40
29		26 40
30		32 37
31		32
32		34
33		34
34		32 37
35		32 37
36		37
37		40
38		40
39		40
40	42	
41	42	
* 42	6 12 17	20 26 32 37 40
* 43	6 12 17	20 26 32 37 40
* 44	6 12 17	20 26 32 37 40

Table 2: Transition Table of NFA

And the NFA:



Following the **subset construction method**, we obtain the table 3(a) and we can rename the states like in the table 3(b).

Table 3: Transition Table of DFA

(a) Transition Table of DFA

	States	0	1
→ * A	1	6.12.17	20.26.32.37.40
B	6.12.17	8.14	20
C	20.26.32.37.40	22.42	28.34.40
D	8.14	6.12.17	20
E	20	22	Dead
* F	22.42	6.12.17	20.26.32.37.40
G	28.34.40	42	26.40.32.37
* H	22	6.12.17	20.26.32.37.40
* I	42	6.12.17	20.26.32.37.40
J	26.40.32.37	42	28.34.40
	Dead	Dead	Dead

(b) Transition Table of DFA (Renamed)

	States	0	1
→ *	A	B	C
	B	D	E
	C	F	G
	D	B	E
	E	H	Dead
*	F	B	C
	G	I	J
*	H	B	C
*	I	B	C
	J	I	G
	Dead	Dead	Dead

Following the **state minimization algorithm** for DFAs, we obtain the table 4 where **X** are all initial pairs of distinguishable states, **X** are all pairs of distinguishable states in the first round and **O** are all pairs of equivalent states.

			*	*	*					*	→
	Dead	J	I	H	G	F	E	D	C	B	A
→ *	A	X	X	X	X	X	X	X	X	X	X
	B	X	X	X	X	X	X	O	X		
	C	O	X	X	O	X	X	X			
	D	X	X	X	X	X	X				
	E	X	X	X	X	X					
*	F	X	O	O	X						
	G	O	X	X							
*	H	X	O								
*	I	X									

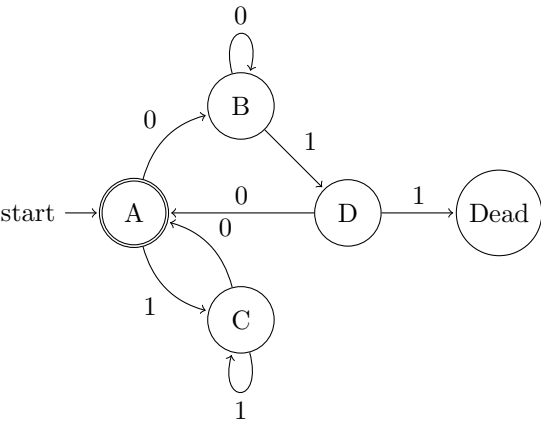
Table 4: Minimization Table of DFA

The Minimum-state DFA is:

Table 5: Minimum-state Table of DFA

(a) Minimum-state Table of DFA			
→ *	States	0	1
	A_I_H_F	B_D	C_J_G
	B_D	B_D	E
	E	A_I_H_F	Dead
	C_J_G	A_I_H_F	C_J_G
	Dead	Dead	Dead

(b) Minimum-state Table of DFA (Renamed)			
→ *	States	0	1
	A	B	C
	B	B	D
	D	A	Dead
	C	A	C
	Dead	Dead	Dead



3. Implement it in a programming language (Python, C/C++, Java) following the table method.

```
DFA = ((1, 2), # Transition table
        (1, 3),
        (0, 2),
        (0, 4),
        (4, 4))

file = open("inputTestCases.txt","r")

for line in file:
    state = 0
    for char in line:
        if ord(char)!=10: # To avoid the 'enter'
            if char=='0': c=0
            else: c=1
            state = DFA[state][c]

    if state==0:
        print(line," Accept")
    else:
        print(line," Reject")
```